
Содержание главы 19

19	<u>Параметры блоков</u>	4
19.1	<u>Общая информация о параметрах блоков</u>	4
19.1.1	<u>Определение параметров блоков</u>	4
19.1.2	<u>Обработка параметров блоков</u>	4
19.1.3	<u>Описание параметров блоков</u>	5
19.1.4	<u>Описание значения функции</u>	7
19.1.5	<u>Инициализация параметров блоков</u>	8
19.2	<u>Формальные параметры</u>	9
19.3	<u>Фактические параметры</u>	13
19.4	<u>«Передача» параметров блоков</u>	18
19.5	<u>Примеры</u>	19
19.5.1	<u>Пример «Ленточный транспортер конвейера»</u>	19
19.5.2	<u>Пример «Счетчик деталей»</u>	20
19.5.3	<u>Пример «Устройство подачи»</u>	21

19 Параметры блоков

В настоящей главе рассказывается о том, как использовать параметры блоков. Вы научитесь

- описывать параметры блоков,
- работать с параметрами блоков,
- инициализировать параметры блоков и
- «передавать» параметры блоков.

Параметры блоков представляют интерфейс передачи между вызывающим и вызываемым блоками. Все функции и функциональные блоки могут работать с параметрами.

19.1 Общая информация о параметрах блоков

19.1.1 Определение параметров блоков

Параметры блока дают возможность параметризовать обрабатываемую в блоке инструкцию, функцию блока. Пример: Вам требуется создать блок, суммирующий значения, и который вы сможете использовать в программе несколько раз с различными переменными. Переменные будут передаваться в качестве параметров блока; в нашем примере три входных параметра и один выходной параметр (рисунок 19.1).

Так как сумматору не нужно хранить внутренних значений, в качестве типа блока подойдет функция.

Вы определяете параметр блока как входной (input parameter), если в программе блока значение этого параметра только считывается или загружается. Если вы только описываете параметр блока (назначаете, устанавливаете, сбрасываете, пересылаете), то вы используете выходной параметр (output parameter).

Вы должны всегда использовать входной/выходной параметр (in/out parameter), если он должен и считываться, и перезаписываться. Программный редактор не проверяет использование параметров блоков.

19.1.2 Обработка параметров блоков

В программе сумматора имена параметров блока применяются как ячейки для последних фактических переменных. Параметры блоков используются так же, как и символически адресованные переменные; в программе они называются формальными параметрами (formal parameters).

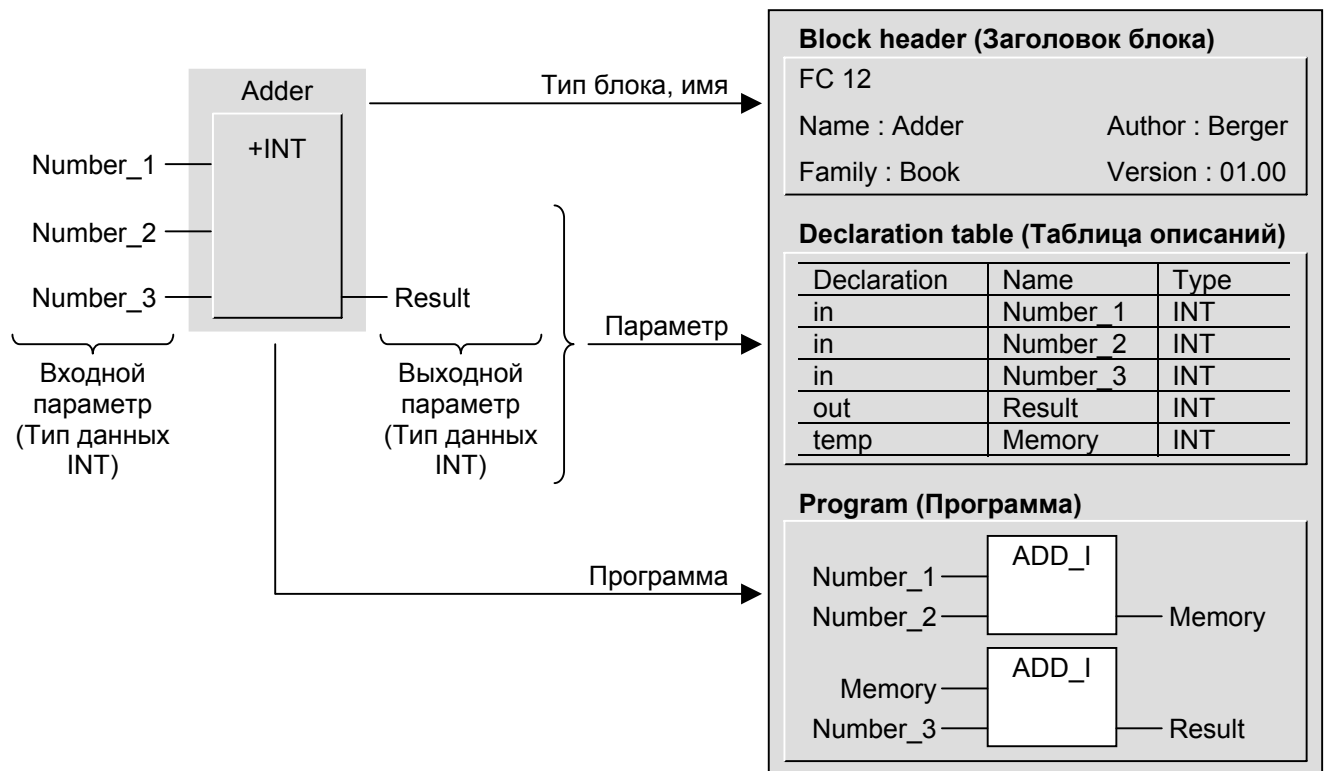


Рисунок 19.1 Пример параметров блока

В программе вы можете вызвать функцию «Adder» («Сумматор») несколько раз. При каждом вызове вы можете передавать другие значения в сумматор в параметрах блока (рисунок 19.2). Значения могут быть константами, операндами или переменными; они называются фактическими параметрами (actual parameters).

Во время исполнения программы CPU заменяет формальные параметры фактическими параметрами. В примере первый вызов складывает содержимое слов памяти MW 30, MW 32 и MW 34 и сохраняет результат в слове памяти MW 40. Тот же самый блок с фактическими параметрами при втором вызове суммирует слова данных DBW 30, DBW 32 и DBW 34 блока данных DB 13 и сохраняет результат в слове данных DBW 40 блока данных DB 14.

19.1.3 Описание параметров блоков

Параметры блока определяются в разделе описаний блока при его программировании. Ниже показана пустая таблица описаний (declaration table). Кроме параметров блока – in (входной), out (выходной), in_out (входной/выходной) – в этой таблице вы также опишите статические локальные данные (stat) и временные локальные данные (temp).

Значения по умолчанию являются необязательными и применяются только в практических целях для функциональных блоков, если параметр блока сохранен как значение. Это относится ко всем параметрам блоков простых типов данных и к входным и выходным параметрам сложных типов данных.

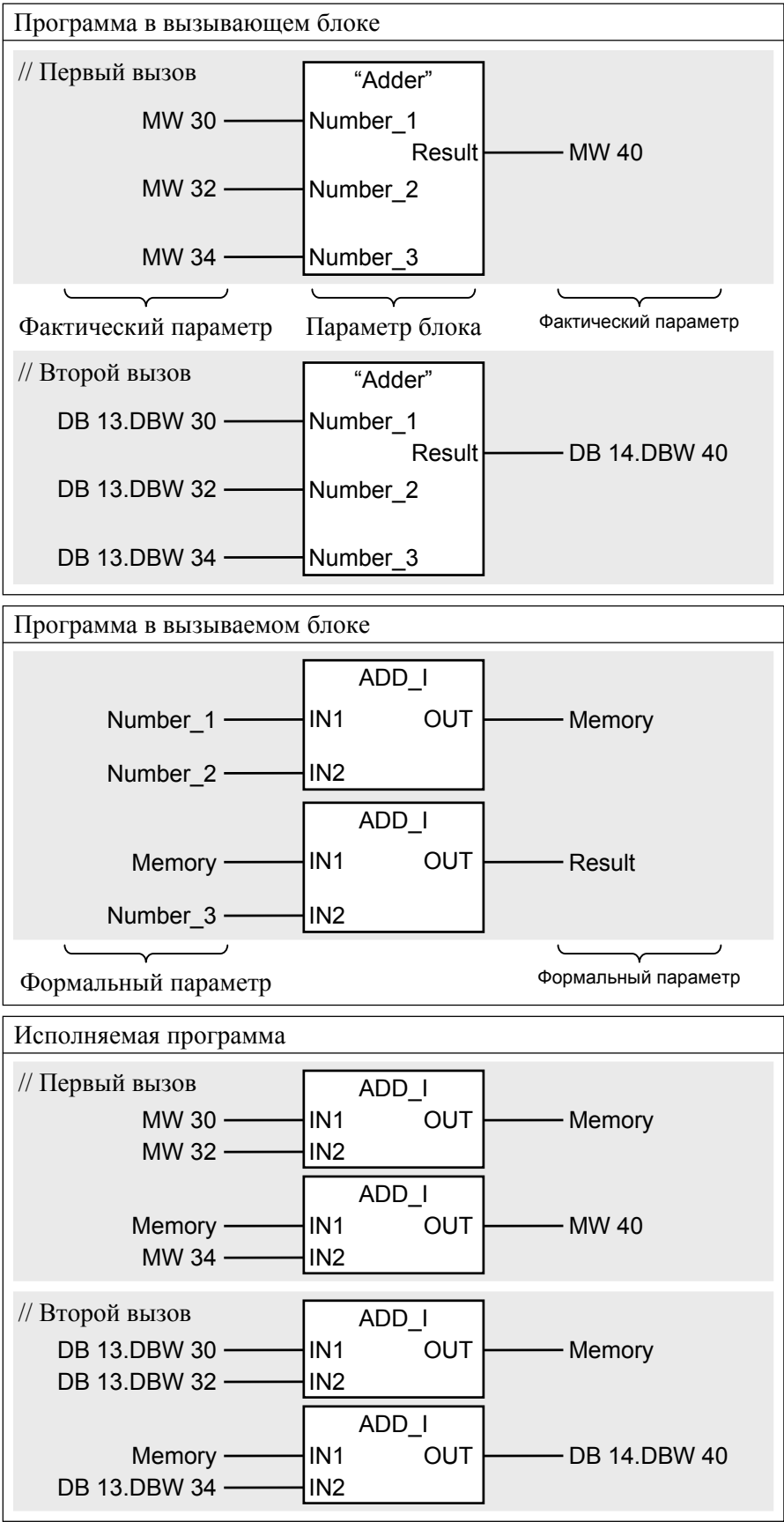


Рисунок 19.2 Вызов блока с параметрами

Таблица 19.1 Пустая таблица описаний

Address (Адрес)	Declaration (Описание)	Name (Имя)	Type (Тип)	Initial value (Начальное значение)	Comment (Комментарий)
	in				Input parameter (for FCs and FBs) / Входной параметр (для FC и FB)
	out				Output parameter (for FCs and FBs) / Выходной параметр (для FC и FB)
	in_out				In/out parameter (for FCs and FBs) / Входной/выходной параметр (для блоков FC и FB)
	stat				Static local data (for FBs) / Статические локальные данные (для FB)
	temp				Temporary local data (for OBs, FCs and FBs) / Временные локальные данные (для OB, FC и FB)

Также могут быть указаны комментарии для параметров.

Имя параметра блока может состоять из 24 знаков максимум. Имя может включать в себя только буквы (кроме национальных литер, таких как немецкий умлаут), цифры и знак подчеркивания. Между верхним и нижним регистром различия не делается. Имя не должно быть ключевым словом.

Для *типа данных* параметра блока допустимы все простые, сложные и определенные пользователем типы данных, а также параметрические типы (см. параграф 3.5 «Переменные, константы и типы данных»).

STEP 7 хранит имена параметров блоков в неисполняемом разделе блоков на носителе информации программирующего устройства. Пользовательская память CPU (в скомпилированном блоке) содержит только типы описаний (declaration types) и типы данных. Поэтому изменения программы, произведенные с блоками в онлайн-режиме в CPU, должны быть отражены в среде хранения программирующего устройства для того, чтобы сохранить исходные имена. Если обновление не осуществлено, или если блоки переданы из CPU в устройство программирования, то неисполняемые разделы блоков перезаписываются или стираются. Программный редактор генерирует символы замены для вывода на экран или при печати.

19.1.4 Описание значения функции

В функциях их значение – особым образом обрабатываемый выходной параметр. Он имеет имя RET_VAL (или ret_val) и определяется как первый выходной параметр. Вы должны объявить значение функции путем назначения имени RET_VAL первому выходному значению в таблице описаний.

В качестве типа данных значения функции разрешено использовать все простые типы, а также типы DATE_AND_TIME, STRING, POINTER, ANY и определенные пользователем типы данных UDT. Типы ARRAY и STRUCT не допускаются.

Как первый выходной параметр, значение функции не играет особой роли в языках программирования LAD и FBD. Он приобретает значение только в языке программирования STL, где тип блока FUNCTION (Функция) используется как «подлинная» функция. Здесь же функция FC может указываться вместо операнда при выводе; в этом случае значение функции представляет собой значение этой функции.

19.1.5 Инициализация параметров блоков

При вызове блока с параметрами вы должны инициализировать параметры блока фактическими параметрами. Это могут быть константы, операнды с абсолютными адресами, полностью адресованные операнды данных или символически адресованные переменные. Фактический параметр должен быть того же типа данных, что и параметр блока.

Каждый параметр функции должен быть инициализирован при каждом вызове. В случае функционального блока инициализация отдельных параметров блока или всех параметров необязательна.

19.2 Формальные параметры

В этом разделе вы узнаете о способах доступа к параметрам внутри блока. Вы также узнаете, что доступ к параметрам блоков простых типов данных, элементам массива или компонентам структуры, таймерам и счетчикам возможен без ограничения. Параграф 19.4 ««Передача» параметров блока» показывает вам, как можно «передать» параметры в вызываемые блоки.

Обращение к параметрам сложных типов данных и параметрических типов POINTER и ANY в LAD или FBD не поддерживается. Тем не менее, вы можете инициализировать готовые или системные блоки, которые имеют такие параметры, используя соответствующую переменную. На дискете, прилагаемой к книге, имеются примеры по этой теме в программе «Data Types» («Типы данных»).

Таблица 19.2 Доступ к параметрам блоков (обобщение)

Типы данных	Разрешено для			Доступ в блоке возможен
	IN	I_O	OUT	
Простые типы данных				
BOOL	x	x	x	да
BYTE, WORD, DWORD, CHAR, INT, DINT, REAL, S5TIME, TOD, DATE	x	x	x	да
Сложные типы данных				
DT, STRING	x	x	x	нет
ARRAY, STRUCT				
Отдельные бинарные компоненты	x	x	x	да
Отдельные числовые компоненты	x	x	x	да
Переменные полностью	x	x	x	нет
Параметрические типы				
TIMER	x	-	-	да
COUNTER	x	-	-	да
BLOCK_FC, BLOCK_FB	x	-	-	да
BLOCK_DB	x	-	-	да
BLOCK_SDB	x	-	-	нет
POINTER, ANY	x	x	x ¹⁾	нет

¹⁾ только функции FC

Параметры блока типа данных BOOL

Параметры блока типа BOOL могут быть отдельными двоичными переменными или бинарными компонентами массивов и структур. Вы можете считать входные параметры и входные/выходные параметры с помощью контактов или двоичных входов

блочных элементов, и можно воздействовать на выходные параметры и входные/выходные параметры, используя двоичные выходы блочных элементов.

После того, как CPU использовал фактический параметр, определенный в качестве параметра блока, он обрабатывает функции, как показано в соответствующих главах.

Параметры блоков числового типа данных

Параметры блока числового типа занимают 8, 16 или 32 бита (все простые типы данных за исключением BOOL). Они могут быть отдельными числовыми переменными или числовыми компонентами массивов и структур.

Вы можете подавать входные и входные/выходные параметры на цифровые входы блочных элементов, и можно записывать выходные и входные/выходные параметры в цифровые входы блочных элементов.

Обмен значениями между параметрами блоков различных типов и разных размеров может быть произведен с использованием блочного элемента MOVE, как это описано в главе 6 «Функции передачи».

Параметры блоков типов данных DT и STRING

Прямой доступ к параметрам блоков типов DT и STRING невозможен. В функциональных блоках вы можете «передать» входные и выходные параметры типов DT и STRING параметрам вызванного блока.

Параметры блоков типов данных ARRAY и STRUCT

Возможно прямое покомпонентное обращение к параметрам блоков типов ARRAY и STRUCT, то есть вы можете обратиться к отдельным двоичным или числовым компонентам с помощью соответствующих операций.

Доступ ко всей переменной (ко всему массиву или всей структуре) невозможен, как и доступ к отдельным компонентам сложных или определенных пользователем типов данных. В функциональных блоках вы можете передать входные и выходные параметры типов ARRAY и STRUCT параметрам вызванного блока.

Параметры блоков определенного пользователем типа данных

Обработка параметров блока определенного пользователем типа данных осуществляется тем же способом, что и параметры блока типа STRUCT.

Возможен прямой покомпонентный доступ к параметрам типа UDT, то есть вы можете обратиться к отдельным двоичным или числовым компонентам с использованием соответствующих операций.

Доступ ко всей переменной невозможен. Также нельзя обратиться к отдельным компонентам сложного или определенного пользователем типа данных.

В функциональных блоках вы можете «передать» входные и выходные параметры типа UDT в параметры вызываемого блока.

Параметры блоков типа TIMER

Вы можете использовать параметры блоков типа TIMER во всех функциях, как описано в главе 7 «Таймеры». При запуске таймера значение времени также может быть параметром блока типа S5TIME.

Параметры блоков типа COUNTER

Вы можете использовать параметры блоков типа COUNTER во всех функциях, как описано в главе 8 «Счетчики». При установке счетчика его значение может быть также параметром блока типа WORD.

Параметры блоков типа BLOCK_DB

Вы можете переслать блок данных через параметр блока типа BLOCK_DB. Вызовите этот блок данных с помощью катушки / блочного элемента OPN путем маркирования катушки / блочного элемента OPN формальными параметрами.

При открытии блока данных через параметр блока CPU всегда использует регистр глобального блока данных (регистр DB).

Параметры блоков типа BLOCK_FC

Можно передать функцию FC через параметр блока типа BLOCK_FC. Вызовите эту функцию при помощи катушки / блочного элемента CALL. Вы можете использовать катушку / блочный элемент CALL с формальным параметром и предшествующей логической операцией или без нее, если в данный момент вы программируете функциональный блок.

Если вы используете катушку / блочный элемент CALL с формальными параметрами в функции, то предшествующей логической операции не допускается (только абсолютный вызов).

Функция FC, передаваемая через параметр блока, не должна иметь параметров.

Параметры блоков типа BLOCK_FB

Вы можете передать функциональный блок FB через параметр блока типа BLOCK_FB. Прямой доступ к параметрам блоков типа BLOCK_FB с использованием функций LAD и FBD невозможен.

Функциональный блок FB, пересылаемый посредством параметра, не имеет параметров.

Параметры блоков типа POINTER и ANY

Прямой доступ к параметрам блока типа POINTER и ANY с использованием функций LAD и FBD невозможен.

19.3 Фактические параметры

Когда блок вызывается, вы инициализируете его параметры значениями констант, операндов или переменных, с которыми он будет работать. Эти параметры являются фактическими. Если в программе вы вызываете блок часто, то каждый раз обычно используются различные параметры.

Фактический параметр должен соответствовать по типу данных параметру блока. Для параметра блока типа `BOOL` вы можете применить только двоичный фактический параметр (например, меркерный бит); инициализировать параметр блока типа `ARRAY` вы можете только переменной, являющейся массивом с идентичной размерностью. Таблица 19.3 представляет обзор, какие операнды вы можете использовать в качестве фактических параметров и с какими типами данных.

Таблица 19.3 Инициализация фактическими параметрами

Тип данных параметра блока	Допустимые фактические параметры
Простой тип данных	➤ Простые операнды, полностью адресованные операнды данных, константы ➤ Элементы массивов или компоненты структур простого типа данных ➤ Параметры вызывающего блока ➤ Компоненты параметров вызывающего блока простого типа данных
Сложный тип данных	➤ Переменные или параметры вызывающего блока
<code>TIMER</code> , <code>COUNTER</code> и <code>BLOCK_xx</code>	➤ Таймеры, счетчики и блоки
<code>POINTER</code>	➤ Простые операнды, полностью адресованные операнды данных ➤ Интервальный указатель (<code>range pointer</code>) или <code>DB-</code>указатель
<code>ANY</code>	➤ Переменные любого типа данных ➤ Указатель <code>ANY</code>

При вызове функций вы должны инициализировать фактическими параметрами все параметры блоков.

Когда вызываются функциональные блоки, инициализация фактическими параметрами параметры блоков не является обязательным требованием. Если вы не определяете параметры блока, то в качестве фактических параметров используются (старые) значения, сохраненные в экземплярном блоке данных. Это могут быть, к примеру, значения по умолчанию или значения фактических параметров более раннего вызова. Входные/выходные параметры сложного типа данных не могут быть назначенными значениями по умолчанию и параметрическими типами. Вы должны обеспечить эти параметры блоков фактическими параметрами, по крайней мере, при первом вызове.

Вы также можете использовать прямой доступ для обращения к параметрам функционального блока. Так как они расположены в блоке данных, вы можете оперировать параметрами блока как операндами данных. Пример: Функциональный блок с экземпляром блока данных «*Station_1*» управляет бинарным выходным параметром с именем «*Up*». После обработки в функциональном блоке (после его вызова), вы можете считать параметр по символическому адресу «*Station_1*». *Up* в отсутствие инициализированного выходного параметра.

Инициализация параметров блоков с простыми типами данных

Фактические параметры, приведенные в таблице 19.4, допускаются в качестве фактических параметров простого типа данных.

Таблица 19.4 Фактические параметры простого типа данных

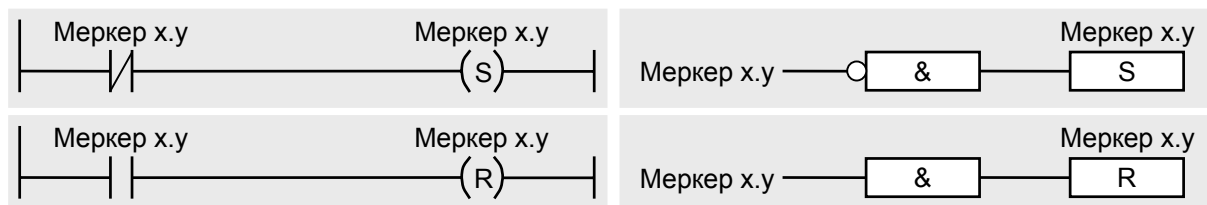
Операнды	Разрешено с			Двоичный операнд или символическое имя	Числовой операнд или символическое имя
	IN	I_O	OUT		
Входы (образ процесса)	x	x	x	I y.x	IB y, IW y, ID y
Выходы (образ процесса)	x	x	x	Q y.x	QB y, QW y, QD y
Меркерная память	x	x	x	M y.x	MB y, MW y, MD y
Периферийные входы	x	-	-		PIB y, PIW y, PID y
Периферийные выходы	-	-	x		PQB y, PQW y, PQD y
Глобальные данные					
Частичная адресация	x	x	x	-	DBB y, DBW y, DBD y
Полная адресация	x	x	x	DB z.DBX y.x	DB z.DBB y и т.д.
Временные локальные данные	x	x	x	L y.x	LB y, LW y, LD y
Статические локальные данные	x	x	x	DIX y.x	DIB y, DIW y, DID y
Константы	x	-	-	-	Все числовые константы
Компоненты ARRAY или STRUCT	x	x	x	Полное имя компонента	Полное имя компонента

x = номер бита, y = адрес байта, z = номер блока данных

Вы можете назначить операндам входа, выхода и меркера либо абсолютный, либо символический адрес. Входные операнды должны использоваться только для входных параметров, а выходные операнды – для выходных параметров (однако это не обязательно). Операнды памяти меркеров подходят для всех описанных в программе типов. Вы должны сочетать периферийные входы только с входными параметрами и периферийные выходы только с выходными параметрами.

В LAD и FBD булевские константы TRUE и FALSE используются для установки значений по умолчанию в описании параметров блоков, статических локальных данных или операндов данных. Инициализация параметров блоков этими константами не допускается. Вы также не можете создавать эти константы на контакте (LAD) или

на входе функции (FBD). Если вы хотите инициализировать параметры блока значениями TRUE или FALSE, то вам потребуется переменная, которой постоянно присвоено сигнальное состояние «1» или «0», и которую вы используете вместо булевских констант. Меркеры в этом случае являются несомненными кандидатами для переменных. Следующий пример демонстрирует вам, как можно устанавливать меркеры в сигнальное состояние «1» или «0» на постоянной основе, скажем при запуске.



Частично адресуемые биты данных в качестве фактических параметров в LAD и FBD не допускаются. Причиной этому служит тот факт, что программный редактор пересылает сигнальное состояние каждого двоичного фактического операнда в бит временных локальных данных перед фактическим вызовом. Поэтому в LAD и FBD возможно инициализировать двоичный вход блока, используя логическую операцию. Во время копирования перед вызовом блока назначение блока данным биту данных теряется. Даже если вы открываете «корректный» блок данных в программе вызванного блока перед обращением к параметру блока, занятому частично адресованным битом данных, то сканироваться будет только предварительно созданная копия сигнального состояния.

Это относится и к двоичным входным/выходным параметрам и выходным параметрам: здесь также назначение с частично адресованными битами данных не разрешается (редактор программ не отклонит такое назначение, но в большинстве случаев это приведет к ошибочным функциям).

Когда используются частично адресованные операнды данных для числовых параметров, вы должны для себя отметить, что при обращении к параметрам блока (в вызванном блоке) открытый в текущий момент блок данных также является «корректным» блоком данных. Так как редактор в определенных обстоятельствах (например, во время вызова блока или если ранее произошло обращение к параметрам блока) может незаметно изменить блок данных, использование частично адресованных числовых операндов данных не рекомендуется. По этой причине применяйте только полностью адресованные данные.

Временные локальные данные обычно адресуются символически. Они расположены в L-стеке вызывающего блока и описываются (объявляются) в вызывающем блоке.

Если вызывающий блок является функциональным блоком, вы можете также использовать его статические локальные данные в качестве фактических параметров (см. параграф 19.4 ««Передача» параметров блоков»). Статические данные обычно адресованы символически; если вы хотите присвоить абсолютный адрес через операнды DI, помните о сказанном в параграфе 18.2.4 «Особые замечания по адресации данных».

Имея параметр блока типа **BOOL**, можно применять константу **TRUE** (сигнальное состояние «1») или **FALSE** (сигнальное состояние «0»), а с использованием параметра блока числового типа данных вы можете применять все константы, соответствующие типу данных. Инициализация с применением констант допускается только для входных параметров.

Вы также можете инициализировать параметр блока простого типа данных элементами массивов или компонентами структур, если последние относятся к тому же типу данных, что и параметр блока.

Инициализация параметров блоков сложных типов данных

Все параметры блока могут быть сложного типа. Переменные того же типа могут использоваться в качестве фактических операндов.

Для инициализации параметров блоков типа **DT** или **STRING** допускаются отдельные переменные или компоненты массивов или структур того же типа.

Переменные **STRING** могут иметь различную длину. Если вы задали размер **STRING** при описании входного или выходного параметра функционального блока, то редактор резервирует указанное пространство в экземплярном блоке данных; если длина не указана, то для переменной **STRING** резервируется 256 байт. Максимальный размер переменной **STRING**, который вы задаете в разделе описаний, должен совпадать с размером фактического и формального параметров. Исключение: в случае функций **FC** при объявлении переменной **STRING** вы либо не задаете длину, либо указываете размер 254 байта; здесь в качестве фактических параметров вы можете использовать переменные **STRING** любой длины.

Для инициализации параметров блоков типа **ARRAY** или **STRUCT** допускаются переменные такой же структуры.

Инициализация параметров блоков определенного пользователем типа данных

Для сложных или громоздких структур данных рекомендуется использование определяемых пользователем типов данных (**UDT**). Сначала вы должны описать **UDT**, затем сможете использовать его, например, для генерирования переменной в блоке данных или объявления параметра блока. Впоследствии переменная может служить для инициализации параметра блока. Здесь также фактический параметр (переменная) должен быть того же типа данных (иметь тот же **UDT**), что и параметр блока.

Инициализация параметров блоков типов данных TIMER, COUNTER и BLOCK_xx

Параметр блока типа **TIMER** инициализируется при помощи таймера, а параметр блока типа **COUNTER** – счетчика. Для параметров блоков параметрических типов

BLOCK_FC и BLOCK_FB применяются блоки без их собственных параметров. Инициализация BLOCK_DB осуществляется с использованием блока данных.

Параметры блоков типов TIMER, COUNTER и BLOCK_хх должны быть входными параметрами.

Инициализация параметров блоков типа POINTER

Указатели (константы) и операнды допустимы для использования в качестве параметров блоков параметрического типа POINTER. Эти указатели являются либо интервальными указателями (range pointer), либо DB-указателями. В первом случае это 32-битные указатели, во втором – 48-битные. Операнды простых типов данных могут быть также полностью адресованными операндами данных.

Для функциональных блоков выходные параметры типа POINTER не допускаются.

Инициализация параметров блоков типа ANY

В качестве параметров блоков параметрического типа ANY могут использоваться переменные всех типов. При программировании вызываемого блока определяется, какие переменные (операнды или типы данных) должны быть применены к параметрам блока или какие переменные допустимы. Вы также можете определить константу в формате ANY-указателя “P#[data block.]Operand Datatype Number” (“P#[блок данных.]Операнд Тип данных Номер”) и определить таким образом абсолютно адресованную область.

Для функциональных блоков выходные параметры типа ANY не могут быть использованы.

19.4 «Передача» параметров блоков

«Передача» параметров блоков – это особая форма доступа к параметрам блоков и их инициализации. Параметры вызывающего блока «передаются» в параметры вызываемого блока. Формальный параметр вызывающего блока, таким образом, становится фактическим параметром вызванного блока.

Вообще, в этом случае также тип фактического параметра должен совпадать с типом формального параметра (то есть соответствующие параметры блока должны совпадать по их типам). Кроме того, вы можете применить входной параметр вызывающего блока только к входному параметру вызываемого блока и, аналогично, выходной параметр к выходному параметру. Входной/выходной параметр вызывающего блока вы можете применить ко всем типам описания вызываемого блока.

Имеются ограничения, связанные с типами данных, которые вызваны различиями в хранении параметров блоков функциями и функциональными блоками. Параметры блоков простого типа данных могут передаваться беспрепятственно в соответствии с информацией предыдущего параграфа. Входные и выходные параметры сложного типа могут быть переданы, только если вызывающий блок является функциональным блоком. Параметры блоков типов TIMER, COUNTER и BLOCK_хх могут передаваться из одного входного параметра в другой только в том случае, если вызывающий блок является функциональным блоком. Эти утверждения представлены в таблице 19.5.

Таблица 19.5 Возможные комбинации при передаче параметров блоков

Вызывающий → вызванный Тип описания	FC вызывает FC			FB вызывает FC			FC вызывает FB			FB вызывает FB		
	Е	С	Р	Е	С	Р	Е	С	Р	Е	С	Р
Input → Input	x			x	x		x		x	x	x	x
Output → Output	x			x	x		x			x	x	
In/out → Input	x						x					
In/out → Output	x			x			x			x		
In/out → In/out	x			x						x		

Е – простые (Elementary) типы данных

С – сложные (Complex) типы данных

Р – параметрические (Parameter) типы TIMER, COUNTER и BLOCK_хх

19.5 Примеры

19.5.1 Пример «Ленточный транспортер конвейера»

Пример показывает передачу сигнальных состояний через параметры блоков. Для этой цели мы используем функцию системы управления ленточным транспортером конвейера, описание которой содержится в главе 5 «Функции для работы с памятью». Система управления транспортером конвейера будет запрограммирована в функциональном блоке, и все входы и выходы должны быть связаны с параметрами блока, что даст возможность вызывать функциональный блок повторно (для нескольких транспортеров конвейера).

На рисунке 19.3 показаны входные и выходные параметры функционального блока, а также используемые статические и временные локальные данные.

Распределение параметров в этом случае вполне простое. Все двоичные операнды, которые были входами, стали входными параметрами, все выходы стали выходными параметрами, а все меркеры стали статическими локальными данными. Вы также заметили, что имена были незначительно изменены, потому что для внутриблочных (локальных) переменных допустимы только буквы, цифры и знак подчеркивания.

Функциональный блок «Conveyor_Belt» будет управлять двумя транспортерами конвейера. Для этого он будет вызываться дважды; первый раз с входами и выходами транспортера конвейера 1 (conveyor belt 1), второй раз с входами и выходами транспортера конвейера 2 (conveyor belt 2).

При каждом вызове функциональному блоку требуется экземплярный блок данных, где он хранит данные транспортера конвейера. Блок данных транспортера конвейера 1 будет называться «BeltData1», а транспортера конвейера 2 – «BeltData2».

Выполненный пример программирования вы можете найти в программе «Conveyor Example» («Пример «Конвейер»») в библиотеке «LAD_Book» или «FBD_Book» на прилагаемой к книге дискете. В нем продемонстрировано программирование функционального блока FB 21 с входными параметрами, выходными параметрами и статическими локальными данными. В качестве экземплярных блоков можно использовать любые блоки данных; в примере DB 21 используется для «BeltData1», и DB 22 – для «BeltData2». В таблице символов эти блоки данных имеют тип данных функционального блока (в примере FB 21, если «Conveyor_Belt» является символом для FB 21).

При вызове функционального блока вы можете использовать входы и выходы из таблицы символов в качестве фактических параметров. В тех случаях, где эти глобальные символы содержат специальные знаки, вы должны в программе заключить эти символы в кавычки. Таблица символов разработана в этой главе для всех трех примеров (таблица 19.7 в конце главы).

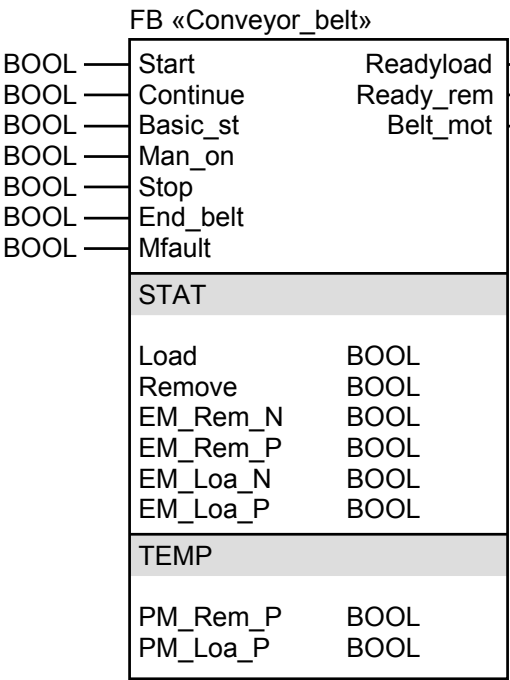


Рисунок 19.3
Функциональный блок для примера
«Ленточный транспортер конвейера»

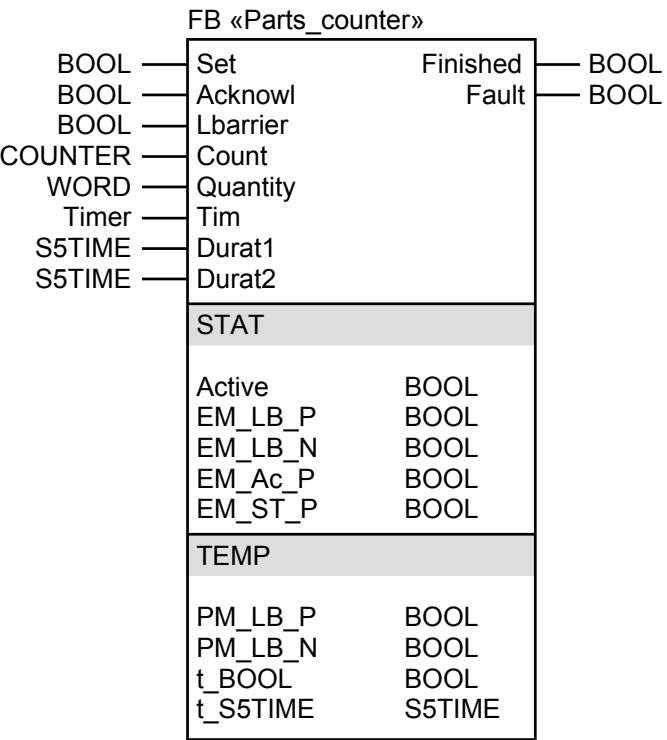


Рисунок 19.4
Функциональный блок для примера
«Счетчик деталей»

19.5.2 Пример «Счетчик деталей»

Пример демонстрирует обработку параметров боков простого типа данных. Основой функции является пример «Счетчик деталей» из главы 8 «Счетчики». Та же функция реализуется здесь как функциональный блок со всеми глобальными переменными, описанных либо как параметры блока, либо как статические локальные данные. Управление таймерами и счетчиками осуществляется здесь через отдельные элементы.

На рисунке 19.4 показаны входные и выходные параметры создаваемого функционального блока, а также используемые статические и временные локальные данные.

Таймеры и счетчики передаются через параметры блока типа TIMER и COUNTER. Эти параметры блока должны быть входными параметрами. Начальные значения счетчика (Quantity) и таймера (Durat1 и Durat2) также могут быть переданы как параметры блока; типы данных параметров блока здесь соответствуют типам фактических параметров.

Меркеры фронта хранятся в статических локальных данных, а меркеры импульса хранятся во временных локальных данных.

Пример программы вы можете найти в библиотеке «LAD_Book» или «FBD_Book» в разделе «Conveyor Example» («Пример «Конвейер»») на прилагаемой к книге дискете. Он содержит функциональный блок FB 22 «Parts_counter» и соответствующий эк-

земпляр блока данных «CountDat». Вы можете использовать входы, выходы, таймер и счетчик из таблицы символов (таблица предыдущего примера) в качестве фактических параметров при вызове функционального блока.

19.5.3 Пример «Устройство подачи»

Те же функции, описанные в двух предыдущих примерах, также могут быть вызваны как локальные экземпляры. В нашем примере это означает, что мы программируем функциональный блок, называемый «Feed» («Подача»), который будет управлять четырьмя транспортерами конвейера и считать проходящие детали. В этом функциональном блоке FB «Conveyor_Belt» вызывается четыре раза, а FB «Parts_counter» – один раз. Вызов не каждый раз происходит со своим собственным экземпляром блока данных, но вызываемые FB должны хранить их данные в экземплярном блоке данных функционального блока «Feed».

На рисунке 19.5 показано, как связаны отдельные элементы управления транспортера конвейера (FB «Parts_counter» здесь не представлен). Стартовый сигнал (Start) подается на вход *Start* контроллера транспортера 1 (belt 1), выход *Ready_rem* соединен с входом *Start* транспортера 2 (belt 2) и так далее. Наконец, выход *Ready_rem* транспортера 4 (belt 4) подключен к выходу *Remove* блока «Feed». Такая же последовательность сигналов проходит в обратном направлении от *Remove* через *Continue* и *Readyload* к *Load*.

Belt_mot, *Lbarr* и */Mfault* являются отдельными сигналами транспортера конвейера; *Reset*, *Start* и *Stop* управляют всеми транспортерами конвейера посредством *Basic_st*, *Man_on* и *Stop*.

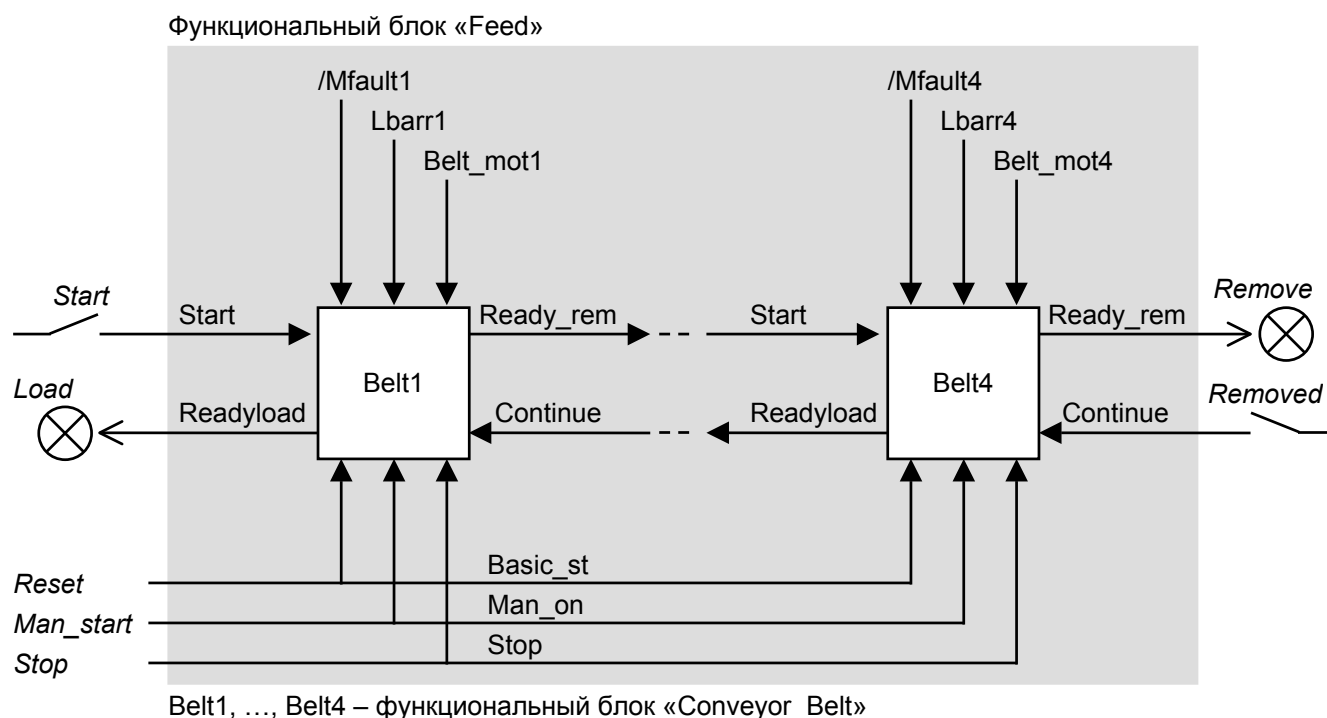


Рисунок 19.5 Пример программирования «Устройство подачи»

Следующая программа для функционального блока «Feed» разработана точно таким же способом. Входные и выходные параметры функционального блока могут быть взяты из рисунка 19.5. Кроме того, числовые значения счетчика деталей *Quantity*, *Durat1* и *Durat2* здесь спроектированы как входные параметры. Мы описываем данные отдельных элементов управления транспортером конвейера и данные счетчика деталей в статических локальных данных точно так же, как и для определенного пользователем типа данных, то есть с именем и типом данных. Переменная «Belt1» предназначена для приема структуры данных функционального блока «Conveyor_Belt», как и переменная «Belt2» и так далее; переменная «Check» принимает структуру данных функционального блока «Parts_counter».

Программа в функциональном блоке начинается с инициализации сигналов, общих для всех транспортеров конвейера. В нашем случае мы используем тот факт, что параметры функциональных блоков, вызываемых как локальные экземпляры, являются статическими локальными данными в текущем блоке и могут рассматриваться как таковые. Параметр блока *Man_on* в текущем функциональном блоке управляет входными параметрами *Man_on* всех четырех элементов управления транспортерами конвейера с использованием одиночного элемента назначения (присваивания). Таким же образом обрабатываются сигналы *Stop* и *Reset*. И теперь элементы управления транспортерами конвейера инициализированы общими сигналами. (Вы, конечно, можете также инициализировать эти входные параметры при вызове функционального блока.)

Последующие вызовы функциональных блоков для элемента управления транспортером конвейера содержат параметры блока только для отдельных сигналов для каждого транспортера конвейера и соединение с параметрами блока «Feed». Отдельные сигналы – это световые барьеры, команды для мотора транспортера и сбой мотора. (Мы здесь используем тот факт, что при вызове функционального блока могут быть инициализированы не все параметры блока.) Мы программируем соединения между отдельными контроллерами транспортеров, используя элементы назначения (присваивания).

FB «Parts_counter» вызывается как локальный экземпляр, даже если он не имеет близкого соединения с сигналами элементов управления транспортерами конвейера. Экземплярный блок данных блока «Feed» хранит данные FB.

Входные параметры *Quantity*, *Durat1* и *Durat2* блока «Feed» должны быть установлены только один раз. Это может быть сделано с использованием значений по умолчанию (как в примере) или в процедуре рестарта в OB 100 (например, через непосредственное назначение, если эти три параметра рассматриваются как глобальные данные).

Таблица 19.6 Раздел описаний FB «Feed»

Address (Адрес)	Declaration (Описание)	Name (Имя)	Type (Тип)	Initial value (Начальное значение)	Comment (Комментарий)
0.0	in	Start	BOOL	FALSE	Start conveyor belts / Запуск транспортеров конвейера
0.1	in	Removed	BOOL	FALSE	Parts have been removed from belt / Детали удалены с транспортера
0.2	in	Man_start	BOOL	FALSE	Start conveyor belts manually / Запуск транспортеров конвейера вручную
0.3	in	Stop	BOOL	FALSE	Stop conveyor belts / Останов транспортеров конвейера
0.4	in	Reset	BOOL	FALSE	Set control to the basic setting / Установить элемент управления в основное состояние (вернуть базовые установки)
2.0	in	Count	COUNTER		Counter for the parts / Счетчик деталей
4.0	in	Quantity	WORD	W#16#200	Number of parts / Количество деталей
6.0	in	Tim	TIMER		Timer for the monitor / Таймер монитора (устройства наблюдения)
8.0	in	Durat1	S5TIME	S5T#5s	Monitoring time for parts / Время наблюдения для деталей
10.0	in	Durat2	S5TIME	S5T#10s	Monitoring time for gap / Время наблюдения для промежутка
12.0	out	Load	BOOL	FALSE	Load new parts onto belt / Помещение новых деталей на транспортер
12.1	out	Remove	BOOL	FALSE	Remove parts from belt / Удаление деталей с транспортера
	in_out				
14.0	stat	Belt1	Conveyor_belt		Control for belt1 / Элемент управления транспортером 1
20.0	stat	Belt2	Conveyor_belt		Control for belt2 / Элемент управления транспортером 2
26.0	stat	Belt3	Conveyor_belt		Control for belt3 / Элемент управления транспортером 3
32.0	stat	Belt4	Conveyor_belt		Control for belt4 / Элемент управления транспортером 4
38.0	stat	Check	Parts_counter		Control for counting and monitoring / Управление подсчетом и наблюдением
	temp				

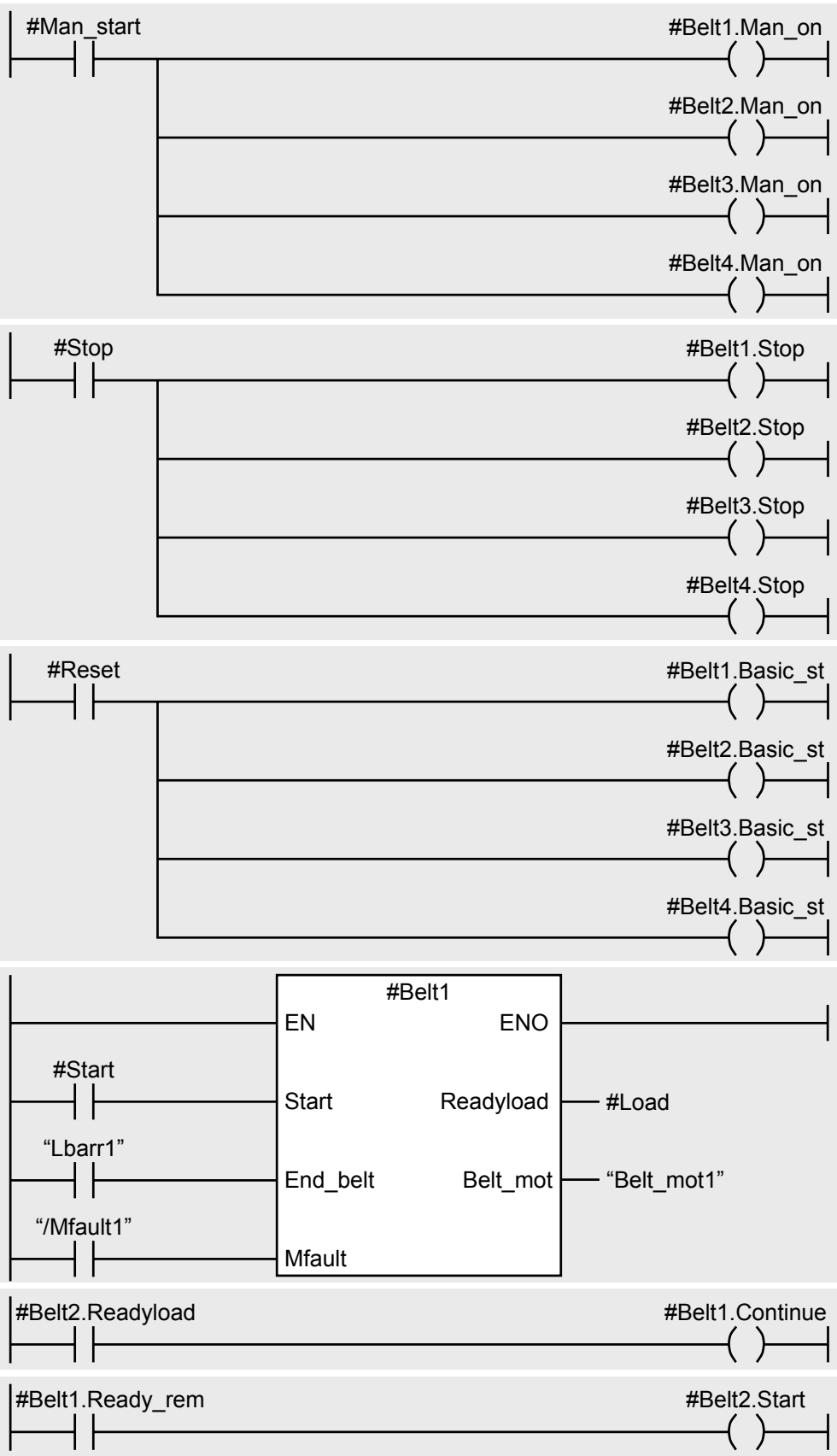
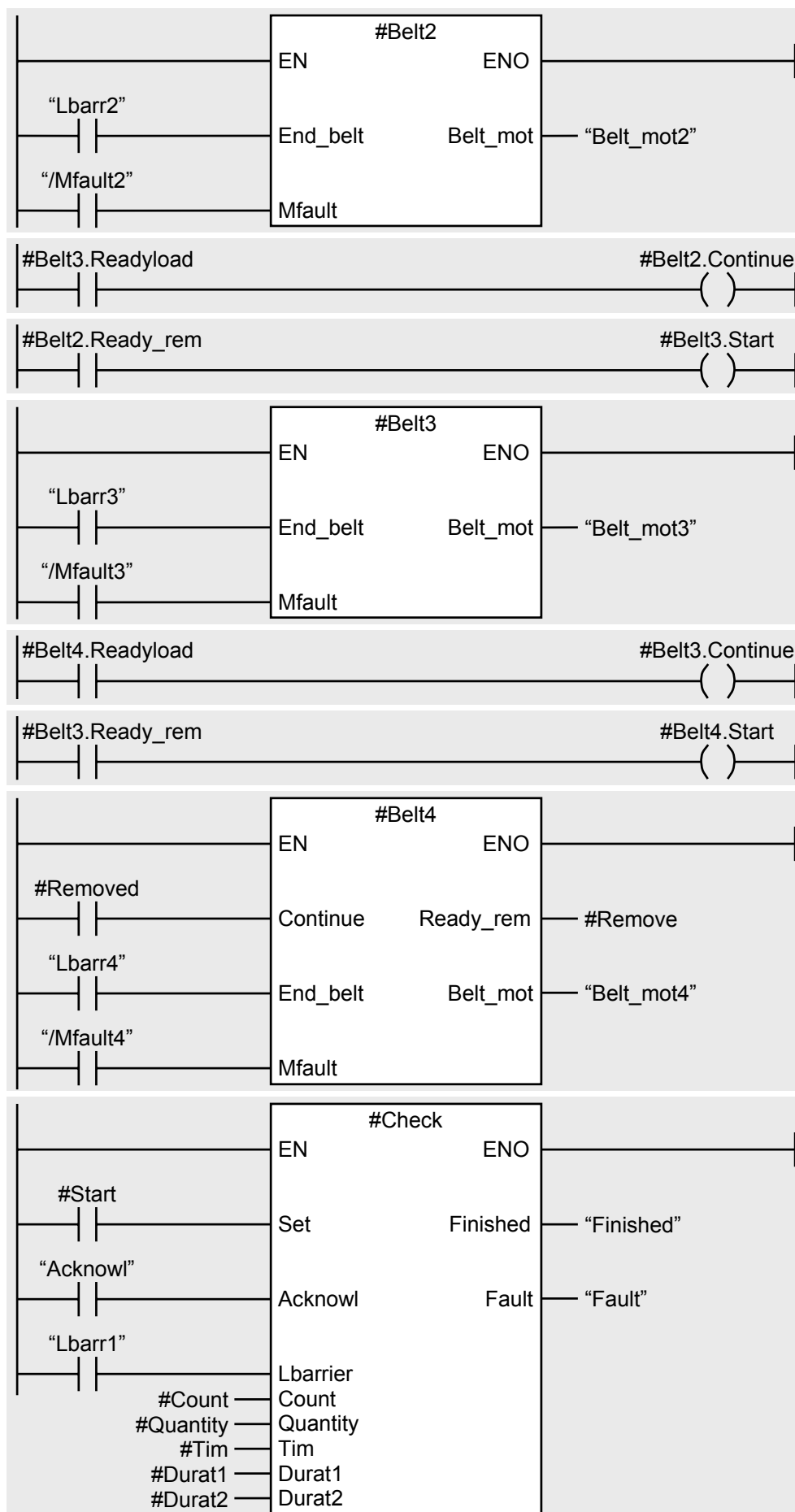


Рисунок 19.6
Программа для приме-
ра «Устройство пода-
чи» (LAD)

Продолжение на следующей странице



Продолжение

Рисунок 19.6
Программа для приме-
ра «Устройство пода-
чи» (LAD)

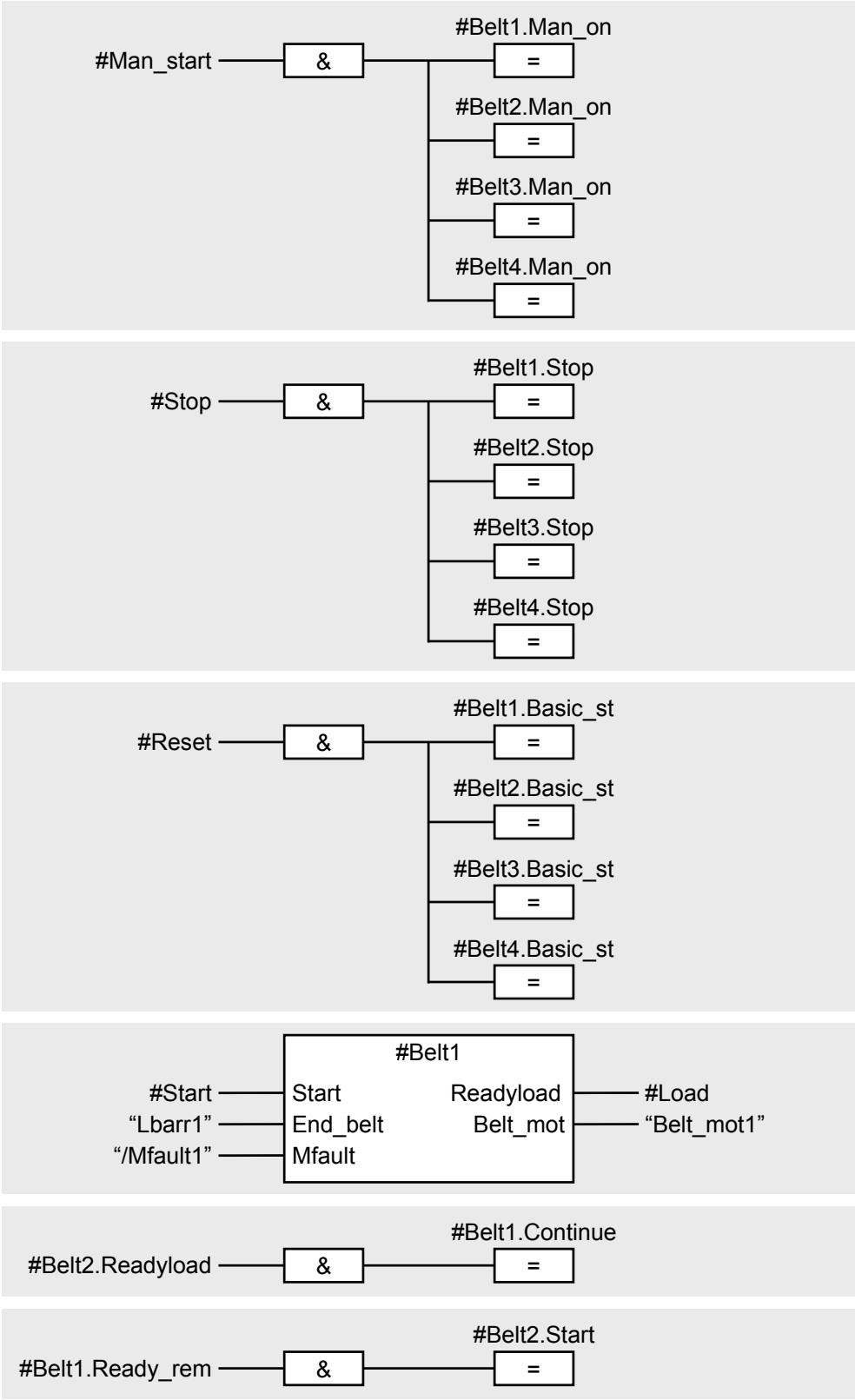
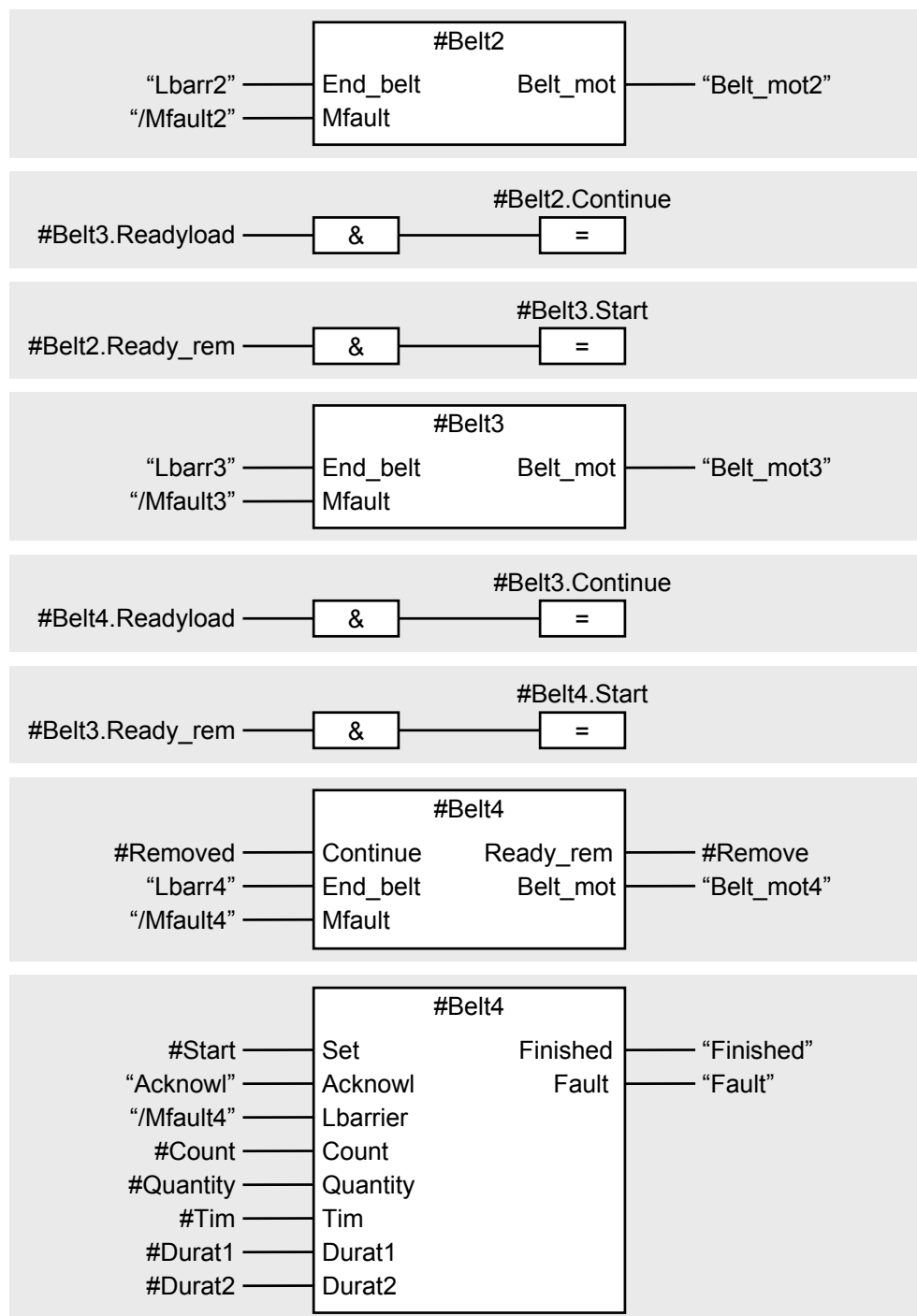


Рисунок 19.7 Программа для примера «Устройство подачи» (FBD)

Продолжение на следующей странице



Продолжение

Рисунок 19.7 Программа для примера «Устройство подачи» (FBD)

Таблица 19.7 Таблица символов для примеров «Ленточный транспортер конвейера», «Счетчик деталей» и «Устройство подачи»

Symbol (Символ)	Address (Адрес)	Data Type (Тип данных)	Comment (Комментарий)
Conveyor_belt	FB 21	FB 21	Conveyor belt control / Элемент управления транспортером конвейера
BeltData1	DB 21	FB 21	Data for conveyor belt1 / Данные для транспортера конвейера1
BeltData2	DB 22	FB 21	Data for conveyor belt2 / Данные для транспортера конвейера2
Parts_counter	FB 22	FB 22	Counter control and monitor / Элемент управления счетчиком и монитор (устройство наблюдения)
CounterDat	DB 29	FB 22	Data for parts counter / Данные для счетчика деталей
Feed	FB 20	FB 20	Feed with several belts / Устройство подачи с несколькими транспортерами
FeedDat	DB 20	FB 20	Data for Feed / Данные для устройства подачи
Cycle	OB 1	OB 1	Main program, cyclic execution / Главная программа, циклическое выполнение
Basic_st	I 0.0	BOOL	Set controller to the basic state / Установка контроллера в основное состояние
Man_on	I 0.1	BOOL	Switch on conveyor belt motors / Включение моторов транспортеров конвейера
/Stop	I 0.2	BOOL	Stop conveyor belt motors (zero active) / Останов моторов транспортеров конвейера (нулевая активность)
Start	I 0.3	BOOL	Start conveyor belt / Запуск транспортера конвейера
Continue	I 0.4	BOOL	Acknowledgement that parts have been removed / Уведомление о том, что детали удалены
Acknowl	I 0.6	BOOL	Acknowledge fault / Подтверждение ошибки
Set	I 0.7	BOOL	Set counter, activate monitor / Установка счетчика, активизация монитора
Lbarr1	I 1.0	BOOL	(Light barrier) «End of belt» sensor signal for conveyor belt 1 / (Световой барьер) сигнал датчика «Конец транспортера» для транспортера конвейера 1
Lbarr2	I 1.1	BOOL	(Light barrier) «End of belt» sensor signal for conveyor belt 2 / (Световой барьер) сигнал датчика «Конец транспортера» для транспортера конвейера 2
Lbarr3	I 1.2	BOOL	(Light barrier) «End of belt» sensor signal for conveyor belt 3 / (Световой барьер) сигнал датчика «Конец транспортера» для транспортера конвейера 3
Lbarr4	I 1.3	BOOL	(Light barrier) «End of belt» sensor signal for conveyor belt 4 / (Световой барьер) сигнал датчика «Конец транспортера» для транспортера конвейера 4
/Mfault1	I 2.0	BOOL	Motor protection switch conveyor belt 1 (zero active) / Включение защиты мотора транспортера конвейера 1 (нулевая активность)
/Mfault2	I 2.1	BOOL	Motor protection switch conveyor belt 2 (zero active) / Включение защиты мотора транспортера конвейера 2 (нулевая активность)
/Mfault3	I 2.2	BOOL	Motor protection switch conveyor belt 3 (zero active) / Включение защиты мотора транспортера конвейера 3 (нулевая активность)

Таблица 19.7 Продолжение

Symbol (Символ)	Address (Адрес)	Data Type (Тип данных)	Comment (Комментарий)
/Mfault4	I 2.3	BOOL	Motor protection switch conveyor belt 4 (zero active) / Включение защиты мотора транспортера конвейера 4 (нулевая активность)
Readyload	Q 4.0	BOOL	Load new parts onto belt / Загрузка новых деталей на транспортер
Ready_rem	Q 4.1	BOOL	Remove parts from belt / Удаление деталей с транспортера
Finished	Q 4.2	BOOL	Number of parts reached / Достигнутое число деталей
Fault	Q 4.3	BOOL	Monitor activated / Устройство наблюдения активировано
Belt_mot1	Q 5.0	BOOL	Switch on belt motor for conveyor belt 1 / Включение мотора транспортера конвейера 1
Belt_mot2	Q 5.1	BOOL	Switch on belt motor for conveyor belt 2 / Включение мотора транспортера конвейера 2
Belt_mot3	Q 5.2	BOOL	Switch on belt motor for conveyor belt 3 / Включение мотора транспортера конвейера 3
Belt_mot4	Q 5.3	BOOL	Switch on belt motor for conveyor belt 4 / Включение мотора транспортера конвейера 4
Quantity	MW 4	WORD	Number of parts / Количество деталей
Durat1	MW 6	S5TIME	Monitoring time for light barrier covered / Время наблюдения для закрытого светового барьера
Durat2	MW 8	S5TIME	Monitoring time for light barrier not covered / Время наблюдения для открытого светового барьера
Count	C 1	COUNTER	Counter for parts / Счетчик деталей
Monitor	T 1	TIMER	Timer for monitor / Таймер устройства наблюдения