

Команды управления программой

20

Обзор главы

В разделе	Вы найдете	на стр.
20.1	Вызов FC/SFC без параметров	20–2
20.2	Вызов FB, FC, SFB, SFC и мультиэкземпляров	20–4
20.3	Возврат	20–8
20.4	Команды Master Control Relay	20–9
20.5	Активизация/деактивизация Master Control Relay	20–10
20.6	Включение/выключение Master Control Relay	20–13

20.1 Вызов FC/SFC без параметров

Описание

С помощью команды *Вызвать FC/SFC без параметров* можно вызвать функцию (FC) или системную функцию (SFC), которые не имеют параметров. Вызов является условным или безусловным в зависимости от предшествующей логической операции (см. пример).

В кодовом разделе функции (FC) Вы не можете задавать никаких параметров типа BLOCK_FC в качестве адреса для условного вызова. Однако, Вы можете задать параметр типа BLOCK_FC в качестве адреса в функциональном блоке (FB).

Условный вызов выполняется только в том случае, если RLO равен 1. Если условный вызов не выполняется, то RLO после команды вызова равен 0. Если команда выполняется, то реализуются следующие функции:

- Сохраняется адрес, необходимый для возврата в вызывающий блок.
- Сохраняются регистры блоков данных (блока данных и экземпляра блока данных).
- Предыдущая область локальных данных заменяется текущей областью локальных данных.
- Бит МА (бит активизации MCR) записывается в стек блоков (BSTACK).
- Для вызываемой FC или SFC создается новая область локальных данных.

Затем исполнение программы продолжается в вызванном блоке.

За более подробной информацией о передаче параметров обращайтесь к *Руководству по программированию /234/*.

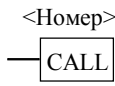
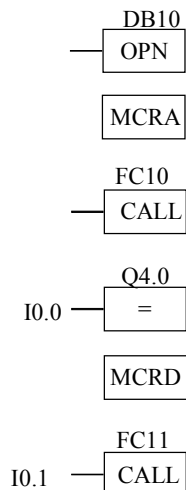
Блок FUP	Параметры	Тип данных	Область памяти	Описание
	Номер	BLOCK_FC	-	Номер FC или SFC (напр., FC10 или SFC59). Доступные SFC зависят от Вашего CPU. Условный вызов с параметром типа BLOCK_FC в качестве адреса возможен только в FB, но не в FC.

Рис. 20-1. Блок вызова FC/SFC без параметров



Если выполняется безусловный вызов для FC10, то команда CALL реализует следующие функции:

- Сохраняет адрес, необходимый для возврата в текущий FB.
- Сохраняет переключатели для DB10 и для экземпляра блока данных FB.
- Помещает в стек блоков (BSTACK) бит MA, установленный в 1 в команде MCRA, и сбрасывает этот бит в 0 для вызванного FC10

Исполнение программы продолжается в FC10. Если Вы хотите использовать функцию MCR в FC10, Вы должны ее там повторно активизировать. Когда FC10 завершается, исполнение программы возвращается в вызывающий FB. Бит MA восстанавливается. DB10 и экземпляр блока данных определенного пользователем FB снова являются текущими DB независимо от того, какие DB были использованы в FC10. После возврата из FC10 состояние сигнала входа I0.0 присваивается выходу Q4.0. Вызов FC11 является условным. Он выполняется только тогда, когда состояние входа I0.1 равно 1. Если вызов исполняется, то происходит то же самое, что и при вызове FC10.

Биты слова состояния

Безусловный вызов	BR	CC1	CC0	OV	OS	OR	STA	RLO	FC
записывает	-	-	-	-	0	0	1	-	0
Условный вызов	BR	CC1	CC0	OV	OS	OR	STA	RLO	FC
записывает	-	-	-	-	0	0	1	1	0

Рис. 20-2. Вызов FC/SFC без параметров

20.2 Вызов FB, FC, SFB, SFC и мультиэкземпляров

Описание

Вы можете вызывать функциональные блоки (FB), функции (FC), системные функциональные блоки (SFB), системные функции (SFC) и мультиэкземпляры, выбирая их из окна списка "Program Elements" ("Элементы программы"). Они находятся в конце списка семейства команд под следующими названиями:

- FB Blocks (блоки FB)
- FC Blocks (блоки FC)
- SFB Blocks (блоки SFB)
- SFC Blocks (блоки SFC)
- Multiple Instances (мультиэкземпляры)
- Libraries (библиотеки)

Когда Вы выбираете один из этих блоков, на экране появляется элемент с номером или символическим именем функции или функционального блока и принадлежащими ему параметрами.

Блок, который Вы вызываете, должен быть скомпилирован и должен уже существовать в Вашем программном файле, в библиотеке или в CPU.

Если команда вызова FB, FC, SFB, SFC и мультиэкземпляров выполняется, то она реализует следующие функции:

- Сохраняет адрес, необходимый для возврата в вызывающий блок..
- Сохраняет оба регистра блоков данных (блока данных и экземпляра блока данных).
- Заменяет предыдущую область локальных данных текущей областью локальных данных.
- Помещает в стек блоков (BSTACK) бит MA (бит активизации MCR).
- Создает новую область локальных данных для вызванной FC или SFC.

Замечание

Когда сохраняются регистры DB и DI, возможно, что они не указывают на блоки данных, которые Вы открыли. Из-за механизма копирования для передачи параметров, особенно, когда это касается функциональных блоков, компилятор иногда переписывает регистр DB. За подробностями обращайтесь к *Руководству по программированию /234/*.

Затем исполнение программы продолжается в вызванном блоке.

Выход ENO

Выход ENO блока FUP соответствует биту BR слова состояния (см. раздел 6.3). Когда Вы пишете функциональный блок или функцию, которые Вы хотите вызывать из FUP, независимо от того, пишете Вы эту FC или этот FB на AWL, KOP или FUP, держите в уме бит BR. В бите BR сохраняется RLO с помощью команды SAVE в соответствии со следующими критериями:

- Сохранить RLO равным 1 в бите BR, если FB или FC выполняется без ошибок.
- Сохранить RLO равным 0 в бите BR, если при исполнении FB или FC происходит ошибка.

Эти команды необходимо запрограммировать в конце FB или FC так, чтобы они были последними командами, исполняемыми в блоке.



Предупреждение

Непреднамеренный сброс бита BR в 0

При записи FB и FC в FUP, если Вы не обрабатываете бит BR так, как описано выше, один FB или FC может переписать бит BR другого FB или FC.

Во избежание этой проблемы сохраняйте RLO в конце каждого FB или FC, как описано выше.

Воздействие вызова на биты слова состояния

На рис. 20–3 показано влияние условного и безусловного вызова блока на биты слова состояния (см. раздел 6.3).

		BR	CC1	CC0	OV	OS	OR	STA	RLO	FC
Условный:	записывает	x	-	-	-	0	0	1	x	x
Безусловный:	записывает	-	-	-	-	0	0	x	x	x

Рис. 20-3. Воздействие вызова блока на биты слова состояния

Параметры

Параметры, определенные в разделе переменных (VAR) блока будут отражены в элементе FUP. Снабжение параметрами зависит от типа блока следующим образом:

- Для функции (FC) Вы должны поставить в соответствие всем формальным параметрам фактические параметры.
- Ввод фактических параметров для функциональных блоков (FB) не обязателен. Однако, Вы должны закрепить за FB блок данных (экземпляр DB). Если какому-то формальному параметру не поставлен в соответствие фактический параметр, то FB работает со значениями, существующими в экземпляре DB.
- В случае мультиэкземпляров Вам нет необходимости указывать экземпляр DB, так как вызываемый блок уже поставлен в соответствие номеру DB (за получением дополнительной информации об описании мультиэкземпляров обратитесь к разделу 3.5).

Для структурных параметров IN/OUT и параметров типов "Pointer" ("Указатель") и "Array" ("Массив") Вы должны сделать доступным фактический параметр (по крайней мере при первом вызове).

Каждый фактический параметр, который Вы делаете доступным при вызове функционального блока должен иметь тот же тип данных, что и соответствующий формальный параметр.

За информацией о том, как программировать функцию или как работать с параметрами, обратитесь к *Руководству по программированию /234/*.

В таблице показан элемент FUP для вызова FB, FC, SFB, SFC и описаны параметры, общие в элементе для всех этих блоков. Номер блока появляется автоматически в верхней части элемента (номер FB, FC, SFB или SFC, например, FC10).

Элемент FUP	Параметры	Тип данных	Область памяти	Описание
	DB no.	BLOCK_DB	-	Номер блока данных. Эта информация необходима только для вызова FB.
	EN	BOOL	I, Q, M, D, L, T, C	Деблокировка входа
	ENO	BOOL	I, Q, M, D, L	Деблокировка выхода

Рис. 20-4. Элемент FUP и параметры для вызова FB, FC, SFB, SFC

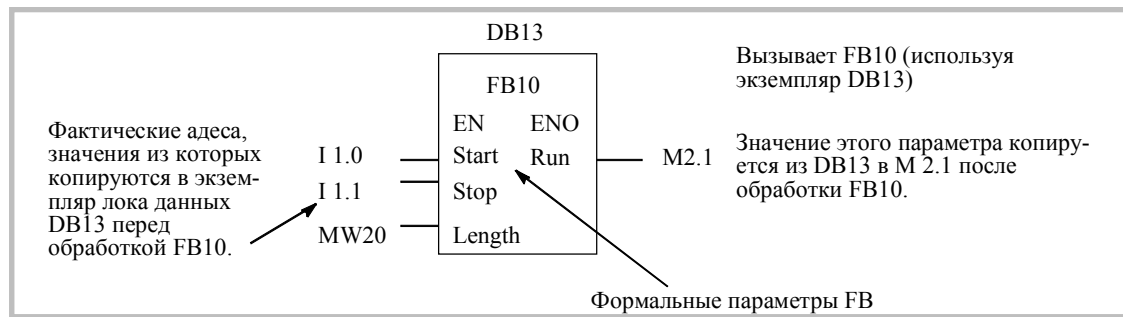


Рис. 20-5. Вызов FB как элемент FUP

20.3 Возврат

Описание

Вы можете использовать команду *Возврат* для выхода из блоков. Вы можете выйти из блока по условию.


Блок FUP	Параметры	Тип данных	Область памяти	Описание
	Отсутствуют	-	-	-

Рис. 20-6. Блок возврата

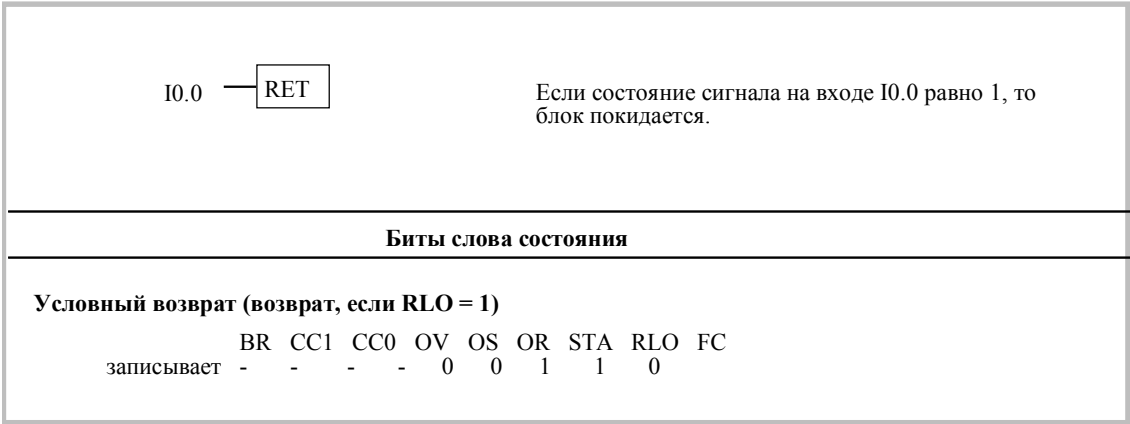


Рис. 20-7. Возврат

20.4 Команды Master Control Relay

Определение Master Control Relay

Master Control Relay (MCR, см. также раздел 20.5) - главное управляющее реле используется для активизации и деактивизации потока сигнала. Деактивированный поток сигнала соответствует последовательности команд, которая записывает нулевое значение вместо рассчитанного значения, или последовательности команд, которая оставляет неизменным существующее значение памяти. Операции, запускаемые командами, описанными в следующей таблице, зависят от MCR.

Команды *Присвоить* и *Коннектор* записывают в память 0, если MCR равно 0. Команды *Установить выход* и *Сбросить выход* оставляют существующее значение неизменным (см. таблицу).

Элемент FUP	Описание	Раздел данного руководства
	Коннектор (промежуточный выход)	8.9
	Присваивание	8.8
	Установка выхода	8.11
	Сброс выхода	8.12
	SR-триггер	8.25
	RS-триггер	8.26
	Передача значения	14.1

Рис. 20-8. Команды, на которые оказывает влияние зона действия MCR

Состояние сигнала MCR	Присваивание, коннектор 	Установка или сброс выхода 	Передача значения
0	Записывает 0 (имитирует реле, которое отпадает при отключении питания)	Ничего не записывает (имитирует реле с защелкой, которое остается в текущем состоянии при отключении питания)	Записывает 0 (имитирует элемент, который выдает значение 0 при отключении питания)
1	Нормальное исполнение	Нормальное исполнение	Нормальное исполнение

Рис. 20-9. Команды, зависящие от MCR, и их реакция на его состояние

20.5 Активизация/деактивизация Master Control Relay

Активизация MCR

С помощью команды *Активизировать Master Control Relay* Вы создаете последовательность команд, зависящих от MCR. После ввода этой команды Вы можете запрограммировать с помощью этих команд зоны MCR (см. раздел 20.6). Когда Ваша программа активизирует область MCR, все действия MCR зависят от содержимого стека MCR (см. рис. В-4).

Начало MCR

Элемент FUP	Параметры	Тип данных	Область памяти	Описание
MCRA	Отсутствуют	-	-	Активизирует MCR

Рис. 20-10. Элемент “Активизация Master Control Relay”

Деактивизация MCR

При подаче команды *Деактивизировать Master Control Relay* последовательность команд более не зависит от MCR. После этой команды Вы не можете больше программировать зоны MCR. Если Ваша программа деактивизирует область MCR, то MCR всегда пропускает поток сигнала независимо от записей в стеке MCR.

Элемент FUP	Параметры	Тип данных	Область памяти	Описание
MCRD	Отсутствуют	-	-	Деактивизирует MCR

Рис. 20-11. Элемент “Деактивизация Master Control Relay”

Стек MCR и бит, контролирующий его зависимость (бит МА), относятся к отдельным уровням и должны быть сохранены и извлечены всякий раз, когда Вы изменяете уровень последовательности. Они предварительно устанавливаются в начале каждого уровня последовательности (входные биты MCR с 1 по 8 устанавливаются 1, указатель стека MCR устанавливается в 0, и бит МА устанавливается в 0).

Стек MCR передается из блока в блок, а бит МА сохраняется и устанавливается в всякий раз, как блок вызывается. Он извлекается обратно в конце блока.

MCR может быть реализован таким образом, что он оптимизирует время выполнения CPU, генерирующего код. Причина этого состоит в том, что последовательность команд, зависящая от MCR, не выполняется блоком; она должна быть явно активизирована командой MCR. CPU, генерирующий код, распознает эту команду и генерирует дополнительный код, необходимый для оценки стека MCR, пока он не распознает команду MCR или не достигнет конца блока. Для команд вне диапазона MCRA/ MCRD время исполнения не увеличивается.

Команды MCRA и MCRD всегда должны использоваться парами внутри Вашей программы.

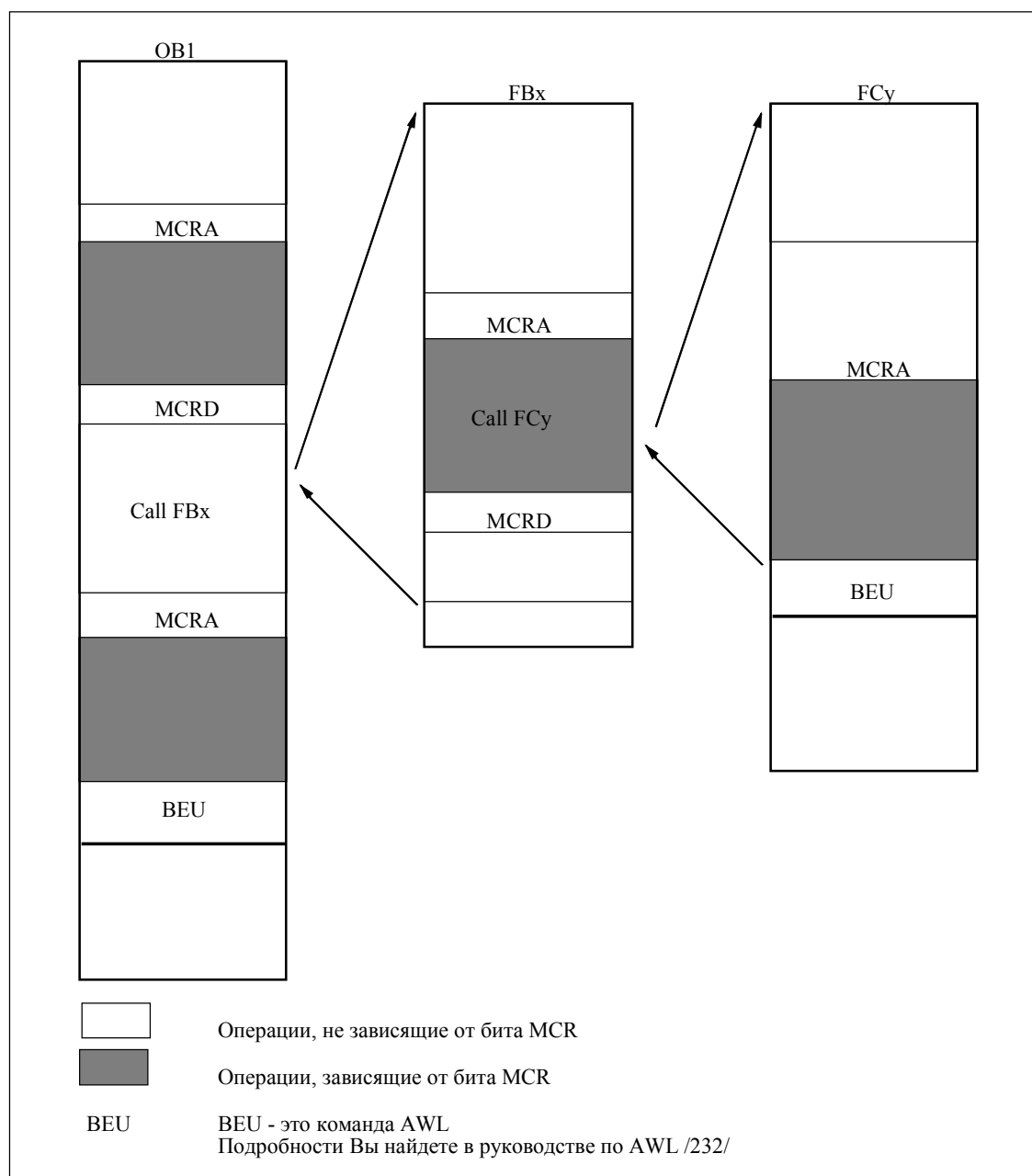


Рис. 20-12. Активизация и деактивизация области MCR

Операции, запрограммированные между MCRA и MCRD, зависят от состояния бита MCR. Операции, запрограммированные вне последовательности MCRA–MCRD, не зависят от состояния бита MCR. Если команда MCRD отсутствует, то команды, запрограммированные между командами MCRA и BEU, зависят от бита MCR.

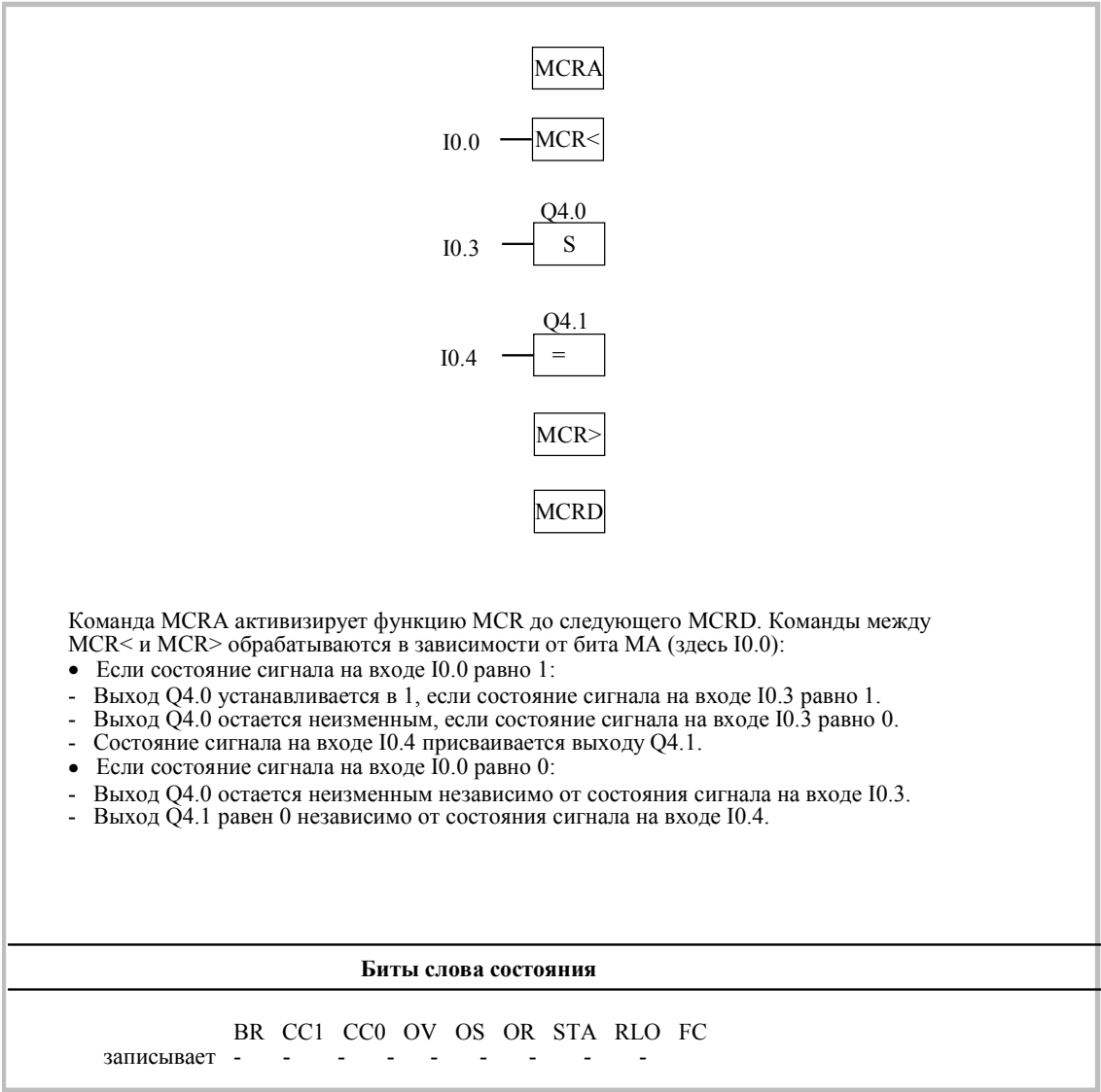


Рис. 20-13. Master Control Relay (активизация и деактивизация)

Зависимость функций (FC) и функциональных блоков (FB) Вы должны запрограммировать в блоке сами. Если эта функция или функциональный блок вызывается из последовательности MCRA/MCRD, то не все команды внутри этой последовательности автоматически зависят от бита MCR. Чтобы достичь этого, используйте команду MCRA вызванного блока.

Предупреждение

Риск травмирования людей и повреждения оборудования.
Никогда не используйте команду MCR для АВАРИЙНОГО ВЫКЛЮЧЕНИЯ или в качестве предохранительного приспособления для персонала. MCR не является заменой для аппаратного главного управляющего реле.



20.6 Включение/выключение Master Control Relay

Включение MCR

Команда *Включить Master Control Relay* (MCR<) запускает операцию, которая сохраняет RLO в стеке MCR и открывает зону действия MCR. На команды, показанные в следующей таблице, оказывает влияние RLO, сохраненное в стеке MCR, когда зона действия MCR открыта. Стек MCR работает по принципу буфера LIFO (Last In, First Out - последний вошел, первый вышел). Возможны только восемь записей. Если стек уже полон, команда *Включить Master Control Relay* генерирует ошибку стека MCR (MCRF).


Элемент FUP	Параметры	Тип данных	Область памяти	Описание
	Отсутствуют	-	-	Открывает зону MCR

Рис. 20-14. Элемент “Включение Master Control Relay”

Выключение MCR

Команда *Выключить Master Control Relay* (MCR>) закрывает зону действия MCR, открытую последней. Эта команда делает это путем удаления записи RLO из стека MCR. RLO было сохранено командой *Включить Master Control Relay*. Запись, освобождаемая на другом конце LIFO-стека MCR, устанавливается в 0. Если стек уже пуст, команда *Выключить Master Control Relay* генерирует ошибку стека MCR (MCRF).


Элемент FUP	Параметры	Тип данных	Область памяти	Описание
	Отсутствуют	-	-	Закрывает зону MCR, которая была открыта последней

Рис. 20-15. Заккрытие Master Control Relay

MCR управляется стеком шириной в один бит и глубиной восемь записей (см. рис. 20-16). MCR активизировано, пока все восемь записей в стеке равны 1. Команда MCR< копирует RLO в стек MCR. Команда MCR> удаляет последнюю запись из стека и устанавливает освобожденный адрес стека в 1. Если происходит ошибка, например, если в последовательности имеется более восьми команд MCR> или Вы пытаетесь выполнить команду MCR>, когда стек пуст, активизируется сообщение об ошибке MCRF. Контроль стека MCR основан на указателе стека (MSP: 0 = пуст, 1 = одна запись, 2 = две записи, ..., 8 = восемь записей).

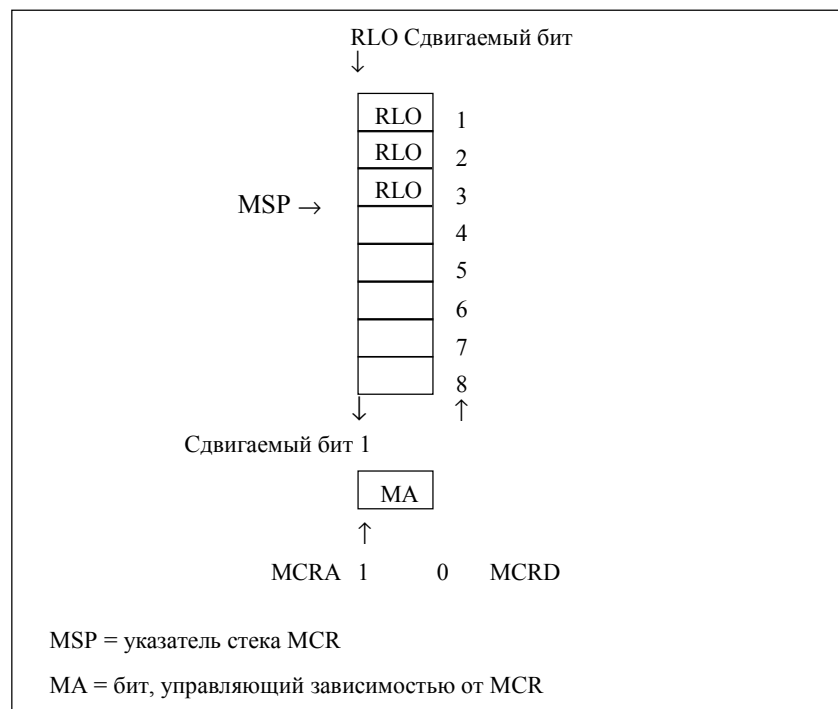


Рис. 20-16. Стек Master Control Relay

Команды MCR< и MCR> внутри Вашей программы всегда должны использоваться парами.

Команда MCR< принимает состояние RLO и копирует его в бит MCR.

Команда MCR> устанавливает бит MCR в 1 безусловно. Вследствие этого все остальные команды между командами MCRA и MCRD выполняются независимо от бита MCR (сведения о MCRA и MCRD см. выше).

Вложение команд MCR< и MCR>

Команды MCR< и MCR> могут быть вложенными. Максимальная глубина вложения равна восьми, иными словами, Вы можете написать максимум восемь команд MCR< последовательно, прежде чем вставить команду MCR>. Вы должны запрограммировать одинаковое количество команд MCR< и MCR>.

Если производится вложение команд MCR<, то формируется бит MCR более низкого уровня вложенности. Затем команда MCR< комбинирует текущее RLO с текущим битом MCR в соответствии с таблицей истинности для И.

Когда команда MCR> завершает уровень вложения, она извлекает бит MCR из следующего более высокого уровня.

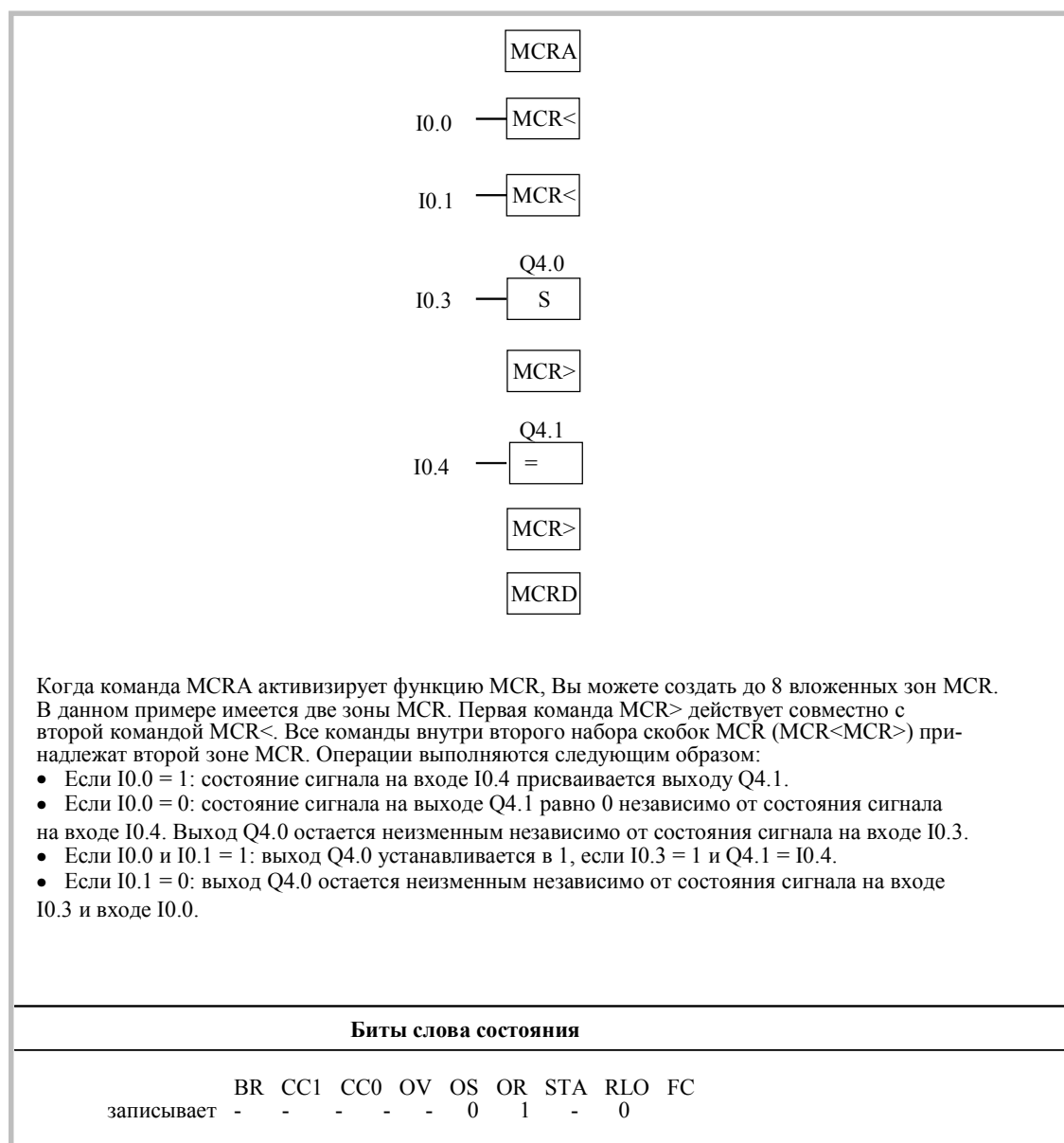


Рис. 20-17. Выключение Master Control Relay