

Обзор главы

В разделе	Вы найдете	на стр.
6.1	Элементы и блоки	6–2
6.2	Булева логика и таблицы истинности	6–7
6.3	Значение регистров CPU в командах	6–13

6.1. Элементы и блоки

Команды КОР

Команды КОР состоят из элементов и блоков, графически объединяемых в сети. Элементы и блоки можно разделить на следующие группы:

Операция как элемент

СТЕР 7 представляет часть операций КОР в виде отдельных элементов, которые не нуждаются ни в операндах, ни в параметрах (см. рис. 6–1).


Элемент	Описание	Глава в этом руководстве
	Инвертирование результата логической операции	8.6

Рис. 6-1. Операция КОР как элемент без операнда и параметров

Операция как элемент с операндом

СТЕР 7 представляет часть операций КОР в виде отдельных элементов, для которых необходимо вводить параметры (см. рис. 6–2). Более подробную информацию по адресации Вы найдете в гл. 7.

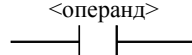
Операнд	Описание	Глава в этом руководстве
	Замыкающий контакт	8.2

Рис. 6-2. Операция КОР как элемент с операндом

Операция как элемент с операндом и значением

СТЕР 7 представляет часть операндов КОР в виде отдельных элементов, для которых необходимо ввести операнд и значение (напр., значение времени или счетное значение, см. рис. 6–3). Более подробную информацию по адресации Вы найдете в гл. 7.

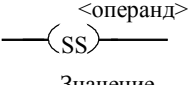
Элемент	Описание	Глава в этом руководстве
	Таймер как формирователь задержка включения с запоминанием	8.16

Рис. 6-3. Операция КОР как элемент с операндом и значением

Операция как блок параметрами

STEP 7 представляет часть операций KOP в виде блоков. Линий у этих с блоков показывают входы и выходы (см. рис. 6–4). Входы находятся с левой, а выходы с правой стороны блока. Введите входные параметры. Программное обеспечение STEP 7 уже предоставляет в распоряжение большинство выходов, за исключением некоторых выходов, которые Вы должны задать. Для параметров Вы должны использовать специфическую нотацию отдельных типов данных.

Описание параметров разрешающего входа (EN) и разрешающего выхода (ENO) Вы найдете ниже. Другую информацию о параметрах входов и выходов Вы найдете в описаниях отдельных операций в данном руководстве.

Блок	Описание	Глава в этом руководстве
<div><div>DIV_R</div><div>EN ENO</div><div>— IN1</div><div>— IN2 OUT —</div></div>	Деление чисел с плавающей точкой	12.5

Рис. 6-4. Операция KOP как блок с входами и выходами

Параметры разрешающего входа и разрешающего выхода

Если разрешающий вход (EN) KOP-блока активизирован, то благодаря этому блок выполняет определенную функцию. Если эта функция обрабатывается блоком без ошибок, то разрешающий выход пропускает ток дальше в цепь тока. Параметры EN и ENO KOP-блока имеют тип BOOL и могут находиться в областях памяти E, A, M, D или L (см. таблицы 6–1 и 6–20).

EN и ENO функционируют в соответствии со следующими принципами:

- Если EN не активизирован (т.е. состояние сигнала равно “0”), то блок не выполняет функцию, и ENO не активизируется (т.е. состояние сигнала тоже равно “0”).
- Если EN активизирован (т.е. состояние сигнала равно “1”) и соответствующий блок выполняет свою функцию без ошибок, то ENO тоже активизируется (т.е. состояние сигнала тоже равно “1”).
- Если EN активизирован (т.е. состояние сигнала равно “1”) и при обработке функции возникает ошибка, то ENO не активизируется (т.е. состояние сигнала равно “0”).

Ограничения для блоков и катушек

Вы не можете разместить блоки и коннекторы в пути тока, который не начинается на левой токовой шине. Исключением являются операции сравнения.

Области памяти и функции

Большинство операндов в KOP относятся к областям памяти. Следующая таблица показывает их виды и функции.

Таблица 6–1. Области памяти и их функции

Название области	Функция области памяти	Доступ к области через	
		единицы следующих размеров:	Сокращение
Отображение процесса на входах	В начале цикла операционная система читает входы с процесса и записывает значения в этой области. Программа применяет эти значения при циклической обработке.	Вход Входной байт Входное слово Двойное входное слово	E EB EW ED
Отображение процесса на выходах	Во время цикла программа рассчитывает выходные значения и сохраняет их в этой области. В конце цикла операционная система считывает рассчитанные выходные значения из этой области и пересылает их на выходы к процессу.	Выход Выходной байт Выходное слово Двойное выходное слово	A AB AW AD
Меркер	Эта область предоставляет место в памяти для промежуточных результатов, рассчитанных программой.	Меркер Меркерный байт Меркерное слово Двойное меркерное слово	M MB MW MD
Периферийная область: внеш. входы	Эта область дает Вашей программе возможность прямого доступа к модулям ввода и вывода (периферийные входы и выходы).	Периферийный входной байт Периферийное входное слово Периферийное двойное входное слово	PEB PEW
Периферийная область: внеш. выходы		Периферийный выходной байт Периферийное выходное слово Периферийное двойное выходное слово	PED PAB PAW PAD
Таймеры	Таймеры - это функциональные элементы в KOP. Эта область предоставляет место в памяти для таймерных ячеек. В этой области датчик импульсов времени обращается к таймерным ячейкам для их актуализации путем уменьшения значения времени. Таймерные операции обращаются к этим ячейкам.	Таймер (T)	T
Счетчики	счетчики - это функциональные элементы в KOP. Эта область предоставляет место в памяти для счетчиков. К ней обращаются операции счета.	счетчик (Z)	Z
Блок данных	В этой области содержатся данные, к которым можно обратиться из любого блока. Если Вам нужно одновременно открыть два различных блока данных, то Вы можете один из них открыть командой "AUF DB", а другой - командой "AUF DI". Нотация операндов, напр., L DBWi и L DIWi, определяет, к какому блоку данных производится обращение. Хотя командой "AUF DI" Вы можете обратиться к любому блоку данных, эта команда применяется главным образом для открытия экземпляров блоков данных, поставленных в соответствие функциональным блокам (FB) и системным функциональным блокам (SFB). Дальнейшую информацию о FB и SFB Вы найдете в Руководстве по программированию /234/ и в Руководстве пользователя /231/	Блок данных, открытый командой "AUF DB": Бит данных Байт данных Слово данных Двойное слово данных Блок данных, открытый командой "AUF DI": Бит данных Байт данных Слово данных Двойное слово данных	DBX DBB DBW DBD DIX DIB DIW DID
Локальные данные	Эта область содержит временные данные кодового блока (DB, FB или FC). Эти данные называются динамическими локальными данными. Они служат в качестве промежуточной памяти. Когда кодовый блок закрывается, эти данные теряются. Эти данные содержатся в стеке локальных данных (L-Stack).	Врем. локальный бит Врем. локальный байт Врем. локальное слово Врем. локальное двойное слово	L LB LW LD

В таблице 6–2 приведены максимальные диапазоны адресов различных областей памяти. Диапазоны адресов для Вашего CPU возьмите, пожалуйста, из руководства для соответствующего CPU S7-300.

Таблица 6–2. Области памяти и их диапазоны адресов

Название области	Доступ к области через		Максимальный диапазон адресов	
	единицы следующих размеров:	Сокр.		
Отображение процесса на входах	Вход	E	от 0.0 до 65 535.7	
	Входной байт	EB	от 0 до 65 535	
	Входное слово	EW	от 0 до 65 534	
	Двойное входное слово	ED	от 0 до 65 532	
Отображение процесса на выходах	Выход	A	от 0.0 до 65 535.7	
	Выходной байт	AB	от 0 до 65 535	
	Выходное слово	AW	от 0 до 65 534	
	Двойное выходное слово	AD	от 0 до 65 532	
Меркер	Меркер	M	от 0.0 до 255.7	
	Меркерный байт	MB	от 0 до 255	
	Меркерное слово	MW	от 0 до 254	
	Двойное меркерное слово	MD	от 0 до 252	
Периферийная область внешние входы	Периферийный входной байт	PEB	от 0 до 65 535	
	Периферийное входное слово	PEW	от 0 до 65 534	
	Периферийное двойное входное слово	PED	от 0 до 65 532	
Периферийная область внешние выходы	Периферийный выходной байт	PAB	от 0 до 65 535	
	Периферийное выходное слово	PAW	от 0 до 65 534	
	Периферийное двойное выходное слово	PAD	от 0 до 65 532	
Таймеры	Таймер	T	от 0 до 255	
Счетчики	Счетчик	Z	от 0 до 255	
Блок данных	Блок данных, открытый командой DB -(OPN)			
		Бит данных в блоке данных	DBX	от 0.0 до 65 535.7
		Байт данных	DBB	от 0 до 65 535
		Слово данных	DBW	от 0 до 65 534
		Двойное слово данных	DBD	от 0 до 65 532
	Блок данных, открытый командой DI -(OPN)			
		Бит данных в экземпляре DB	DIX	от 0.0 до 65 535.7
		Байт данных	DIB	от 0 до 65 535
		Слово данных	DIW	от 0 до 65 534
		Двойное слово данных	DID	от 0 до 65 532
Локальные данные	Бит локальных данных	L	от 0.0 до 65 535.7	
	Байт локальных данных	LB	от 0 до 65 535	
	Слово локальных данных	LW	от 0 до 65 534	
	Двойное слово локальных данных	LD	от 0 до 65 532	

6.2. Булева логика и таблицы истинности

Поток сигнала

Программа КОР следует за потоком сигнала между токовыми шинами, проходя через различные входы, выходы, а также другие элементы и блоки. Многие операции КОР работают по правилам булевой логики.

Каждая логическая операция опрашивает состояние сигнала электрического контакта на “0” (не активизирован, выключен) или на “1” (активизирован, включен) и вслед за этим выдает результат. Затем операция или сохраняет это результат, или выполняет с ним булеву логическую операцию. Результат логической операции (Verknüpfungsergebnis) называется VKE.

Далее правила булевой логики наглядно поясняются с помощью замыкающих и размыкающих контактов.

Замыкающий контакт

Рис. 6–5 показывает два изображения релейно-контактной схемы с одним контактом между токовой шиной и катушкой реле. В нормальном состоянии контакт открыт. Если контакт не активизирован, он остается открытым. Состояние сигнала открытого контакта равно “0” (не активизирован).

Если контакт остается открытым, то ток не может протекать от токовой шины к катушке в конце токовой цепи. Если контакт активизирован (состояние сигнала контакта равно “1”), то ток течет к катушке. Переключательная схема слева на рис. 6–5 показывает замыкающий контакт в таком виде, как он иногда изображается на релейно-контактных схемах. Переключательная схема справа в этом примере используется, чтобы показать, что контакт был активизирован и поэтому замкнут.

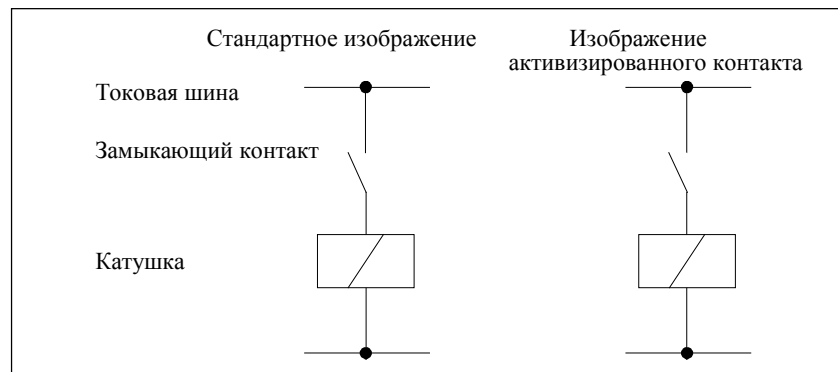


Рис. 6-5. Релейно-контактная схема с замыкающим контактом

С помощью операции *Замыкающий контакт* (см. гл. 8.2) Вы можете опросить состояние сигнала замыкающего контакта. От результата опроса зависит, может ли ток протекать через контакт. Если ток может протекать через контакт, то операция дает результат “1”. Если ток не может протекать, то операция дает результат “0” (см. таблицу). Это результат теперь может быть сохранен операцией или использован для логического сопряжения.

Размыкающий

замкнут. Если контакт не активизирован, то он остается замкнутым. Сигнальное состояние контакта равно “0” (не активизирован).

Если контакт остается замкнутым, то ток может протекать от токовой шины через контакт, и катушка в конце цепи тока проводит ток. При активизации контакта (состояние сигнала контакта равно “1”) он размыкается и прерывает поток сигнала к катушке.

Переключательная схема слева на рис. 6–6 показывает размыкающий контакт так, как он иногда изображается в релейно-контактных схемах. Переключательная схема справа в этом примере используется, чтобы показать, что контакт активизирован и потому разомкнут.

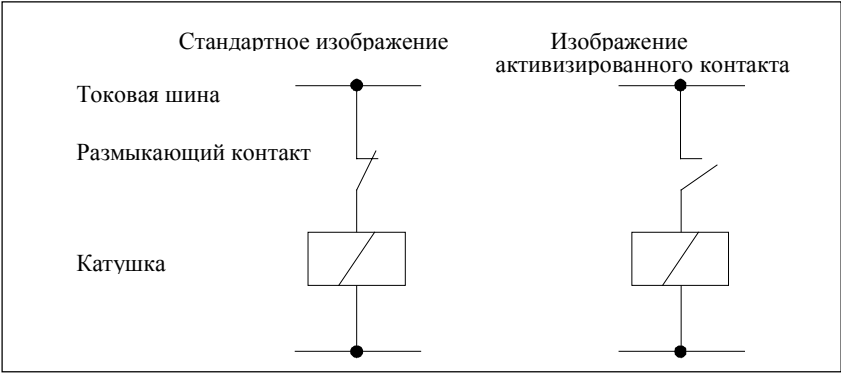


Рис. 6-6. Релейно-контактная схема с размыкающим контактом

С помощью операции *Размыкающий контакт* (см. гл. 8.3) Вы можете опросить состояние сигнала размыкающего контакта. От результата опроса зависит, может ли ток протекать через контакт. Если ток может протекать, то операция дает результат “1”. Если ток не может протекать, то операция дает результат “0” (см. рис. 6–7). Этот результат теперь может быть или сохранен операцией или использован для логического сопряжения.

Операция	Результат, если состояние сигнала контакта равно “1” (контакт активен)	Результат, если состояние сигнала контакта равно “0” (контакт не активен)
	1 (Ток может протекать, так как замыкающий контакт замкнут)	0 (Ток не может протекать, так как замыкающий контакт разомкнут)
	0 (Ток не может протекать, так как размыкающий контакт разомкнут)	1 (Ток может протекать, так как размыкающий контакт замкнут)

Рис. 6-7. Результат опроса состояния сигнала операциями “Замыкающий контакт” и “Размыкающий контакт”

Последовательное включение контактов

На рис. 6–8 показана цепь сопряжений операций КОР, представляющих два замыкающих контакта, последовательно соединенных с катушкой. Контакты обозначаются “Е” (“Eingang” - вход), а катушка - “А” (“Ausgang” - выход). При активизации замыкающего контакта он замыкается. Если активизированы (замкнуты) оба контакта в цепи сопряжений, то ток течет от токовой шины через оба контакта Е 1.0 и Е 1.1 к катушке. Катушка А 4.0 проводит ток.

В релейной схеме 1 оба контакта активизированы. При активизации замыкающего контакта он замыкается. Ток может протекать от токовой шины через замкнутые контакты к катушке в конце токовой цепи.

В релейных схемах 2 и 3 ток не может протекать к катушке, так как один из двух контактов не активизирован. Катушка ток не проводит.

В релейной схеме 4 ни один из двух контактов не активизирован. Оба контакта остаются разомкнутыми. Ток через катушку протекать не может. Катушка ток не проводит.

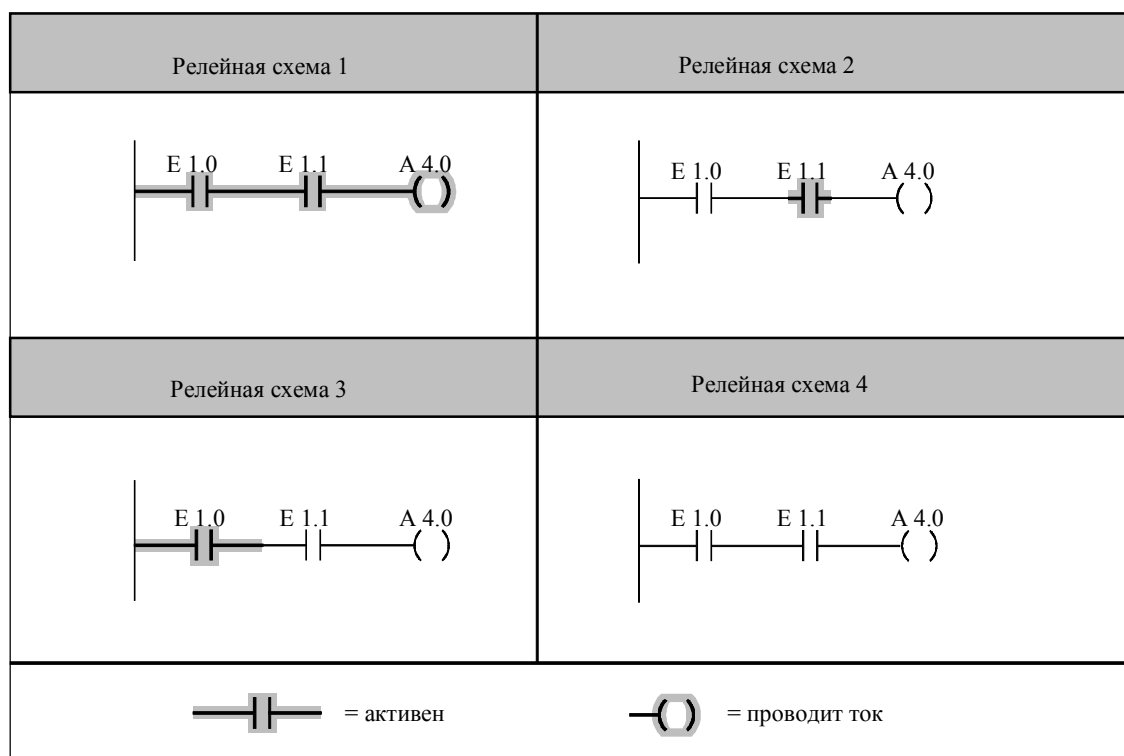


Рис. 6-8. Последовательное включение контактов с операцией “Замыкающий контакт”

**Последовательное
включение
контактов
с операцией
“Замыкающий
контакт”**

замкнут и ток может протекать через контакт. Результат “02” означает, что контакт разомкнут, вследствие чего поток сигнала на

Рис. 6–8 показывает контактный план, с помощью которого Вы можете запрограммировать последовательное включение двух замыкающих контактов с катушкой реле. Первая операция “Замыкающий контакт” в цепи сопряжения опрашивает состояние сигнала первого контакта в последовательном соединении (вход E 1.0) и выдает в зависимости от результата опроса “1” или “0” (см. таблицу). Результат “1” означает, что контакт замкнут и ток может протекать через контакт. Результат “02” означает, что контакт разомкнут, вследствие чего поток сигнала на контакте прерывается.

Первая операция *Замыкающий контакт* копирует результат “1” или “0” в меркерный бит в слове состояния контроллера. Этот бит называется бит VKE. VKE - это сокращение от “Verknüpfungsergebnis” (“Результат логической операции”).

Вторая операция *Замыкающий контакт* в цепи логического сопряжения опрашивает состояние сигнала второго контакта в последовательном соединении (E 1.1) и выдает в зависимости от результата опроса “1” или “0”, смотря по тому, замкнут или разомкнут контакт (см. таблицу). В этот момент вторая операция *Замыкающий контакт* выполняет логическое сопряжение.

Теперь операция берет результат опроса состояния сигнала на втором контакте и логически соединяет со значением, сохраненным в бите VKE, причем она заменяет сохраненное там старое значение. Операция “Катушка реле, выход” (см. гл. 8.4) присваивает это новое значение катушке (выход A 4.0).

Возможные результаты этого логического сопряжения могут быть представлены в “таблице истинности”. При этом “1” означает “истину”, а “0” - “ложь”. Возможные логические сопряжения и их результаты обобщены в табл. 6–3. При этом слова “контакт замкнут” и “ток может протекать” соответствуют высказыванию “истина”, а “контакт разомкнут” и “ток не может протекать” высказыванию “ложь” (для контактов: см. рис. 6–8).

Таблица 6–3. Таблица истинности И

Если результат опроса состояния сигнала контакта E 1.0	и результат опроса состояния сигнала контакта E 1.1	то результат логической операции на рис. 6–3
1 (контакт замкнут)	1 (контакт замкнут)	1 (ток может протекать)
0 (контакт разомкнут)	1 (контакт замкнут)	0 (ток не может протекать)
1 (контакт замкнут)	0 (контакт разомкнут)	0 (ток не может протекать)
0 (контакт разомкнут)	0 (контакт разомкнут)	0 (ток не может протекать)

Параллельное включение контактов

На рис. 6–9 показана цепь сопряжения операций КОР, которые представляют два параллельно включенных замыкающих контакта с катушкой реле. Контакты обозначены “Е” (“Eingang” - вход), а катушка - “А” (“Ausgang” - выход). При активизации замыкающего контакта он замыкается. Если контакт Е 1.0 или Е 1.1 в цепи логического сопряжения активизирован (замкнут), то ток может протекать от токовой шины через контакт к катушке. Катушка А 4.0 проводит ток. Если активизированы оба контакта в цепи логического сопряжения, то ток тоже может протекать к катушке.

В релейно-контактных схемах 1 и 2 один контакт активизирован, а другой нет. При активизации замыкающего контакта он замкнут. Ток может протекать от токовой шины через замкнутый контакт к катушке в конце цепи тока. Так как оба контакта включены параллельно, то этим достигается, что при замыкании одного из двух контактов ток может протекать к катушке в конце цепи тока.

В релейно-контактной схеме 3 активизированы оба контакта, благодаря чему ток может протекать к катушке в конце токовой цепи через оба замкнутых контакта.

В релейно-контактной схеме 4 не активизирован ни один контакт. Оба контакта остаются разомкнутыми. Ток к катушке протекать не может. Катушка не проводит ток.

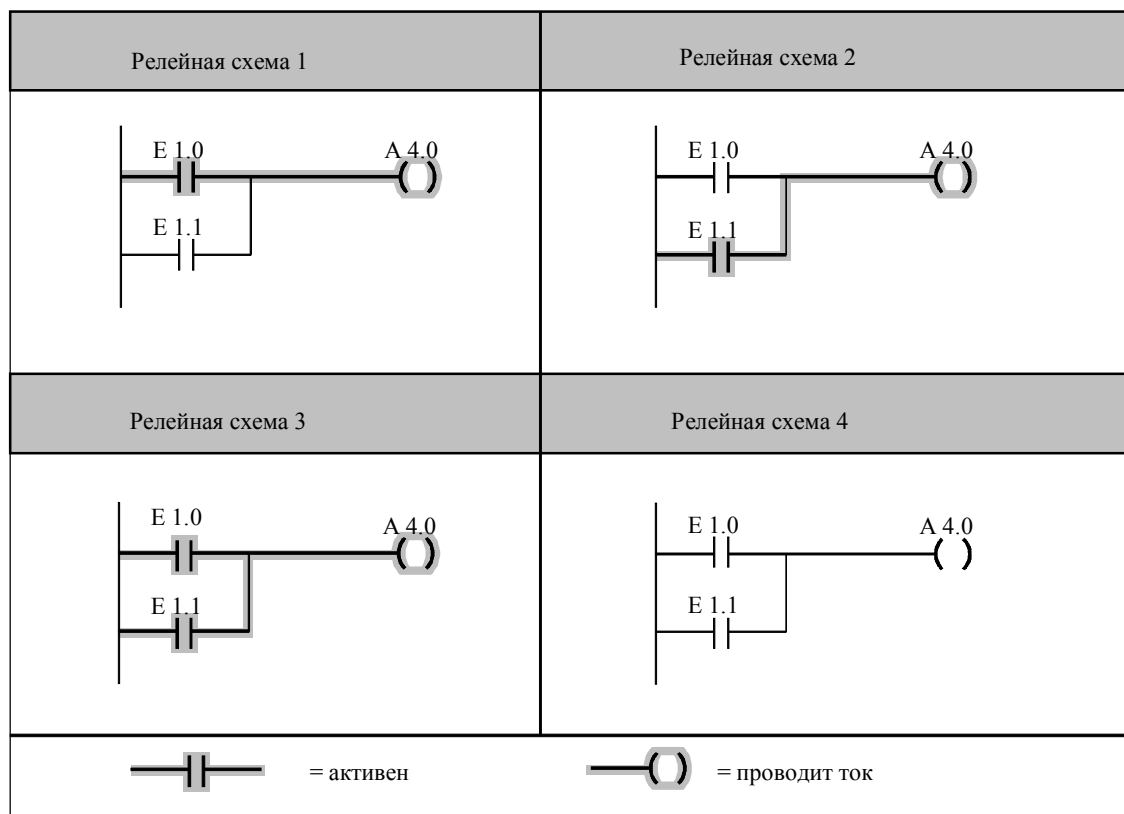


Рис. 6-9. Параллельное включение контактов с операцией “Замыкающий контакт”

**Параллельное
включение
контактов с
операцией
“Замыкающий
контакт”**

Результат “0” означает, что контакт разомкнут, благодаря сему поток сигнала на контакте прерывается. Первая операция *Замыкающий контакт* копирует результат “1” или “0” в меркерный бит в слове состояния контроллера. Этот бит называется бит VKE. VKE - это сокращение от “Verknüpfungsergebnis” (“Результат логической операции”).

Вторая операция *Замыкающий контакт* в цепи логического сопряжения опрашивает состояние сигнала второго контакта (E1.1) и выдает в зависимости от результата опроса “1” или “0”, смотря по тому, замкнут контакт или разомкнут (см. таблицу). В этот момент вторая операция *Замыкающий контакт* выполняет логическое сопряжение. Теперь операция берет результат опроса состояния сигнала на втором контакте и логически сопрягает этот результат со значением, сохраненным в бите VKE. Результат этого сопряжения (“1” или “0”) сохраняется в бите VKE слова состояния, причем он заменяет в нем сохраненный старый результат. Операция *Катушка реле, выход* (см. гл. 3.4) присваивает это новое значение катушке (выход 4.0).

Возможные результаты этого логического сопряжения могут быть представлены в “таблице истинности”. При этом “1” означает “истину”, а “0” - “ложь”. Возможные логические сопряжения и их результаты обобщены в табл. 6–4. При этом слова “контакт замкнут” и “ток может протекать” соответствуют высказыванию “истина”, а “контакт разомкнут” и “ток не может протекать” высказыванию “ложь” (для контактов: см. рис. 6–9).

Таблица 6–4. Таблица истинности ИЛИ

Если результат опроса состояния сигнала контакта E 1.0	и результат опроса состояния сигнала контакта E 1.1	то результат логической операции на рис. 6–9
1 (контакт замкнут)	0 (контакт разомкнут)	1 (ток может протекать)
0 (контакт разомкнут)	1 (контакт замкнут)	1 (ток может протекать)
1 (контакт замкнут)	1 (контакт замкнут)	1 (ток может протекать)
0 (контакт разомкнут)	0 (контакт разомкнут)	0 (ток не может протекать)

6.3. Значение регистров CPU в командах

Объяснение

Регистры помогают CPU выполнять логические операции, арифметические операции и операции сдвига или циклического сдвига. Эти регистры описаны ниже.

Аккумуляторы

Аккумуляторы (АККУ) - это универсальные регистры для обработки байтов, слов и двойных слов. Они имеют размер 32 бита.

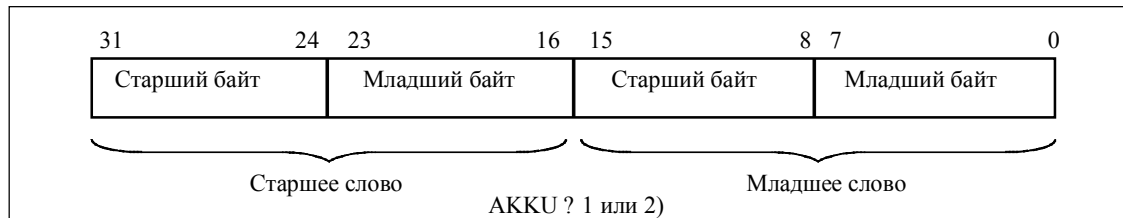


Рис. 6-10. Области аккумулятора

Слово состояния

Слово состояния содержит биты, к которым Вы можете обращаться в операндах битовых логических операций. Следующие разделы объясняют значения битов с 0 по 8.

$2^{15} \dots$	$\dots 2^9$	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
		BIE	A1	A0	OV	OS	OR	STA	VKE	/ER

Рис. 6-11. Структура слова состояния

Изменение битов в слове состояния

0	устанавливает состояние сигнала в 0
1	устанавливает состояние сигнала в 1
x	изменяет состояние
-	состояние остается неизменным

Первичный опрос

Бит 0 слова состояния называется битом первичного опроса (бит /ER: см. рис. 6–11). В начале сети КОР состояние сигнала бита /ER всегда равно “0”, разве только предыдущая сеть закончилась командой --(SAVE). (Косая черта перед сокращением ER указывает, что он инвертирован, т.е. состояние его сигнала в начале сети КОР всегда равно “0”).

Каждая операция логического сопряжения опрашивает состояние сигнала бита /ER и обрабатываемого контакта. Сигнальное состояние бита /ER управляет выполнением цепи логических сопряжений. Если бит /ER равен “0” (в начале сети КОР), то операция сохраняет результат в бите VKE слова состояния и устанавливает бит /ER в “1”. Этот процесс называется первичным опросом. Результат “1” или “0”, сохраняемый после первичного опроса в бите VKE, называют результатом первичного опроса.

Если сигнальное состояние бита /ER равно “1”, то операция логически сопрягает результат опроса состояния сигнала на обрабатываемом ею контакте с ранее образованным VKE и сохраняет этот результат в бите VKE.

Путь тока из операций КОР (цепь логических сопряжений) всегда заканчивается операцией вывода (“Установка выхода”, “Сброс выхода”, “Катушка реле, выход”) или операцией перехода, связанной с результатом логической операции. Эти операции сбрасывают бит /ER на “0”.

Результат логической операции

Бит 1 слова состояния называется битом VKE (VKE означает “Verknüpfungsergebnis” - “Результат логической операции”, см. рис. 6–11).

Этот бит сохраняет результат цепи операций логического сопряжения или операций сравнения. Состояние сигнала бита VKE дает информацию для потока сигнала. Состояние сигнала “1” может отображать поток сигнала (включен). Состояние сигнала “0” может отображать, что поток сигнала отсутствует (выключен).

Первая операция в сети КОР опрашивает состояние сигнала контакта и получает результат “1” или “0”. Операция сохраняет этот результат в бите VKE. Вторая операция в пути тока из операций КОР также опрашивает состояние сигнала контакта и получает результат. Теперь операция логически сопрягает этот результат по правилам булевой логики со значением, сохраненным в бите VKE слова состояния (см. выше “Первичный опрос” и гл. 8). Результат этой операции логического сопряжения сохраняется в бите VKE слова состояния и заменяет предыдущее значение в этом бите. Каждая следующая операция в пути тока выполняет логическое сопряжение с двумя величинами: результатом опроса сигнала на контакте и текущим VKE.

Вы можете назначить VKE при первичном опросе состояние булева меркера с помощью булевой логической операции. С помощью VKE Вы можете запустить также операцию перехода.

Бит состояния

Бит 2 слова состояния называется битом состояния (бит STA, см. рис. 6–11). Бит состояния сохраняет значение того бита, к которому производится обращение. Состояние логической операции, осуществляющей доступ к памяти для чтения (“Замыкающий контакт”, “Размыкающий контакт”), всегда равно значению бита, который эта операция опрашивает (бит, с которым она выполняет логическое сопряжение). Состояние логической операции, осуществляющей доступ к памяти для записи (“Установка выхода”, “Сброс выхода”, “Катушка реле, выход”) равно значению бита, в который операция производит запись. Бит состояния не имеет значения для логических операций, которые не обращаются к памяти. Эти операции устанавливают бит состояния в “1” (STA=1). Бит состояния не опрашивается операциями. Он анализируется только при тестировании программы (Programmstatus - статус программы).

Бит OR

Бит 3 слова состояния называется битом OR (см. рис. 6–11). Бит OR необходим, если Вы с помощью операций над контактами выполняете И перед ИЛИ. Логические сопряжения ИЛИ соответствуют параллельному включению контактов. Логические сопряжения И соответствуют последовательному включению контактов (см. гл. 6.2). Сопряжения И может содержать следующие операции: “Замыкающий контакт” и “Размыкающий контакт”. Бит OR указывает этим операциям, что ранее выполненное сопряжение И дало значение “1”, тем самым предвосхищая результат логического сопряжения ИЛИ. Любая другая операция, обрабатывающая биты, сбрасывает бит OR.

Бит OV

Бит 5 слова состояния называется битом OV (переполнение, см. рис. 6–11). Бит OV (переполнение) указывает на наличие ошибки. Он устанавливается арифметической операцией или операцией сравнения с числами с плавающей точкой, после которой возникла ошибка (переполнение, недопустимая операция, недопустимое число с плавающей точкой). Бит устанавливается (в случае ошибки) или сбрасывается в соответствии с результатом арифметической операции или операции сравнения.

Бит OS

Бит 4 слова состояния называется битом OS (переполнение с сохранением, см. рис. 6–11). Бит OS (переполнение с сохранением) устанавливается вместе с битом OV при наступлении ошибки. Так как бит OS при безошибочном выполнении арифметических операций остается неизменным (в противоположность битову OV), он сохраняет значение бита OV и указывает, произошла ли ошибка в одной из ранее выполненных операций. Бит OS сбрасывают следующие операции: SPS (Перейти, если OS=1, программирование на AWL), вызовы блоков и конец блока.

A1 и A0

Биты 7 и 6 слова состояния называются индикаторными битами 1 и 0 (A1 и A0, см. рис. 6–11). Биты A1 и A0 (индикаторные биты) информируют о следующих результатах или битах:

- результат арифметической операции
- результат операции сравнения
- результат цифровой операции
- биты, которые были сдвинуты из операнда операцией сдвига или циклического сдвига

Таблицы от 6–5 до 6–10 перечисляют значение A1 и A0, после того как Ваша программа выполнила определенные операции.

Таблица 6–5. A1 и A0 после арифметических операций, без переполнения

A1	A0	Объяснение
0	0	Результат = 0
0	1	Результат < 0
1	0	Результат > 0

Таблица 6–6. A1 и A0 после арифметических операций (арифметика с фиксированной точкой, с переполнением)

A1	A0	Объяснение
0	0	Переполнение отрицательной области при сложении целых чисел (16 бит) и сложении целых чисел (32 бита)
0	1	Переполнение отрицательной области при умножении целых чисел (16 бит) и целых чисел (32 бита) Переполнение положительной области при сложении целых чисел (16 бит), вычитании целых чисел (16 бит), сложении целых чисел (32 бита), вычитании целых чисел (32 бита), получении дополнения до 2 16-битного целого числа, получении дополнения до 2 32-битного целого числа
1	0	Переполнение положительной области при умножении целых чисел (16 бит), умножении целых чисел (32 бита), делении целых чисел (16 бит), делении целых чисел (32 бита), Переполнение отрицательной области при сложении целых чисел (16 бит), вычитании целых чисел (16 бит), сложении целых чисел (32 бита), вычитании целых чисел (32 бита)
1	1	Деление на 0 при делении целых чисел (16 бит), делении целых чисел (32 бита) и получении остатка от деления (32 бита)

Таблица 6–7. A1 и A0 после арифметических операций (арифметика с плавающей точкой), с переполнением

A1	A0	Объяснение
0	0	Ступенчатая потеря значимости
0	1	Переполнение отрицательной области
1	0	Переполнение положительной области
1	1	Недопустимое число с плавающей точкой

Таблица 6–8. A1 и A0 после операций сравнения		
A1	A0	Объяснение
0	0	IN2 = IN1
0	1	IN2 < IN1
1	0	IN2 > IN1
1	1	IN1 или IN2 недопустимое число с плавающей точкой

Таблица 6–9. A1 и A0 после операций сдвига и циклического сдвига		
A1	A0	Объяснение
0	0	Последний сдвинутый бит = 0
1	0	Последний сдвинутый бит = 1

Таблица 6–10. A1 и A0 после цифровых логических операций		
A1	A0	Объяснение
0	0	Результат = 0
1	0	Результат > 0

Бит ВІЕ

Бит 8 слова состояния называется битом ВІЕ (двоичный результат, см. рис. 6–11). Бит ВІЕ является связующим звеном между обработкой битов и слов. Он эффективно обеспечивает двоичную интерпретацию результата операции со словом и его включение в цепь двоичных логических операций. ВІЕ представляет собой с этой точки зрения внутримашинный меркер, в котором сохраняется VKE перед операцией со словом, изменяющей VKE, чтобы после этой операции VKE снова находилось в распоряжении для продолжения прерванной цепи битов.

Бит ВІЕ дает Вам возможность, например, программировать функциональный блок (FB) или функцию (FC) на AWL и вызвать этот FB или эту FC в KOP.

Когда Вы пишете функциональный блок или функцию, которые хотели бы вызвать из KOP, безразлично, пишете ли Вы FB или FC в AWL или KOP, Вы должны принимать во внимание бит ВІЕ. Бит ВІЕ соответствует разрешающему входу блока KOP (ENO) блока KOP. Вы сохраняете VKE в бите ВІЕ операцией SAVE (в AWL) или с помощью катушки --- (SAVE) (в KOP) в соответствии со следующими критериями:

- сохраните VKE равным "1" в бите ВІЕ для случая, когда FB или FC обрабатывается без ошибок
- сохраните VKE равным "0" в бите ВІЕ для случая, когда при обработке FB или FC происходит ошибка.

Запрограммируйте эти операции в конце FB или FC, так чтобы они выполнялись как последние операции в блоке.



Предупреждение

Бит ВІЕ может быть непреднамеренно сброшен в "0".

Если Вы пишете FB или FC в KOP и обрабатываете бит ВІЕ не так, как описано выше, то FB или FC может переписать бит ВІЕ другого FB или другой FC.

Во избежание этой ошибки сохраняйте VKE в конце FB или FC, как описано выше.

Значение EN/ENO

Параметры разрешающего входа (EN) и разрешающего выхода (ENO) блока KOP функционируют в соответствии со следующими принципами:

- Если EN не активизирован (т.е. состояние сигнала равно "0"), то блок свою функцию не выполняет, и ENO не активизируется (т.е. состояние сигнала тоже равно "0").
- Если EN активизирован (т.е. состояние сигнала равно "1") и соответствующий блок выполняет свою функцию без ошибок, то ENO тоже активизируется (т.е. состояние сигнала тоже равно "1").
- Если EN активизирован (т.е. состояние сигнала равно "1") и при обработке функции возникает ошибка, то ENO не активизируется (т.е. состояние сигнала равно "0").

Если Вы в своей программе вызываете системный функциональный блок (SFB) или системную функцию (SFC), то SFB или SFC указывает через сигнальное состояние бита ВІЕ, выполнил ли CPU эту функцию без ошибок или с ошибкой:

- Если при исполнении возникает ошибка, то бит ВІЕ равен "0".
- Если функция была обработана без ошибок, то бит ВІЕ равен "1".