

Управление программным потоком

LAD и FBD предоставляют вам инструментарий для управления программным потоком. Вы можете прервать линейное исполнение программы внутри блока или структурировать программу с использованием программируемых вызовов блоков. Можно воздействовать на выполнение программы в зависимости от значений, полученных во время исполнения, параметров процесса или в соответствии с состоянием управляемой системы (предприятия).

Биты состояния предоставляют информацию о результате арифметической или математической функции и об ошибках (таких, как выход за границы допустимого диапазона во время вычисления). Вы можете использовать совместно (объединить) сигнальные состояния битов состояния непосредственно в вашей программе с помощью контактов.

Функции перехода вы можете использовать для разветвления как безусловного, так и в зависимости от RLO (результата логической операции).

Еще один метод воздействия на исполнение программы предоставляет **главное реле управления** (Master Control Relay, MCR). Изначально разработанный для управления релейными контакторами этот метод предлагается языками LAD и FBD в программной версии.

Для структурирования программы вы можете использовать **функции блоков**. Функции и функциональные блоки можно использовать снова и снова, определяя **параметры блоков**.

Глава 19 «Параметры блоков» содержит примеры, показанные в главе 5 «Функции для работы с памятью» и главе 8 «Счетчики». В этот раз они запрограммированы как функциональные блоки с параметрами блоков. Эти функциональные блоки позже вызываются также в примере «Feed» («Устройство подачи») как локальные экземпляры.

15 Биты состояния

Биты состояния RLO, BR, CC0, CC1 и переполнение (overflow); установка и оценка битов состояния; использование бинарного результата; EN/ENO

16 Функции перехода

Безусловный переход; переход в зависимости от RLO

17 Главное реле управления

MCR-зависимость; диапазон MCR; зона MCR

18 Функции блоков

Типы блоков, вызов блока, конец блока; статические локальные данные (локальные данные состояния); регистр блока данных, использование операндов данных; управление блоками данных

19 Параметры блоков

Объявление параметров; формальные параметры, фактические параметры; передача параметров в вызываемые блоки; примеры: транспортер конвейера, счетчик деталей и подача

Содержание главы 15

<u>15</u>	<u>Биты состояния</u>	4
<u>15.1</u>	<u>Описание битов состояния</u>	4
<u>15.2</u>	<u>Установка битов состояния</u>	7
<u>15.3</u>	<u>Считывание битов состояния</u>	11
<u>15.4</u>	<u>Использование бинарного результата</u>	13
<u>15.4.1</u>	<u>Установка бинарного результата BR</u>	13
<u>15.4.2</u>	<u>Главная цепь, механизм EN/ENO</u>	14
<u>15.4.3</u>	<u>ENO в случае блоков, написанных пользователем</u>	15

15 Биты состояния

Биты состояния представляют собой двоичные «флаги» (кодовые биты состояния). CPU использует их для управления операциями бинарной логики и устанавливает их во время цифровой обработки. Вы можете считать эти биты состояния или воздействовать на определенные биты. Биты состояния объединены в слово, слово состояния. Однако, вы не можете получить доступ к этому слову состояния посредством LAD или FBD.

15.1 Описание битов состояния

Таблица 15.1 содержит доступные биты состояния. CPU использует бинарные флаги для управления бинарными функциями; цифровые флаги индицируют в основном результаты арифметических и математических функций.

Таблица 15.1 Биты состояния

Бинарные флаги	
/FC	Первичный опрос / First Check
RLO	Результат логической операции / Result of logic operation
STA	Состояние / Status
OR	Бит состояния OR / Status bit OR
BR	Бинарный результат / Binary result
Цифровые флаги	
OS	Переполнение с запоминанием / Stored overflow
OV	Переполнение / Overflow
CC0	Кодовый бит условия (состояния) 0 / Condition code (status) bit 0
CC1	Кодовый бит условия (состояния) 1 / Condition code (status) bit 1

Первичный опрос (First check)

Бит состояния /FC управляет бинарной логикой внутри логической системы управления. Шаг битовой логики всегда начинается с /FC = «0» и инструкции бинарной проверки, первичного опроса. Первичный опрос устанавливает /FC = «1».

Первичный опрос соответствует в LAD первому контакту в сети и в FBD входу первой двоичной функции.

Шаг битовой логики завершается присваиванием бинарного значения (например, единичной катушки или назначения), условным переходом или сменой блока. Эти элементы устанавливают /FC = «0».

Результат логической операции (RLO)

Бит состояния RLO – это промежуточный буфер в операциях бинарной логики. В ходе начальной проверки CPU пересылает результат считывания (проверки) в RLO, комбинирует результат считывания с хранимым RLO в каждое последующее считывание и сохраняет результат, в свою очередь, в RLO.

Вы можете сохранить RLO с помощью катушки / блочного элемента SAVE в бинарном результате BR. При помощи RLO управляются функции для работы с памятью, таймеры и счетчики, а также выполняются некоторые функции перехода. RLO соответствует в LAD наличию тока в цепи ($RLO = \langle 1 \rangle$ – то же самое, что и «протекание тока»).

Состояние

Бит состояния STA соответствует сигнальному состоянию считанного бинарного операнда. В случае функций для работы с памятью значение STA то же, что и записанное значение, или (если операции записи не было произведено, например, если $RLO = \langle 0 \rangle$ или MCR активировано) STA соответствует значению адресованного (и не модифицированного) бинарного операнда.

В случае оценок фронта FP или FN значение RLO перед оценкой фронта записывается в STA. Все остальные бинарные функции устанавливают $STA = \langle 1 \rangle$.

Бит состояния STA не влияет на обработку функций LAD или FBD.

Бит состояния OR

Бит состояния OR хранит результат выполненной последовательной схемы или выполненного условия AND и показывает следующим обрабатываемым параллельным схемам или функции OR, что результат уже определен. Все другие бинарные функции сбрасывают бит состояния OR.

Переполнение

Бит состояния OV индицирует выход из допустимого диапазона или использование недействительных чисел REAL. На бит состояния OV влияют следующие функции: арифметические, математические функции, некоторые функции преобразования, функции сравнения типа REAL.

Бит состояния OV можно считать непосредственно.

Переполнение с запоминанием

Бит состояния OS хранит установленный бит состояния OV. Всякий раз, когда CPU устанавливает бит состояния OV, он также устанавливает бит состояния OS. Однако, если следующая корректно выполненная операция сбрасывает OV, OS остается установленным. Это дает возможность обнаружить выход за границы диапазона или операцию с недействительным числом REAL, даже позже по ходу программы.

Бит состояния OS доступен для прямого чтения. Смена блока сбрасывает бит состояния OS.

Биты состояния CC0 и CC1 (кодовые биты условия)

Биты состояния CC0 и CC1 предоставляют информацию о результате функции сравнения, арифметической или математической функции, побитовые логические операции или о бите, выдвинутом функцией сдвига.

Вы можете непосредственно считывать комбинации CC0 и CC1 (см. ниже).

Бинарный результат

LAD и FBD используют бит состояния BR для выполнения механизма EN/ENO для блочных элементов. Вы также можете самостоятельно устанавливать, сбрасывать или считывать бит состояния BR.

15.2 Установка битов состояния

Цифровые функции воздействуют на биты состояния CC0, CC1, OV и OS, как показано в таблице 15.2. Вы можете считать эти биты состояния сразу после блочного элемента функции.

Таблица 15.1 Установка битов состояния

Вычисление с типом INT				
Результат:	CC0	CC1	OV	OS
< -32 768 (ADD_I, SUB_I)	0	1	1	1
< -32 768 (MUL_I)	1	0	1	1
от -32 768 до -1	1	0	0	-
0	0	0	0	-
от +1 до +32 767	0	1	0	-
> +32 767 (ADD_I, SUB_I)	1	0	1	1
> +32 767 (MUL_I)	0	1	1	1
32 768 (DIV_I)	0	1	1	1
(-) 65 536	0	0	1	1
Деление на ноль	1	1	1	1

Вычисление с типом DINT				
Результат:	CC0	CC1	OV	OS
< -2 147 483 648 (ADD_DI, SUB_DI)	0	1	1	1
< -2 147 483 648 (MUL_DI)	1	0	1	1
от -2 147 483 648 до -1	1	0	0	-
0	0	0	0	-
от +1 до +2 147 483 647	0	1	0	-
> +2 147 483 647 (ADD_DI, SUB_DI)	1	0	1	1
> +2 147 483 647 (MUL_DI)	0	1	1	1
2 147 483 648 (DIV_DI)	0	1	1	1
(-) 4 294 967 296	0	0	1	1
Деление на ноль (DIV_DI, MOD_DI)	1	1	1	1

Сравнение				
Результат:	CC0	CC1	OV	OS
Равно	0	0	0	-
Больше	0	1	0	-
Меньше	1	0	0	-
Недействительное число REAL	1	1	1	1

Таблица 15.1 (Продолжение)

Вычисление с типом REAL				
Результат:	CC0	CC1	OV	OS
+ нормализованное	0	1	0	-
± ненормализованное	0	0	1	1
± ноль	0	0	0	-
– нормализованное	1	0	0	-
+ бесконечность (деление на ноль)	0	1	1	1
– бесконечность (деление на ноль)	1	0	1	1
± действительное число REAL	1	1	1	1

Преобразование NEG_I				
Результат:	CC0	CC1	OV	OS
от +1 до +32 767	0	1	0	-
0	0	0	0	-
от –1 до –32 767	1	0	0	-
(–) 32 768	1	0	1	1

Преобразование NEG_D				
Результат:	CC0	CC1	OV	OS
от +1 до +2 147 483 647	0	1	0	-
0	0	0	0	-
от –1 до –2 147 483 647	1	0	0	-
(–) 2 147 483 648	1	0	1	1

Функция сдвига				
Выдвинутый бит:	CC0	CC1	OV	OS
«0»	0	0	0	-
«1»	0	1	0	-
с числом сдвига 0	-	-	-	-

Побитовые логические операции				
Результат:	CC0	CC1	OV	OS
ноль	0	0	0	-
не ноль	0	1	0	-

Биты состояния в вычислениях с типами INT и DINT

Арифметические функции с форматами данных INT и DINT устанавливают все цифровые биты состояния. Нулевой результат устанавливает CC0 и CC1 в «0». Биты CC0 = «0» и CC1 = «1» указывают на положительный результат, CC0 = «1» и CC1 = «0» сообщают об отрицательном результате. Выход из диапазона допустимых значений устанавливает OV и OS (обратите внимание на другое значение CC0 и CC1 в случае переполнения). Все цифровые биты состояния, установленные в «1», свидетельствуют о делении на ноль.

Биты состояния в вычислениях с типом REAL

Арифметические функции с форматом данных REAL и математические функции устанавливают все цифровые биты состояния. Нулевой результат устанавливает CC0 и CC1 в «0». CC0 = «0» и CC1 = «1» указывают на положительный результат, CC0 = «1» и CC1 = «0» сообщают об отрицательном результате. Выход за границы допустимого диапазона устанавливает OV и OS (обратите внимание на другое значение CC0 и CC1 в случае переполнения). Недействительное число REAL указывается в виде установленных в «1» всех цифровых битов.

Число типа REAL называется «ненормализованным», если оно представляется с уменьшенной точностью. В таком случае экспонента равна нулю; абсолютное значение ненормализованного числа REAL меньше $1.175\,494 \times 10^{-38}$. CPU S7-300 рассматривают ненормализованные числа REAL, как нуль (см. также параграф 3.5.4 «Простые типы данных»).

Биты состояния для функций преобразования

Из функций преобразования на все цифровые биты состояния влияет дополнение до двух (дополнительный код). Кроме того, следующие функции преобразования устанавливают биты состояния OV и OS в случае ошибки (выход за границы допустимого диапазона или недействительное число REAL):

- I_BCD и DI_BCD:
Преобразование INT и DINT в BCD;
- CEIL, FLOOR, ROUND, TRUNC:
Преобразование REAL в DINT.

Биты состояния для функций сравнения

Функции сравнения устанавливают биты состояния CC0 и CC1. Флаги устанавливаются независимо от выполняемой функции сравнения.

Биты состояния для функций сдвига

В случае функций сдвига сигнальное состояние последнего выдвигаемого бита пересылается в бит состояния CC1. CC0 и OV сбрасываются.

Биты состояния для побитовых логических операций

Если результат побитовой логической операции нулевой (все биты обнулены), то CC1 сбрасывается; если хотя бы один бит результата равен «1», то CC1 устанавливается. CC0 и OV сбрасываются.

15.3 Считывание битов состояния

LAD: Для считывания цифровых битов состояния и бинарного результата вы можете использовать нормально разомкнутый (NO) и нормально замкнутый (NC) контакт. Рисунок 15.1 показывает чтение с помощью нормально разомкнутого контакта. Считывание с применением нормально замкнутого контакта возвращает инвертированный результат считывания. Вы можете оперировать NO- и NC-контактами для оценки битов состояния точно так же, как и «нормальными» контактами. Библиотека «LAD_Book» на прилагаемой к книге дискете содержит примеры оценки битов состояния (FB 115 в программе «Program Flow Control», «Управление программным потоком»).

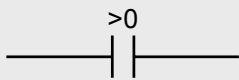
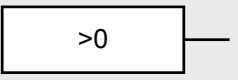
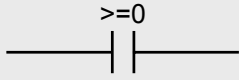
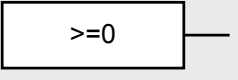
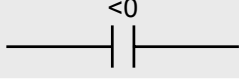
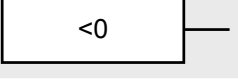
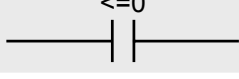
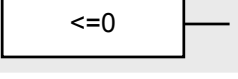
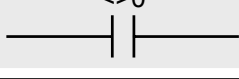
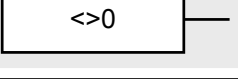
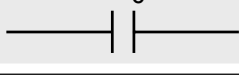
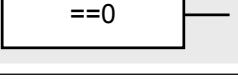
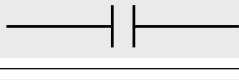
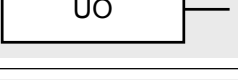
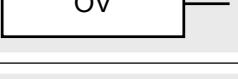
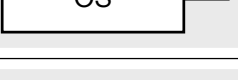
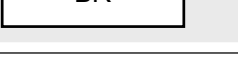
Представление LAD	Представление FBD	Ток течет или считывание успешно, когда
		Результат больше нуля [(CC0 = 0) & (CC1 = 1)]
		Результат больше или равен нулю [(CC0 = 0)]
		Результат меньше нуля [(CC0 = 1) & (CC1 = 0)]
		Результат меньше или равен нулю [(CC1 = 0)]
		Результат не равен нулю [(CC0 = 0) & (CC1 = 1) v (CC0 = 1) & (CC1 = 0)]
		Результат равен нулю [(CC0 = 0) & (CC1 = 0)]
		Результат недействителен (неупорядоченный) [(CC0 = 1) & (CC1 = 1)]
		Переполнение диапазона чисел [OV = 1]
		Сохраненное переполнение [OS = 1]
		Бинарный результат [BR = 1]

Рисунок 15.1 Чтение битов состояния

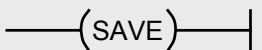
FBD: Возможно прямое или инвертированное считывание цифровых битов состояния и бинарного результата. Это подтверждается прямым чтением с ожиданием сигнального состояния «1». Считывание с ожиданием сигнального состояния «0» возвращает инвертированный результат проверки. Операции чтения для оценки битов состояния могут проводиться так же, как и «нормальные» проверки бинарных операндов. Библиотека «FBD_Book» на поставляемой с книгой дискете содержит примеры оценки битов состояния (FB 115 в программе «Program Flow Control», «Управление программным потоком»).

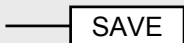
При пошаговом программировании вы можете найти эти элементы проверок в каталоге программных элементов (Program Element Catalog), вызываемого командой меню View → Catalog (Вид → Каталог) [Ctrl + K], или использовать пункт меню Insert → Program Elements (Вставка → Программные элементы), раздел «Status Bits» («Биты состояния»).

15.4 Использование бинарного результата

15.4.1 Установка бинарного результата BR

Присваивание бинарного результата

Представление LAD: 

Представление FBD: 

Катушка SAVE в цепи (LAD)

Вы можете сохранить RLO в бинарном результате с помощью катушки SAVE (Сохранить). Если в катушке SAVE течет электрический ток, то BR устанавливается, иначе он сбрасывается. Программируется катушка SAVE таким же образом, как и «единичная» катушка без бинарных операндов.

Обратите внимание на то, что катушка SAVE не завершает логическую операцию (бит состояния /FC не устанавливается в «0»). Это означает, что логическая операция, предшествующая катушке SAVE, также является предшествующей логической операцией для следующей сети.

Катушка SAVE не может быть запрограммирована совместно с Т-ветвью.

Блочный элемент SAVE в логической цепи (FBD)

При помощи блочного элемента SAVE вы можете сохранить RLO в бинарном результате. Если перед блочным элементом SAVE результат RLO равен «1», то BR устанавливается; в противном случае BR обнуляется. Блочный элемент SAVE программируется так же, как и блочный элемент присваивания без бинарного операнда.

Обратите внимание на то, что блочный элемент SAVE не завершает логическую операцию (бит состояния /FC не устанавливается в «0»). Следующая сеть, таким образом, начинается с «открытой» логической операции.

Блочный элемент SAVE после Т-ветви не допускается.

Управление бинарным результатом

LAD и FBD воздействуют на бинарный результат, чтобы управлять выходом ENO (рисунок 15.2). Если разрешающий выход ENO назначен, то его сигнальное состояние то же, что и у BR. В определенных случаях («BR соответствует функции») выполняемая функция LAD или FBD устанавливает бинарный результат в следующем виде:

- $BR := \langle 1 \rangle$
для MOVE, функций сдвига и логических операций со словами;
- $BR := OV$
для арифметических и математических функций;
- $BR := OV$ или $\langle 1 \rangle$
для функций преобразования;
- $BR := BR$ вызванного блока в случае вызовов блоков.

ENO подключен ?					
ДА			НЕТ		
EN подключен ?			EN подключен ?		
ДА		НЕТ	ДА		НЕТ
EN == $\langle 1 \rangle$?		BR соответствует функции	EN == $\langle 1 \rangle$?		на BR влияние не оказывается
ДА	НЕТ		ДА	НЕТ	
BR соответствует функции	$BR := \langle 0 \rangle$		$BR := \langle 1 \rangle$	$BR := \langle 0 \rangle$	

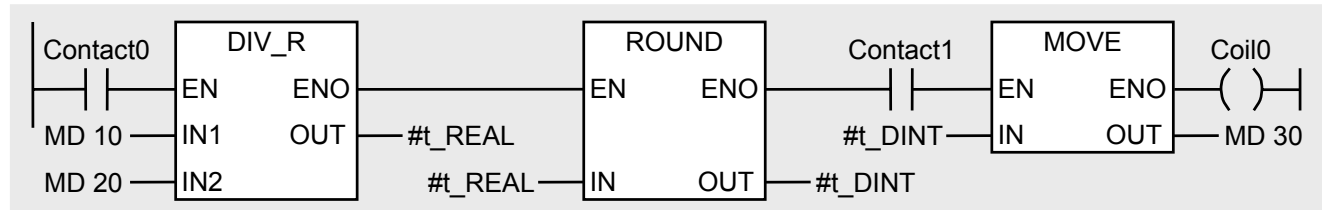
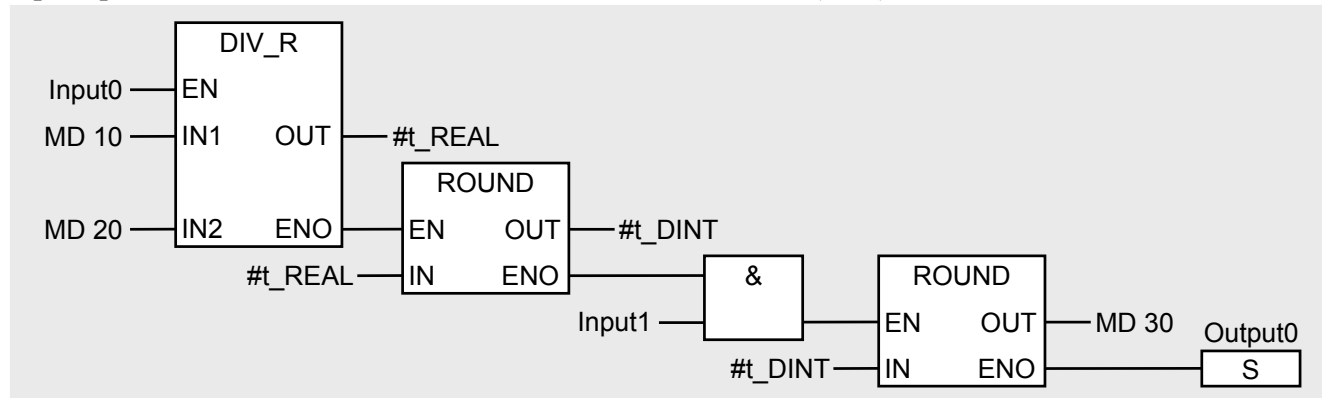
Рисунок 15.2 Общая схема установки бинарного результата

15.4.2 Главная цепь, механизм EN/ENO

В языках программирования LAD и FBD многие блочные элементы имеют разрешающий вход EN и разрешающий выход ENO. Если на разрешающем входе присутствует $\langle 1 \rangle$, то функция блочного элемента обрабатывается. Когда блочный элемент обработан корректно, разрешающий выход также имеет сигнальное состояние $\langle 1 \rangle$. Если во время обработки блочного элемента возникает ошибка (к примеру, переполнение при выполнении арифметической функции), то ENO устанавливается в $\langle 0 \rangle$. Если EN имеет сигнальное состояние $\langle 0 \rangle$, то ENO также установлен в $\langle 0 \rangle$.

Эти характеристики EN и ENO могут быть использованы для соединения в цепи нескольких блочных элементов с подключением разрешающего выхода к разрешающему входу следующего элемента (рисунок 15.3). Это означает, например, что вся цепь может быть «выключена» (блочные элементы не обрабатываются, если вход I 1.0 в примере имеет нулевое сигнальное состояние) или оставшаяся часть цепи не будет обработана, если один блочный элемент выдаст ошибку.

Вход EN и выход ENO не являются параметрами блока, но представляют собой последовательности операторов, которые программный редактор сам генерирует до и после всех блочных элементов (также в случае функций и функциональных блоков). Программный редактор использует бинарный результат для сохранения сигнального состояния на EN, пока блок обрабатывается, или для проверки флага ошибки блочного элемента.

Пример главной цепи (LAD)**Пример последовательного соединения блочных элементов (FBD)****Рисунок 15.3** Пример последовательного соединения EN и ENO**15.4.3 ENO в случае блоков, написанных пользователем**

Редактор программ обеспечивает вызов ваших собственных блоков с использованием разрешающего входа EN и разрешающего выхода ENO. Вы можете использовать разрешающий вход EN для условного вызова блока. Разрешающий выход ENO может использоваться, например, для сообщения о групповой ошибке (сигнальное состояние «1», если блок обработан правильно; «0», если при обработке блока возникла ошибка). Все системные блоки также сигнализируют о групповых ошибках через BR.

Вы можете управлять выходом ENO с помощью бинарного результата BR. Выход ENO имеет то же сигнальное состояние, что и BR по выходу из блока.

Например, при запуске блока BR может быть установлен в «1». Если затем во время обработки блока возникает ошибка (к примеру, если результат выходит за границы фиксированного диапазона, и дальнейшая обработка не допускается), установите с помощью катушки / блочного элемента SAVE бинарный результат в «0» и перейдите в конец блока, где будет произведен выход из него (в случае ошибки условие должно выдавать сигнальное состояние «0»). Заметьте, что катушка / блочный элемент RET устанавливает BR в «1», если вы выходите из блока посредством этой катушки / блочного элемента.