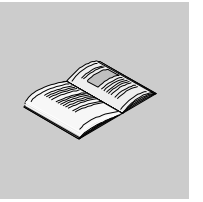
Twido Программируемые Контроллеры

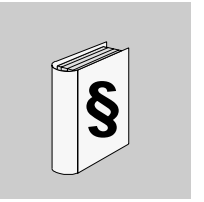
Справочное руководство по программному обеспечению

TWD USE 10AE Версия 2.1

Оглавление



Информация о безопасности



Важная информация

Обратите внимание

Внимательно прочитайте все инструкции и посмотрите оборудование для ознакомления с устройством до установки, использования или обслуживания. Следующие специальные сообщения, изображенные в этой документации или на оборудовании, предупреждают о потенциальных опасностях или привлекают внимание к информации, которая разъясняет или упрощает порядок действий.

Дополнение этой информации к ярлыку безопасности Опасность или Предупреждение указывает, что существует электрическая опасность, приводящая к персональному вреду, если не соблюдается инструкция.

Это символ тревоги. Он используется, чтобы оповестить Вас о потенциальной опасности персонального вреда. Исполняйте все указания безопасности, которые следуют за этим символом, чтобы избежать возможного вреда или смерти.

ОПАСНОСТЬ

ОПАСНОСТЬ указывает неминуемую опасную ситуацию, которая, если ее не избежать, **приведет** к смерти, серьезному ущербу или повреждению оборудования.

ПРЕДУПРЕЖДЕНИЕ

ПРЕДУПРЕЖДНИЕ указывает потенциально опасную ситуацию, которая, если ее не избежать, **может привести** к к смерти, серьезному ущербу или повреждению оборудования.

⚠ ОСТОРОЖНО

ОСТОРОЖНО указывает потенциально опасную ситуацию, которая, если ее не избежать, **может привести** к повреждению или поломке оборудования.

Пожалуйста, обратите внимание

Электрическое оборудование должно обслуживаться только квалифицированным персоналом. Schneider Electric не несет никакой ответственности за любые последствия, произошедшие из использования этого материала.
Этот документ не предназначен в качестве инструкции для неподготовленного персонала. Сборочная и установочная инструкции представлены в справочном руководстве по аппаратным средствам Twido, TWD USE 10AE.

Дополнительная информация о безопасности

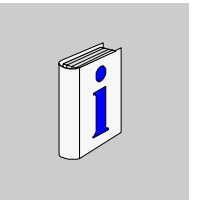
Ответственный за применение, реализацию или использование этого изделия должен гарантировать, что необходимые соображения разработки были включены в каждое применение, полностью придерживаясь действующих законов, выполнения требований безопасности, инструкций, положений и стандартов.

Общие предупреждения и предостережения

	**ПРЕДУПРЕЖДЕНИЕ** **ОПАСНОСТЬ ВЗРЫВА** ● Замена компонентов может навредить пригодности для класса 1, пункт 2. ● Не разъединяйте оборудование, если не было выключено питание или если неизвестно, является ли зона опасной. **Несоблюдение этих правил может привести к серьёзному ущербу или повреждению оборудования.**

	**ПРЕДУПРЕЖДЕНИЕ** **НЕПРЕДСКАЗУЕМАЯ РАБОТА ОБОРУДОВАНИЯ** ● Выключайте питание перед установкой, удалением, монтажом или техническим обслуживанием. ● Это изделие не предназначено для использования в критически опасных функциях механизмов. Там, где существует опасность для персонала и/или оборудования, используйте жестко смонтированные блоки безопасности. ● Не разбирайте, не ремонтируйте и не изменяйте модули. ● Этот контроллер разработан для использования в закрытом корпусе. ● Устанавливайте модули при описанных рабочих условиях окружающей среды. ● Используйте источники электропитания датчиков только для подвода энергии на датчики, соединенные с модулем. ● Используйте плавкий предохранитель, одобренный IEC60127, на линии электропитания и выходном контуре схемы для удовлетворения требованиям по току и напряжению. Рекомендуемый плавкий предохранитель: Liitlefuse 5x20 мм типа 218000 серии тип T. **Несоблюдение этих правил может привести к серьёзному ущербу или повреждению оборудования.**

О книге



Обзор

Область действия документа

Это справочник по программному обеспечению для программируемых контроллеров Twido. Он состоит из следующих основных частей:

- Описание программного обеспечения Twido и введение в основы, необходимые для программирования контроллеров Twido.
- Описание коммуникаций, управления аналоговыми входами/выходами, установки интерфейсного модуля шины AS-Interface и других специальных функций.
- Описание языков программирования, использующихся при создании программ для Twido.
- Описание инструкций и функций контроллеров Twido.

Замечание о пригодности

Информация в этом справочнике применима **только** для программируемых контроллеров Twido.

Предупреждения, связанные с продуктом

Шнейдер Электрик не несёт никакой ответственности за любые ошибки в этом документе. Ни одна из частей этого документа не может быть воспроизведена ни в какой форме, включая электронную, без предварительного письменного согласия Шнейдер Электрик.

Комментарии пользователей

Мы будем рады вашим комментариям по поводу этого документа. Вы можете написать нам по e-mail на TECHCOMM@modicon.com

Описание программного обеспечения Twido

I

Обзор

Тема этой части

В этой части представлено введение в языки программирования и базовая информация, необходимая при создании программ управления для программируемых контроллеров Twido.

Содержание этой части

Эта часть содержит следующие темы:

Введение в программное обеспечение Twido

1

Обзор

Предмет

В этой главе представлено краткое введение в TwidoSoft, в ПО для программирования и конфигурирования контроллеров Twido и в языки программирования лестничной логики, списка инструкций и Grafcet.

Содержание главы

Эта глава содержит следующие темы:

Введение в TwidoSoft

Введение

TwidoSoft является графической средой разработки для создания, конфигурирования и поддержки приложений для программируемых контроллеров Twido. TwidoSoft позволяет создавать программы на различных типах языков (См. *Языки Twido , стр. 19*) и затем переносить приложение на контроллер.

TwidoSoft

TwidoSoft является 32-битной Windows-программой для персонального компьютера, работающей под управлением операционных систем Microsoft Windows 98 Second Edition, Microsoft Windows 2000 Professional или Microsoft Windows XP.

Основные черты TwidoSoft:

- Стандартный пользовательский интерфейс Windows
- Программирование и конфигурирование контроллеров Twido
- Связь с контроллером и контроль

Примечание: Для связи контроллера с персональным компьютером используется протокол TCP/IP. Необходимо, чтобы этот протокол был установлен на персональном компьютере.

Минимальная конфигурация

Минимальная конфигурация для использования TwidoSoft:

- Pentium 300MHz,
- 128 Mb ОЗУ,
- 40 Mb свободного места на жестком диске.

Введение в языки Twido

Введение

Программируемый контроллер считывает значения на входах, записывает значения на выходы и обрабатывает логические выражения под управлением программы. Создание управляющей программы для контроллера Twido состоит из написания последовательности инструкций на одном из языков программирования Twido.

Языки Twido

Следующие языки могут использоваться при создании программ для управления Twido:

- Язык списка инструкций:
  Программа на языке списка инструкций представляет собой набор логических выражений, записанных как последовательность булевых инструкций.
- Лестничные диаграммы:
  Лестничная диаграмма является графическим средством для отображения логического выражения.
- Язык Grafcet:
  Язык Grafcet состоит из набора шагов и переходов. Twido поддерживает использование списка инструкций Grafcet, но не графический Grafcet.

Вы можете использовать персональный компьютер для создания и редактирования управляющих программ для Twido, используя эти языки программирования.

Обратимость языков списка инструкций и лестничной логики позволяет удобно переносить программы с одного языка на другой.

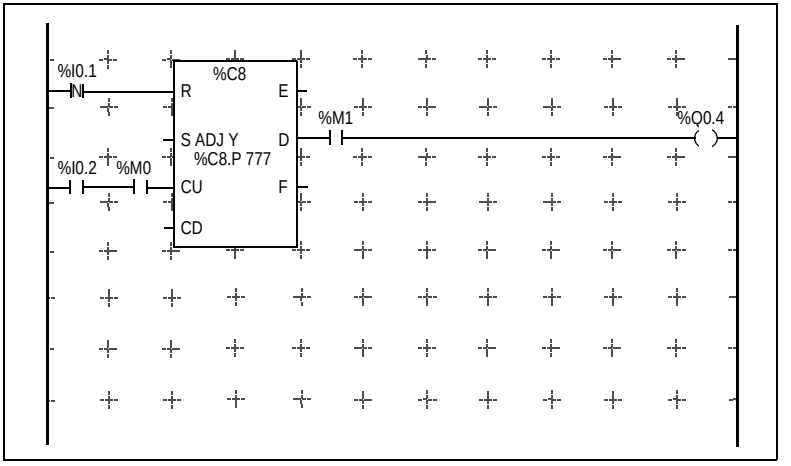
Язык списка инструкций

Программа, написанная на языке списка инструкций, состоит из набора инструкций, последовательно выполняемых контроллером. Ниже приведен пример программы на языке списка инструкций.

```
0    BLK     %C8
1    LDF     %I0.1
2    R
3    LD      %I0.2
4    AND     %M0
5    CU
6    OUT_BLK
7    LD      D
8    AND     %M1
9    ST      %Q0.4
10   END_BLK
```

Лестничные диаграммы

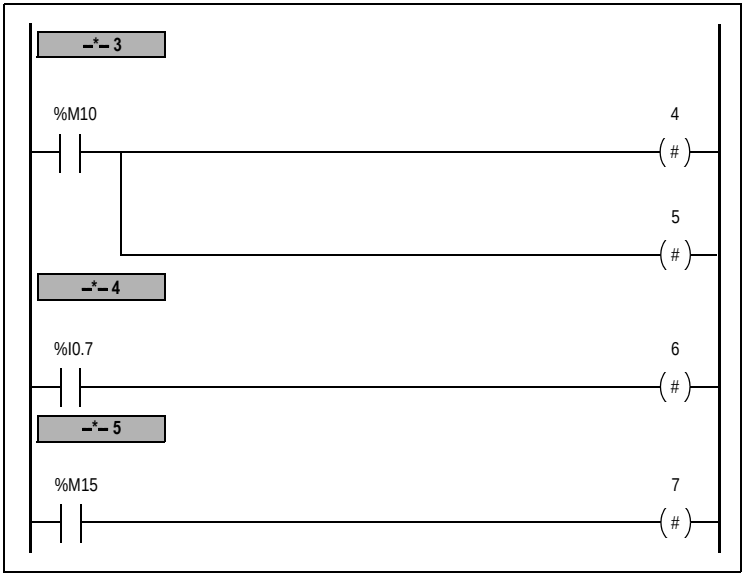
Лестничные диаграммы похожи на диаграммы релейной логики, которые представляют релейные схемы управления. Графические элементы, такие как катушки, контакты и блоки, представляют инструкции. Ниже приведен пример лестничной диаграммы.



Язык Grafcet

Аналитический метод Grafcet разбивает любую последовательную систему управления на набор шагов, с которыми ассоциируются действия, переходы и условия. Ниже показаны примеры инструкций Grafcet на языке списка инструкций и языке лестничной логики соответственно.

0	-*-	3
1	LD	%M10
2	#	4
3	#	5
4	-*-	4
5	LD	%I0.7
6	#	6
7	-*-	5
8	LD	%M15
9	#	7
10	...	



Объекты языка Twido

2

Обзор

Предмет

В этой главе представлено подробное описание объектов языка, использующихся для программирования контроллеров Twido.

Содержание главы

Эта глава содержит следующие темы:

Подтверждение действительности языковых объектов

Введение

Слова и битовые объекты действительны, если для них было выделено пространство в памяти контроллера. Для этого они должны быть использованы в приложении перед загрузкой в контроллер.

Пример

Диапазон для разрешенных объектов: от нуля до максимального значения, к которому было обращение для этого типа объекта. Например, если в Вашем приложении максимальное обращение к словам памяти было %MW9 , тогда будет выделено место в памяти от %MW0 до %MW9. Слово %MW10 в этом примере не действительно и не может быть достигнуто ни изнутри, ни снаружи.

Битовые объекты

Введение

Битовые объекты являются программными переменными битового типа, которые могут использоваться в качестве операндов и проверяться булевыми инструкциями. Ниже приведен список битовых объектов:

- Биты вх/вых
- Внутренние биты (биты памяти)
- Системные биты
- Биты шагов
- Биты, извлеченные из слов

Список битов, являющихся операндами

В следующей таблице перечислены и описаны все основные битовые объекты, которые используются в качестве операндов в логических инструкциях.

Тип	Описание	Адрес или значение	Максимальное число	Доступ к режиму записи (1)
Непосредственные значения	0 или 1 (Ложь или Истина)	0 или 1	-	-
Входы Выходы	Эти биты являются "логическим отображением" электрических состояний входов/выходов. Они хранятся в памяти данных и обновляются во время каждого сканирования программной логики.	%Ix.y.z (2) %Qx.y.z (2)	Примечание (4)	Нет Да
Внутренние (память)	Внутренние биты являются участками внутренней памяти, используемыми для хранения промежуточных значений во время исполнения программы. Примечание: неиспользуемые биты вх/вых не могут использоваться в качестве внутренних битов.	%Mi	128 TWDLCAA10 DRF, TWDLCAA16 DRF 256 для всех остальных контролл.	Да
Системные	Системные биты с %S0 по %S127 отслеживают правильную работу контроллера и правильное выполнение прикладной программы.	%Si	128	В соответствии с i
Функциональные блоки	Биты функциональных блоков соответствуют выходам функциональных блоков. Эти выходы могут либо прямо соединяться, либо использоваться как объекты.	%TMi.Q, %Ci.P и т.д.	Примечание (4)	Нет (3)
Реверсивные функциональные блоки	Функциональные блоки, запрограммированные используя реверсивные инструкции BLK, OUT_BLK и END_BLK.	E, D, F, Q, TH0, TH1	Примечание (4)	Нет

Тип	Описание	Адрес или значение	Максимальное число	Доступ к режиму записи (1)
Извлеченные из слов	Один из 16 битов в некоторых словах может извлекаться в качестве операнда.	Изменяется	Изменяется	Изменяется
Шаги Grafcet	Биты с %X1 по %Xi ассоциируются с шагами Grafcet. Бит шага Xi устанавливается в 1, когда активен соответствующий шаг, и устанавливается в 0, когда шаг дезактивируется.	%X21	62 TWDLCAA10DRF, TWDLCAA16DRF 94 TWDLCAA24DRF, Модульные контроллеры	Да

Примечания:

1. Записываются программой или при помощи редактора анимационных таблиц.
2. См. адресацию входов/выходов
3. Исключая %SBRi.j и %SCi.j, эти биты могут быть считаны и записаны.
4. Число определяется моделью контроллера.

Объекты-слова

Введение

Объекты-слова являются объектами, которые адресуются как 16-битные слова, хранящиеся в памяти, и могут содержать целое значение в пределах от -32768 до 32767 (исключая функциональный блок быстрого счетчика, для которого допустимые значения находятся между 0 и 65535).
Примеры объектов-слов:

- Прямые значения
- Внутренние слова (%MWi) (слова памяти)
- Постоянные слова (%KWi)
- Слова обмена вх/вых (%IWi, %QWi)
- Системные слова (%SWi)
- Функциональные блоки (конфигурация и/или исполняемые данные)

Форматы слов

Содержимое слов или значения хранятся в пользовательской памяти в 16-битном двоичном коде (дополнительный код) используя следующие соглашения:

	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	Позиция бита
	0	1	0	1	0	0	1	0	0	1	0	0	1	1	0	1	Состояние бита
|+	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1		Значение бита

В представлении двоичного числа со знаком 15-й бит выделен для представления знака закодированного числа:

- 15-й бит установлен в 0: содержимое слова является положительным значением.
- 15-й бит установлен в 1: содержимое слова является отрицательным значением (отрицательные значения выражаются в дополнительном коде).

Слова и прямые значения могут быть введены или извлечены в следующем формате:

- Десятичные
  Мин.: -32768, Макс.: 32767 (1579, например)
- Шестнадцатеричные
  Мин.: 16#0000, Макс.: 16#FFFF (например, 16#A536)
  Альтернативный синтаксис: #A536

Описание объектов-слов

В следующей таблице описываются объекты-слова.

Слова	Описание	Адрес или значение	Максимальное число	Доступ к режиму записи(1)
Непосредственные значения	Это целые значения в том же формате, что и 16-битные слова, что позволяет ассоциировать значения с этими словами.		-	Нет
	Основание 10	от -32768 до 32767		
	Основание 16	от 16#0000 до 16#FFFF		
Внутренние (Память)	Используются как "рабочие" слова для хранения значений в памяти данных во время исполнения. Слова с %MW0 по %MW255 считываются или записываются прямо из программы.	%MWi	3000	Да
Константы	Хранят константы или буквенно-цифровые сообщения. Их содержимое может быть записано или модифицировано только при использовании TwidoSoft во время конфигурации. Слова-константы с %KW0 по %KW63 для программы доступны только для чтения.	%KWi	256	Да, только при использовании TwidoSoft
Системные	Эти 16-битные слова выполняют несколько функций: ● Предоставляют доступ к данным, поступающим прямо из контроллера посредством чтения слов %SWi. ● Выполняют операции над приложением (например, настройка блоков планировщика).	%SWi	128	В соответствии с i
Функциональные блоки	Эти слова соответствуют текущим параметрам или значениям функциональных блоков.	%TM2.P, %Ci.P и т.д.		Да

Слова	Описание	Адрес или значение	Максимальное число	Доступ к режиму записи(1)
Слова сетевого обмена	Присвоены контроллерам, подсоединенным как удаленные. Эти слова используются для связи между контроллерами:			
	Сетевой вход	%INWi.j	4/ удаленную связь	Нет
	Сетевой выход	%QNWi.j	4/ удаленную связь	Да
Слова обмена вх/вых	Присвоены контроллерам, подсоединенным как удаленные. Эти слова используются для связи между контроллерами и модулями вх/вых.			
	Входы	%IWi.j	Прим. (2)	Нет
	Выходы	%QWi.j	Прим. (2)	Да
Извлеченные биты	Возможно извлечь один из 16 битов из следующих слов:			
	Внутренние	%MWi:Xk	1500	Да
	Системные	%SWi:Xk	128	Зависит от i
	Константы	%KWi:Xk	64	Нет
	Вход	%IWi.j:Xk	Прим. (2)	Нет
	Выход	%QWi.j:Xk	Прим. (2)	Да
	Сетевой вход	%INWi.j:Xk	Прим. (2)	Нет
	Сетевой выход	%QNWi.j:Xk	Прим. (2)	Да

Примечание:
1. Записываются программой или при помощи редактора анимационных таблиц.
2. Число определяется конфигурацией.

Объекты с плавающей точкой и двойные слова

Введение

TwidoSoft позволяет выполнять операции над объектами с ПТ и целыми двойными словами.
Число с ПТ является математическим аргументом, который содержит десятичную точку в своем выражении (примеры: 3.4E+38, 2.3 или 1.0).
Двойные целые слова состоят из 4 байт, хранящихся в памяти данных, и содержат значение от -2147483648 до +2147483647.

Формат и значение чисел с плавающей точкой

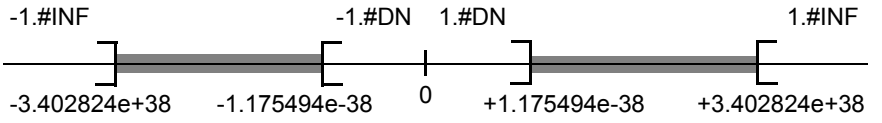
Используется формат, соответствующий стандарту IEEE STD 734-1985 (эквивалент IEC 559). Длина слов 32 бита, что соотв. одинарным числам с ПТ.
Таблица, показывающая формат числа с плавающей точкой:

Бит 31	Биты {30...23}	Биты {22...0}
Знак (S)	Экспонента (Exponent)	Мантисса (Fractional part)

Значение числа, имеющего указанный выше формат, определяется следующим выражением:

32-битное значение = $(-1)^{S} * 2^{(\text{Exponent}(-127))} *$ 1.Fractional part

Числа с ПТ могут быть представлены с и без экспоненты, но они всегда должны иметь десятичную точку (ПТ).
Значения чисел с ПТ находятся в диапазане от -3.402824e+38 до -1.175494e-38 и от 1.175494e-38 до 3.402824e+38 (выделены серым цветом на диаграмме). Они также имеют значение 0 (записывается 0.0).



Когда результат вычислений:

- Меньше -3.402824e+38, отображается символ -1.#INF(для -бесконеч.),
- Больше +3.402824e+38, отображается символ 1.#INF(для +бесконеч.),
- Между -1.175494e-38 и 1.175494e-38, округляется до 0.0. Значение в этих пределах не может быть введено, как значение с ПТ.
- Неопределен (например, квадратный корень из отрицательного числа) отображается символ 1.#NAN или -1.#NAN.

Точность представления 2-24. Для отображения чисел с ПТ, бесполезно отображать больше 6 цифр после плавающей точки.

Примечание:

- Значение "1285" интерпретируется как целое число; чтобы оно распознавалось как число с плавающей точкой, оно должно быть записано: "1285.0"

Аппаратная совместимость

Операции с ПТ и двойными словами поддерживаются не всеми контроллерами Twido.
В следующей таблице показана аппаратная совместимость:

Контроллер Twido	Поддерживаются двойные слова	Поддерживается ПТ
TWDLMDA40DUK	Да	Да
TWDLMDA40DTK	Да	Да
TWDLMDA20DUK	Да	Нет
TWDLMDA20DTK	Да	Нет
TWDLMDA20DRT	Да	Да
TWDLCAA24DRF	Да	Нет
TWDLCAA16DRF	Да	Нет
TWDLCAA10DRF	Нет	Нет

Проверка действитель-ности

Когда результат выходит за рамки допустимого диапазона, системный бит %S18 устанавливается в 1.
Биты слова статуса %SW17 указывают причину ошибки операции с плавающей точкой:
Различные биты слова %SW17:

%SW17:X0	Неправильная операция, результат не является числом (1.#NAN или -1.#NAN)
%SW17:X1	Зарезервировано
%SW17:X2	Деление на 0, результат является бесконечностью (-1.#INF или 1.#INF)
%SW17:X3	Результат по модулю больше +3.402824e+38, результат является бесконечностью (-1.#INF или 1.#INF)
с %SW17:X4 по X15	Зарезервировано

Это слово сбрасывается в 0 системой при холодном старте, а также программой для повторного использования.

Описание объектов с плавающей точкой и двойных слов

В следующей таблице описаны объекты с плавающей точкой и двойные слова:

Тип объекта	Описание	Адрес	Макси-мальное число	Доступ к режиму записи	Индексная форма
Непосредственные значения	Целые числа или десятичные дроби с форматом идентичным 32-битным объектам.	-	[-]	Нет	-
Внутренние слова с плавающей точкой	Объекты, используемые для хранения значений в памяти данных во время работы.	%MFi	1500	Да	%MFi[Индекс]
Внутренние двойные слова		%MDi	1500	Да	%MDi[Индекс]
Константа с плавающей точкой	Используются для хранения констант.	%KFi	128	Да, только при использовании TwidoSoft	%KFi[Индекс]
Константа двойной длины		%KDi	128	Да, только при использовании TwidoSoft	%KDi[Индекс]

Возможность перекрытия объектов

Одинарные, двойные и слова с плавающей точкой хранятся в пространстве данных в одной зоне памяти. Таким образом, слово с плавающей точкой %MFi и двойное слово %MDi соответствуют словам одинарной длины %MWi и %MWi+1 (слово %MWi содержит младшие значащие биты, слово %MWi+1 старшие значащие биты %MFi).
В следующей таблице показано, как перекрываются внутренние слова с плавающей точкой и слова двойной длины:

ПТ и двойные	Нечетный адрес	Внутренние слова
%MF0 / %MD0		%MW0
	%MF1 / %MD1	%MW1
%MF2 / %MD2		%MW2
	%MF3 / %MD3	%MW3
%MF4 / %MD4		%MW4
	...	%MW5
...		...
	%MFi / %MDi	%MWi
%MFi+1 / %MDi+1		%MWi+1

В следующей таблице показано, как перекрываются константы с плавающей точкой и константы двойной длины:

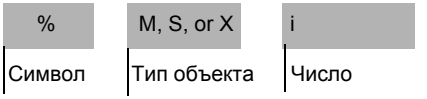
ПТ и двойные	Нечетный адрес	Внутренние слова
%KF0 / %KD0		%KW0
	%KF1 / %KD1	%KW1
%KF2 / %KD2		%KW2
	%KF3 / %KD3	%KW3
%KF4 / %KD4		%KW4
	...	%KW5
...		...
	%kFi / %kDi	%KWi
%KFi+1 / %KDi+1		%KWi+1

Пример:
%MF0 соответствует %MW0 и %MW1. %KF543 соответствует %KW543 и %KW544.

Адресация битовых объектов

Синтаксис

Используйте следующий формат для адресации внутренних, системных объектов и битов шагов:



Описание

В следующей таблице описаны элементы формата адресации.

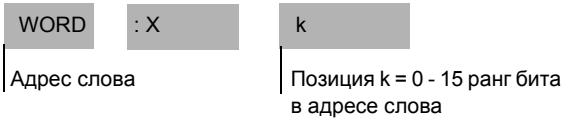
Группа	Элемент	Описание
Символ	%	Символ процента всегда предшествует программной переменной.
Тип объекта	M	Внутренние биты хранят промежуточные значения во время выполнения программы.
	S	Системные биты представляют статусную и управляющую информацию контроллеру.
	X	Биты шагов представляют статус шага.
Число	i	Максимальное значение числа зависит от числа конфигурируемых объектов.

Примеры адресации битовых объектов:

- %M25 = внутренний бит номер 25
- %S20 = системный бит номер 20
- %X6 = бит шага номер 6

Битовые объекты, извлеченные из слов

TwidoSoft используется для извлечения одного из 16 битов из слов. Адрес слова дополняется номером битом. Синтаксис извлечения:



Примеры:

- %MW5:X6 = бит номер 6 внутреннего слова %MW5
- %QW5.1:X10 = бит номер 10 слова выхода %QW5.1

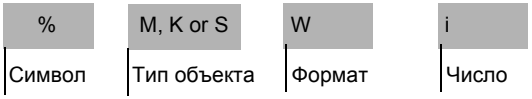
Адресация слов

Введение

Для адресации слов, за исключением адресации входов/выходов (см. *Адресация входов/выходов, стр. 39)* и функциональных блоков (см. *Объекты типа “Функциональный блок”, стр. 42)*, используется формат, описанный ниже.

Синтаксис

Используйте следующий формат для адресации внутренних, системных слов и слов-констант:



Описание

В следующей таблице описаны элементы формата адресации.

Группа	Элемент	Описание
Символ	%	Символ процента всегда предшествует внутреннему адресу.
Тип объекта	M	Внутренние слова хранят промежуточные значения во время выполнения программы.
	K	Слова-константы хранят постоянные значения или буквенно-цифровые сообщения. Их содержимое может быть записано или изменено только при помощи TwidoSoft.
	S	Системные слова представляют статусную и управляющую информацию контроллеру.
Синтаксис	W	16-битное слово.
Число	i	Максимальное значение числа зависит от числа конфигурируемых объектов.

Примеры адресации слов:

- %MW15 = внутреннее слово номер 15
- %KW26 = слово-константа номер 26
- %SW30 = системное слово номер 30

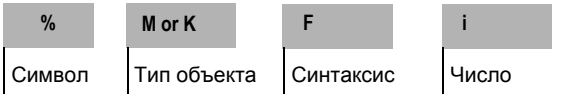
Адресация объектов с плавающей точкой

Введение

Для адресации объектов с плавающей точкой, за исключением адресации входов/выходов (см. *Адресация входов/выходов, стр. 39)* и функциональных блоков (см. *Объекты типа "Функциональный блок", стр. 42)*, используется формат, описанный ниже.

Синтаксис

Используйте следующий формат для адресации внутренних объектов и объектов-констант с плавающей точкой:



Описание

В следующей таблице описаны элементы формата адресации.

Группа	Элемент	Описание
Символ	%	Символ процента всегда предшествует внутреннему адресу.
Тип объекта	M	Внутренние объекты с плавающей точкой хранят промежуточные значения во время выполнения программы.
	K	Константы с плавающей точкой используются для хранения постоянных значений. Их содержимое может быть записано или изменено только при помощи TwidoSoft.
Синтаксис	F	32-битный объект.
Число	i	Максимальное значение числа зависит от числа конфигурируемых объектов.

Примеры адресации объектов с плавающей точкой:

- %MF15 = внутренний объект с плавающей точкой номер 15
- %KF26 = объект-константа с плавающей точкой номер 26

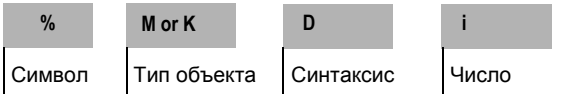
Адресация двойных слов

Введение

Для адресации двойных слов, за исключением адресации входов/выходов (см. *Адресация входов/выходов, стр. 39)* и функциональных блоков (см. *Объекты типа “Функциональный блок”, стр. 42)*, используется формат, описанный ниже.

Синтаксис

Используйте следующий формат для адресации внутренних двойных слов и двойных слов-констант:



Описание

В следующей таблице описаны элементы формата адресации.

Группа	Элемент	Описание
Символ	%	Символ процента всегда предшествует внутреннему адресу.
Тип объекта	M	Внутренние двойные слова хранят промежуточные значения во время выполнения программы.
	K	Двойные слова-константы хранят постоянные значения и буквенно-цифровые сообщения. Их содержимое может быть записано или изменено только при помощи TwidoSoft.
Синтаксис	D	32-битное двойное слово.
Число	i	Максимальное значение числа зависит от числа конфигурируемых объектов.

Примеры адресации двойных слов:

- %MD15 = внутреннее двойное слово номер 15
- %KD26 = двойное слово-константа номер 26

Адресация входов/выходов

Введение

У каждого входа/выхода в конфигурации Twido свой уникальный адрес: например, адрес "%I0.0.4" присвоен входу номер 4 контроллера.
Адреса входов/выходов могут быть присвоены следующим аппаратным средствам:

- Контроллеру, сконфигурированному как Master контроллер дистанционной связи;
- Контроллеру, сконфигурированному как Удаленный вх/вых ;
- Модулям расширения вх/вых.

Интерфейсный модуль шины TWDNOI10M3 AS-Interface имеет специальную систему адресации входов/выходов подчиненных устройств (См. *Адресация вх/вых , связанных с подчиненными устройствами, соединенными с шиной AS-Interface V2, стр. 184*).

Многократное обращение к выходу или катушке

В программе можно многократно обратиться к одному выходу или катушке. На аппаратные выходы подается результат последнего вычисления. Например, %Q0.0.0 может использоваться в программе больше одного раза, и не будет предупреждения о многократном вхождении, поэтому важно подтвердить только то уравнение, которое даст требуемое состояние на выходе.

ОСТОРОЖНО

Непредсказуемая работа

Проверка дублирования выходов или предупреждения не обеспечиваются. Просмотрите использование выходов и катушек перед изменений их состояния в приложении.

Несоблюдение этого предостережения может привести к ущербу или повреждению оборудования.

Формат

Используйте следующий формат для адресации входов/выходов.

%	I, Q	x	.	y	.	z
Символ	Тип объекта	Позиция контролл.	точка	Тип вх/вых	точка	Номер канала

Используйте следующий формат для адресации слов обмена входов/выходов.

%	I, Q	W	x	.	y
Символ	Тип	Формат	Позиция контролл.	точка	Тип вх/вых

Описание

В следующей таблице описаны элементы формата адресации входов/выходов.

Группа	Элемент	Значение	Описание
Символ	%	-	Символ процента всегда предшествует внутреннему адресу.
Тип объекта	I	-	Вход. “Логическое отображение” электрического состояния входа контроллера или модуля расширения вх/вых.
	Q	-	Выход. “Логическое отображение” электрического состояния выхода контроллера или модуля расширения вх/вых.
Позиция контроллера	x	0 1 - 7	Master контроллер (master контроллер дистанционной связи). Удаленный контроллер (slave контроллер дистанционной связи).
Тип вх/вых	y	0 1 - 7	Основные вх/вых (локальные вх/вых контроллера). Модули расширения вх/вых.
Номер канала	z	0 - 31	Номер канала вх/вых контроллера или модуля расширения вх/вых. Число доступных выводов зависит от модели контроллера или от типа модуля расширения вх/вых.

Примеры

В следующей таблице приведены примеры адресации входов/выходов.

Объект вх/вых	Описание
%I0.0.5	Входной вывод номер 5 базового контроллера (локальный вх/вых).
%Q0.3.4	Выходной вывод номер 4 модуля расширения вх/вых с адресом 3 для базового контроллера (вх/вых расширения).
%I0.0.3	Входной вывод номер 3 базового контроллера.
%I3.0.1	Входной вывод номер 1 удаленного контроллера с адресом 3 дистанционной связи.
%I0.3.2	Входной вывод номер 2 модуля расширения вх/вых с адресом 3 для базового контроллера.

Сетевая адресация

Введение

Удаленные контроллеры обмениваются прикладными данными с master контроллером по сети дистанционной связи Twido, используя сетевые слова %INW and %QNW.
См. *Коммуникации, стр. 83* для более подробной информации.

Формат

Используйте следующий формат для сетевой адресации.

Описание формата

В следующей таблице описаны элементы формата сетевой адресации.

Группа	Элемент	Значение	Описание
Символ	%	-	Символ процента всегда предшествует внутреннему адресу.
Тип объекта	IN	-	Сетевое входящее слово. Данные, передаваемые от master контроллера удаленному контроллеру.
	QN	-	Сетевое исходящее слово. Данные, передаваемые от удаленного контроллера master контроллеру .
Формат	W	-	16-битное слово.
Позиция контроллера	x	0 1 - 7	Master контроллер (master контроллер дистанционной связи). Удаленный контроллер (slave контроллер дистанционной связи).
Слово	j	0 - 3	Каждый удаленный контроллер использует от одного до четырех слов для обмена данными с master контроллером.

Примеры

В следующей таблице приведены примеры сетевой адресации.

Сетевой объект	Описание
%INW3.1	Сетевое слово номер 1 удаленного контроллера номер 3.
%QNW0.3	Сетевое слово номер 3 базового контроллера.

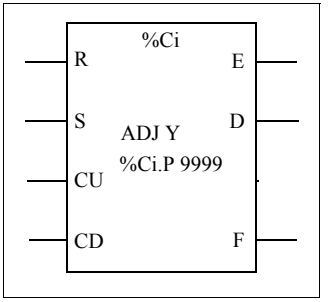
Объекты типа “Функциональный блок”

Введение

Функциональные блоки предоставляют битовые объекты и специальные слова, которые доступны программе.

Пример функциональ-ного блока

На следующем рисунке изображен функциональный блок счетчика.



Блок счетчика Up/down

Битовые объекты

Битовые объекты соответствуют выходам блока. Эти биты доступны инструкциям логической проверки любым из следующих методов:

- Прямо (например, LD E), если они связаны с блоком в обратимом программировании (см. *Основные принципы программирования функциональных блоков, стр. 277*).
- Указанием типа блока (например, LD %Ci.E).

Входы могут быть достигнуты в виде инструкций.

Слова

Слова соответствуют определенным параметрам и значениям:

- Параметры конфигурации блока: некоторые параметры доступны программе (например, предварительно выбранные параметры), некоторые недоступны программе (например, временная ось).
- Текущие значения: например, %Ci.V, текущее значение счетчика.

Объекты, доступные программе

Список объектов, доступных программе, см. в соответствующих разделах.

- Для основных функциональных блоков, см. *Основные функциональные блоки, стр. 275*.
- Для дополнительных функциональных блоков, см. *Битовые объекты и слова, связанные с дополнительными функциональными блоками, стр. 328*.

Структурированные объекты

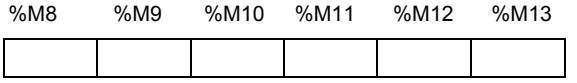
Введение

Структурированные объекты являются комбинацией смежных объектов. Twido поддерживает следующие типы структурированных объектов:

- Битовые строки
- Таблицы слов
- Таблицы двойных слов
- Таблицы слов с плавающей точкой

Битовые строки

Битовые строки являются последовательностью смежных битовых объектов одного типа определенной длины (L).
Пример: Битовая строка %M8:6



Примечание: %M8:6 допустима (8 кратно 8), в то время как %M10:16 не допустима (10 не кратно 8).

Битовые строки могут использоваться в инструкциях присваивания (см. *Инструкции присваивания, стр. 300*).

Допустимые типы битов

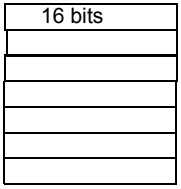
Допустимые типы битов для битовых строк:

Тип	Адрес	Максимальный размер	Доступ к режиму записи
Отдельные биты входов	%I0.0:L или %I1.0:L (1)	0<L<17	Нет
Отдельные биты выходов	%Q0.0:L или %Q1.0:L (1)	0<L<17	Да
Системные биты	%Si:L при i кратном 8	0<L<17 и i+LJ 128	Зависит от i
Биты шагов Grafcet	%Xi:L при i кратном 8	0<L<17 и i+LJ 95 (2)	Да (программно)
Внутренние биты	%Mi:L при i кратном 8	0<L<17 и i+LJ 256 (3)	Да

Пояснения:
1. Биты вх/вых можно считать в битовую строку только с 0 по 16. Для контроллеров с 24 входами и модулей вх/вых 32, биты больше 16 не могут быть считаны в битовую строку.
2. Максимумом для i+L для TWWDLCAA10DRF и TWDLCAA16DRF является 62
3. Максимумом для i+L для TWWDLCAA10DRF и TWDLCAA16DRF является 128.

Таблицы слов

Таблицы слов являются последовательностью смежных слов одного типа определенной длины (L).
Пример: Таблица слов %KW10:7



Таблицы слов можно использовать в инструкциях присваивания (см. *Инструкции присваивания, стр. 300*).

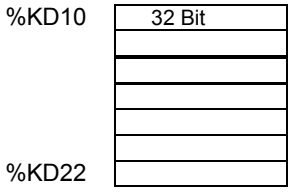
Допустимые типы слов

Допустимые типы слов для таблиц слов:

Тип	Адрес	Максимальный размер	Доступ к режиму записи
Внутренние слова	%MWi:L	0<L<256 и i+L< 3000	Да
Слова-константы	%KWi:L	0<L<256 и i+L< 256	Нет
Системные слова	%SWi:L	0<L и i+L<128	Зависит от i

Таблицы двойных слов

Таблицы двойных слов являются последовательностью смежных слов одного типа определенной длины (L).
Пример: Таблица двойных слов %KD10:7



Таблицы двойных слов можно использовать в инструкциях присваивания (см. *Инструкции присваивания, стр. 300*).

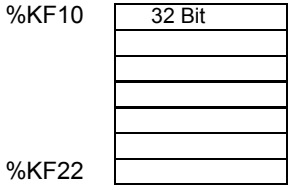
Допустимые типы двойных слов

Допустимые типы слов для таблиц двойных слов:

Тип	Адрес	Максимальный размер	Доступ к режиму записи
Внутренние слова	%MDi:L	0<L<256 и i+L< 3000	Да
Слова-константы	%KDi:L	0<L и i+L< 256	Нет

Таблицы слов с плавающей точкой

Таблицы слов с плавающей точкой являются последовательностью смежных слов одного типа определенной длины (L).
Пример: Таблица слов с плавающей точкой %KF10:7



Таблицы слов с плавающей точкой можно использовать в инструкциях присваивания (см. Дополнительные инструкции).

Допустимые типы слов с плавающей точкой

Допустимые типы слов с плавающей точкой для таблиц слов с ПТ:

Тип	Адрес	Максимальный размер	Доступ к режиму записи
Внутренние слова	%MFi:L	0<L<256 и i+L< 3000	Да
Слова-константы	%KFi:L	0<L и i+L<256	Нет

Индексированные объекты

Введение

Индексированное слово - это одинарное или двойное слово или слово с плавающей точкой с индексным адресом. Существует два типа адресации объектов:

- Прямая адресация
- Индексная адресация

Прямая адресация

Прямой адрес объекта устанавливается и определяется при написании программы.
Пример: %M26 это внутренний бит с прямым адресом 26.

Индексная адресация

Индексный адрес объекта предоставляет метод модификации адреса объекта путем добавления индекса к прямому адресу объекта. Содержимое индекса добавляется к прямому адресу объекта. Индекс определяется внутренним словом %MWi. Число "индексных слов" не ограничено.
Пример: %MW108[%MW2] это слово с адресом, состоящим из прямого адреса 108, плюс содержимое слова %MW2.
Если слово %MW2 имеет значение 12, запись в %MW108[%MW2] эквивалентна записи в %MW120 (108 плюс 12).

Объекты, доступные для индексной адресации

Допустимые типы объектов для индексной адресации.

Тип	Адрес	Максимальный размер	Доступ к режиму записи
Внутренние слова	%MWi[MWj]	0J i+%MWj<3000	Да
Слова-константы	%KWi[%MWj]	0J i+%MWj<256	Нет
Внутренние двойные слова	%MDi[MWj]	0J i+%MWj<2999	Да
Двойные слова-константы	%KDi[%MWj]	0J i+%MWj<255	Нет
Внутренние слова с ПТ	%MFi[MWj]	0J i+%MWj<2999	Да
Константы с ПТ	%KFi[%MWj]	0J i+%MWj<255	Нет

Индексированные объекты можно использовать в инструкциях присваивания (см. *Инструкции присваивания, стр. 300* для одинарных и двойных слов) и в инструкциях сравнения (см. *Инструкции сравнения, стр. 305* для одинарных и двойных слов). Этот тип адресации позволяет успешно сканировать последовательность объектов одного типа (например, внутренние слова или константы) при помощи изменения содержимого индексного объекта в программе.

Переполнение индекса, системный бит %S20

Переполнение индекса возникает в случаях, когда адрес индексированного объекта выходит за пределы зоны памяти, включающей тот же тип объекта, т.е. когда:

- Адрес объекта + содержимое индекса меньше нуля.
- Адрес объекта + содержимое индекса больше, чем максимальный адрес слова, к которому есть прямое обращение в приложении. Максимальное число 2999 (для слов %MWi) или 255 (для слов %KWi).

Если имеет место событие переполнения индекса, система устанавливает системный бит %S20 в 1, а объекту присваивается значение индекса равное 0.

Примечание: Отслеживание переполнения - задача пользователя: бит %S20 должен считываться пользовательской программой для дальнейшей обработки. Возвращение его в исходное состояние также лежит на пользователе.

%S20 (первоначальное состояние = 0):

- При переполнении индекса: устанавливается в 1 системой.
- После уведомления о переполнении: устанавливается в 0 пользователем после модификации индекса.

Символизированные объекты

Введение

Символы можно использовать для адресации объектов языка Twido по имени или подходящим мнемоникам. Использование символов позволяет быстро просмотреть и проанализировать логику программы и значительно упрощает разработку и тестирование приложения.

Пример

Например, WASH_END это символ, который может использоваться для идентификации функционального блока таймера, который отображает конец цикла стирки. Проще восстановить смысл этого названия, чем пытаться вспомнить роль адреса программы %TM3.

Указания по определению символов

Ниже приведены указания по определению символов:

- Максимум 32 знака.
- Буквы (A-Z), цифры (0 -9), символ подчеркивания (_).
- Первый знак должен быть буквой или символом подчеркивания. Нельзя использовать знак процента (%).
- Не используйте пробелы и специальные символы.
- Регистр не учитывается. Например, Pump1 и PUMP1 это один и тот же символ и может использоваться в приложении только один раз.

Редактирование символов

Символы определяются и ассоциируются с языковыми объектами в Редакторе Символов. Символы и комментарии к ним хранятся вместе с приложением на жестком диске, но не в контроллере. Поэтому их нельзя переслать вместе с приложением в контроллер.

Пользовательская память

3

Обзор

Предмет

В этой главе описаны структура и использование пользовательской памяти Twido.

Содержание главы

Эта глава содержит следующие темы:

Структура пользовательской памяти

Введение

Память контроллера, доступная приложению, делится на два отдельных блока:
- Биты
- Слова (16-битные значения со знаком)

Битовая память

Битовая память находится в оперативной памяти, интегрированной в контроллер, и содержит изображение 128 битовых объектов.

Память слов

Память слов (16 бит) обеспечивает:
- **Динамические слова**: рабочая память (хранятся только в ОЗУ).
- **Слова памяти (%MW)**: динамические системные данные и системные данные.
- **Программу**: дескрипторы и выполняемый код задач.
- **Конфигурационные данные**: слова-константы, исходные значения и конфигурация входа/выхода.

Типы памяти

Ниже представлены различные типы памяти контроллеров Twido.
- Оперативное запоминающее устройство(RAM).
  Внутренняя временная память: содержит динамические слова, слова памяти, программу и конфигурационные данные.
- Электронно-перепрограммируемая постоянная память (EEPROM)
  Встроенная 32KB EEPROM, которая обеспечивает резервное хранение внутренней программы и данных. Защищает программу от сбоев из-за повреждения батареи или отсутствия питания более 30 дней. Хранит программу и конфигурационные данные. Содержит максимум 512 слов памяти. Программа не сохраняется в этой памяти, если используется картридж расширения памяти 64K и Twido сконфигурирован для принятия картриджа расширения памяти 64K.
- Стираемый картридж резервного хранения 32K
  Опциональный внешний картридж, используемый для сохранения программы и переноса ее на другие контроллеры Twido. Может использоваться для обновления программы в ОЗУ контроллера. Содержит программу и константы, но никаких слов памяти.
- Картридж расширения памяти 64K.
  Опциональный внешний картридж, который хранит программу размером до 64K. Должен оставаться вставленным в контроллер все время использования программы.

Сохранение памяти

Ваша программа и слова памяти могут храниться:
- RAM (до 30 дней при хорошей батарее)

- EEPROM (максимум 32 Кб)

Перенос программы из памяти EEPROM в ОЗУ производится автоматически, когда программа потеряна в ОЗУ (или если нет батареи).
Может быть также осуществлен ручной перенос при помощи TwidoSoft.

Конфигурации памяти

В следующей таблице описаны возможные типы конфигураций памяти контроллеров Twido.

Тип памяти	**Компактные контроллеры**			**Модульные контроллеры**		
	10DRF	**16DRF**	**24DRF**	**20DUK 20DTK**	**20DRT 40DUK 40DTK (32k)**	**20DRT 40DUK 40DTK** (64k)**
Внутреннее ОЗУ Mem 1*	10Кб	10Кб	10Кб	10Кб	10Кб	10Кб
Внешнее ОЗУ Mem 2*		16Кб	32Кб	32Кб	32Кб	64Кб
Внутренняя EEPROM	8Кб	16Кб	32Кб	32Кб	32Кб	32Кб***
Внешняя EEPROM	32Кб	32Кб	32Кб	32Кб	32Кб	64Кб
Максимальный размер программы	8Кб	16Кб	32Кб	32Кб	32Кб	64Кб
Максимальный размер внешней резервной копии	8Кб	16Кб	32Кб	32Кб	32Кб	64Кб

(*) Mem 1 и Mem 2 - использование памяти.
(**)в этом случае картридж 64KB должен быть установлен в Twido и объявлен в конфигурации, если до этого не был объявлен,
(***) зарезервировано для резервного копирования слов %MW.

Резервное хранение и восстановление без использования картриджа резервного хранения или расширенной памяти

Введение

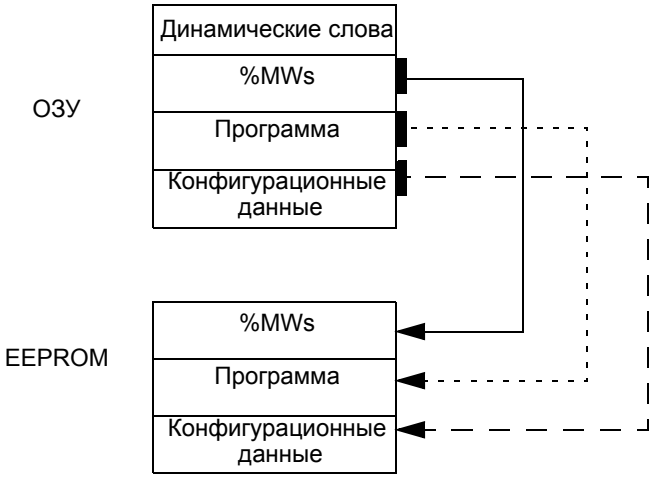
Далее подробно описываются функции резервного хранения и восстановления памяти в модульных и компактных контроллерах без использования картриджа резервного хранения или установленной расширенной памяти.

Обзор

Программы Twido, слова памяти или конфигурационные данные могут быть сохранены, используя внутреннюю память EEPROM. Сохранение программы во внутреннюю EEPROM стирает все предварительно сохраненные слова памяти, поэтому программа должна сохранятся первой, а затем слова памяти. Динамические данные могут храниться в словах памяти, а затем сохраняться в EEPROM. Если во внутренней памяти EEPROM нет сохраненной программы, то Вы не можете сохранить в нее слова памяти.

Структура памяти

Ниже приведена структура памяти контроллера. Стрелки показывают, что может быть сохранено в EEPROM из ОЗУ:



Резервное сохранение программы

Шаги для резервного сохранения программы в EEPROM:

Шаг	Действие
1	Следующее должно быть истинно: В ОЗУ есть допустимая программа.

Шаг	Действие
2	В окне TwidoSoft раскройте меню ‘Controller’, прокрутите до пункта ‘Backup’ и кликните на нем.

Восстановление программы

Есть только один путь восстановления программ в в ОЗУ из EEPROM во время запуска (если не установлено картриджа или памяти расширения):

- Программа в ОЗУ не допустима

Чтобы восстановить программу вручную из EEPROM, сделайте следующее:

- В окне TwidoSoft раскройте меню ‘Controller’, прокрутите до пункта ‘Restore’ и кликните на нем.

Резервное сохранение данных (%MWs)

Шаги для резервного сохранения данных (слов памяти) в EEPROM:

Шаг	Действие
1	Для того, чтобы это сработало, следующее должно быть истинным: В ОЗУ есть допустимая программа (%SW96:X6=1). Та же программа уже сохранена в EEPROM. слова .
2	Запишите в %SW97 длину сохраняемых слов памяти. **Примечание:** длина не может превышать сконфигурированную длину области слов памяти, она должна быть больше 0, но не больше 512.
3	Установите%SW96:X0 в 1.

Восстановление данных (%MWs)

Восстановите %MWs вручную установкой системного бита %S95 в 1.
Для того, чтобы это сработало, следующее должно быть истинным:

- В EEPROM есть пригодное резервное приложение.
- Приложение в ОЗУ совпадает с резервным приложением в EEPROM.
- Резервные слова памяти действительны.

Резервное хранение и восстановление при помощи картриджа 32K

Введение

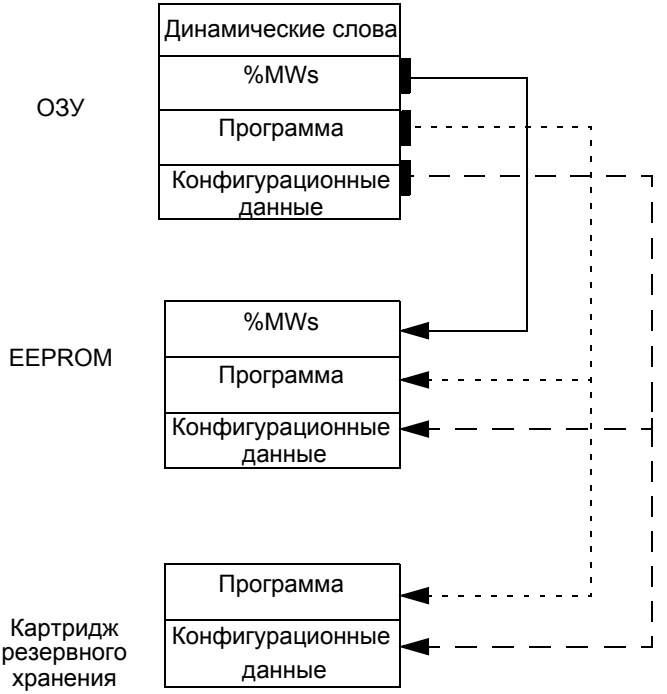
Далее подробно описываются функции резервного хранения и восстановления памяти в модульных и компактных контроллерах при помощи картриджа резервного хранения 32K.

Обзор

Картридж резервного хранения используется, чтобы сохранить программу и перенести ее на другие контроллеры Twido. Он должен быть удален из контроллера и отложен после того, как программа была установлена или сохранена. На картридж можно сохранить только программу и конфигурационные данные (%MWs не могут быть сохранены на картридж 32K). Динамические данные могут храниться в словах памяти и затем сохраняться в EEPROM. После завершения установки программы любые слова %MWs, которые были сохранены во внутренней EEPROM до установки, будут утеряны.

Структура памяти

Ниже приведена структура памяти контроллера с присоединенным картриджем резервного хранения. Стрелки показывают, что может быть сохранено в EEPROM и картридж из ОЗУ.



Резервное сохранение программы

Шаги для сохранения Вашей программы на картридж резервного хранения:

Шаг	Действие
1	Отключите питание контроллера.
2	Подсоедините картридж.
3	Включите питание контроллера.
4	В окне TwidoSoft раскройте меню ‘Controller’, прокрутите до пункта ‘Backup’ и кликните на нем.
5	Отключите питание контроллера.
6	Удалите картридж из контроллера.

Восстановление программы

Для загрузки программы из картриджа в контроллер выполните следующее:

Шаг	Действие
1	Отключите питание контроллера.
2	Подсоедините картридж.
3	Включите питание контроллера. (Если сконфигурирована автозагрузка, Вы должны повторить отключение/включение питания, чтобы перейти в режим выполнения).
4	Отключите питание контроллера.
5	Удалите картридж из контроллера.

Резервное сохранение данных (%MWs)

Шаги для сохранения данных (слов памяти) в память EEPROM:

Шаг	Действие
1	Для того, чтобы это сработало, следующее должно быть истинным: В ОЗУ есть пригодная программа. Та же программа уже сохранена в EEPROM. Слова памяти сконфигурированы в программе.
2	Запишите в %SW97 длину сохраняемых слов памяти. **Примечание:** длина не может превышать сконфигурированную длину области слов памяти, она должна быть больше 0, но не больше 512.
3	Установите %SW96:X0 в 1.

Восстановление данных (%MWs)

Восстановите %MWs вручную установкой системного бита %S95 в 1.
Для того, чтобы это сработало, следующее должно быть истинным:

- В EEPROM есть пригодное резервное приложение.
- Приложение в ОЗУ совпадает с резервным приложением в EEPROM.
- Резервные слова памяти допустимы.

Использование картриджа расширения памяти 64K

Введение

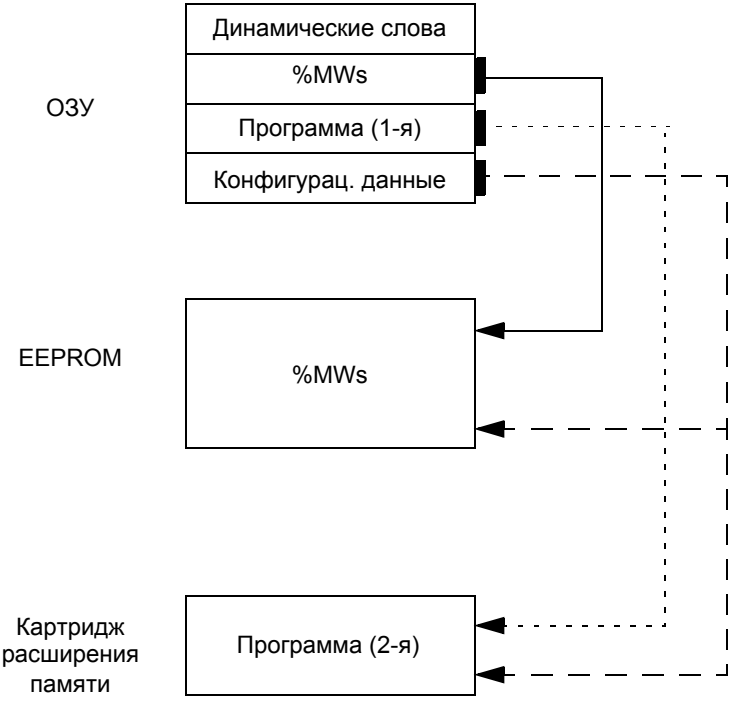
Далее подробно описываются функции памяти в модульных контроллерах при использовании картриджа расширения памяти 64K.

Обзор

Картридж расширения памяти 64K используется для увеличения памяти программы контроллера Twido с 32K до 64K. Он должен оставаться в контроллере во время использования программы. При удалении картриджа, контроллер войдет в состояние Остановка. Слова памяти так же, как и раньше, сохраняются в память EEPROM контроллера. Динамические данные могут храниться в словах памяти и затем сохраняться в память EEPROM. Картридж расширения памяти 64K ведет себя при включении так же, как и картридж резервного хранения 32K.

Структура памяти

Здесь приведена структура памяти при использовании картриджа расширения памяти. Стрелки показывают, что сохраняется в EEPROM и картридж расширения памяти 64K из ОЗУ:



Конфигурирование ПО и установка расширенной памяти

Перед записью расширенной программы Вы должны установить картридж расширения памяти в контроллер. Следующие четыре шага показывают, как это сделать:

Шаг	Действие
1	В меню аппаратных опций в окне TwidoSoft введите ‘TWDXCPMFK64’.
2	Отключите питание контроллера.
3	Подсоедините картридж расширения памяти 64K.
4	Включите питание контроллера.

Сохраните Вашу программу

Сохраните программу один раз после того, как она будет написана и картридж расширения памяти будет установлен:

- В окне TwidoSoft раскройте меню ‘Controller’, прокрутите до пункта ‘Backup’ и кликните на нем.

Резервное сохранение данных (%MWs)

Ниже приведены шаги для сохранения данных (слов памяти) в EEPROM:

Шаг	Действие
1	Для того, чтобы это сработало, следующее должно быть истинным: Есть пригодная программа. Слова памяти сконфигурированы в программе.
2	Запишите в %SW97 длину сохраняемых слов памяти. Примечание: длина не может превышать сконфигурированную длину области слов памяти, она должна быть больше 0, но не больше 512.
3	Установите %SW96:X0 в 1.

Восстановление данных (%MWs)

Восстановите %MWs вручную установкой системного бита %S95 в 1.
Для того, чтобы это сработало, следующее должно быть истинным:

- Есть пригодная программа.
- Резервные слова памяти действительны.

Режимы работы контроллера

4

Обзор

Предмет

В этой главе описаны режимы работы контроллера и циклическое и периодическое выполнение программы. Содержится подробная информация об отключении и включении питания .

Содержание главы

Эта глава содержит следующие темы:

Циклическое выполнение

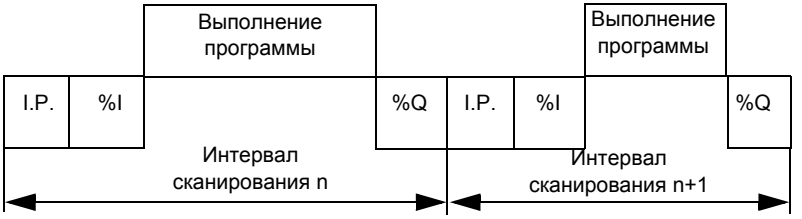
Введение

При циклическом выполнении циклы контроллера последовательно соединяются вместе один за другим. После обновления выходов (третья фаза цикла задачи) система выполняет свои определенные операции и затем присоединяет следующий цикл задачи.

Примечание: Время сканирования пользовательской программы контролируется сторожевым таймером контроллера и не должно превышать 500 мс. В противном случае появляется ошибка, вызывающая немедленный переход контроллера в режим ОСТАНОВ. Выходы в этом режиме устанавливаются в аварийное состояние.

Работа

На следующем рисунке показаны фазы интервала сканирования при циклическом выполнении.



Описание фаз цикла

В следующей таблице описаны фазы цикла.

Адрес	Фаза	Описание
I.P.	Внутренняя обработка	Система полностью контролирует контроллер (управление системными битами и словами, обновление текущих значений таймера, обновление индикаторов статуса, определение изменения состояния РАБОТА/ОСТАНОВКА (RUN/STOP) и т.п.) и обрабатывает запросы от TwidoSoft (изменения и анимация).
%I, %IW	Чтение входов	Записывает в память состояние входов дискретных и определенных приложением модулей.
-	Обработка программы	Выполняет написанную пользователем программу приложения.
%Q, %QW	Обновление выходов	Записывает выходные биты или слова, связанные с дискретными и определяемыми приложением модулями.

Режимы работы

Контроллер в режиме РАБОТА, процессор выполняет:

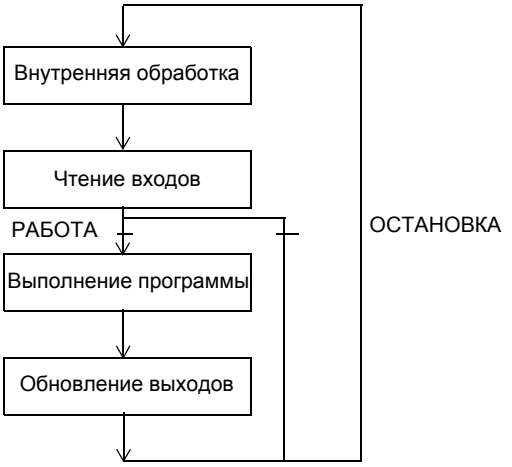
- Внутреннюю обработку
- Чтение входов
- Выполнение программы приложения
- Обновление выходов

Контроллер в режиме ОСТАНОВКА, процессор выполняет:

- Внутреннюю обработку
- Чтение входов

Иллюстрация

На следующем рисунке показаны рабочие циклы.



Проверка цикла

Сканирование контролируется сторожевым таймером.

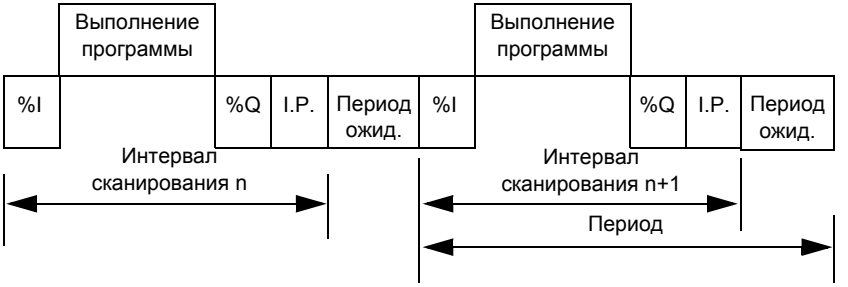
Периодическое выполнение

Введение

В этом режиме чтение входов, обработка программы приложения и обновление выходов выполняется периодически в соответствии с длительностью периода, определенного во время конфигурации (от 2 до 150 мс).
В начале сканирования таймер, чье текущее значение устанавливается равным длительности периода, определенного во время конфигурации, начинает считать на уменьшение. Сканирование должно закончиться прежде, чем таймер достигнет нулевого значения, которое в свою очередь запускает новое сканирование.

Работа

На следующем рисунке показаны фазы интервала сканирования при периодическом выполнении.



Описание фаз цикла

В следующей таблице описаны фазы цикла.

Адрес	Фаза	Описание
I.P.	Внутренняя обработка	Система полностью контролирует контроллер (управление системными битами и словами, обновление текущих значений таймера, обновление индикаторов статуса, определение изменения состояния РАБОТА/ОСТАНОВКА (RUN/STOP) и т.п.) и обрабатывает запросы от TwidoSoft (изменения и анимация).
%I, %IW	Чтение входов	Записывает в память состояние входов дискретных и определенных приложением модулей.
-	Обработка программы	Выполняет написанную пользователем программу приложения.
%Q, %QW	Обновление выходов	Записывает выходные биты или слова, связанные с дискретными и определяемыми приложением модулями.

Режимы работы

Контроллер в режиме РАБОТА, процессор выполняет:

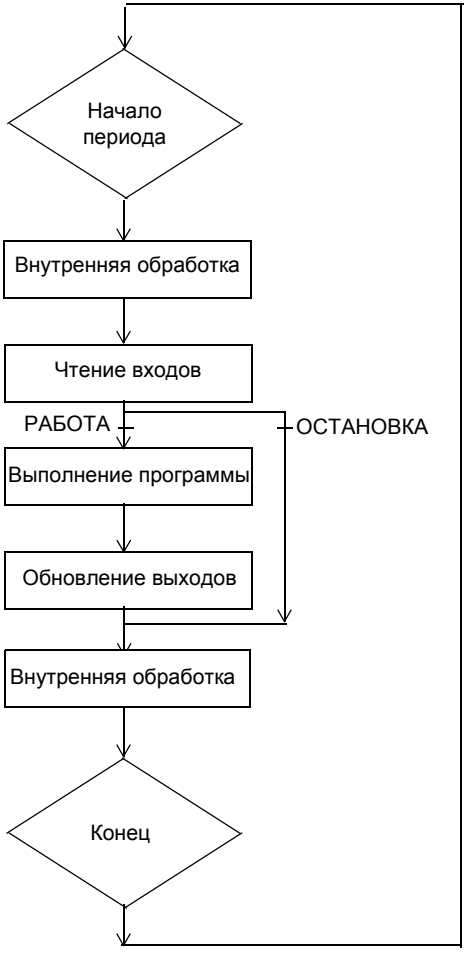
- Внутреннюю обработку
- Чтение входов
- Выполнение программы приложения
- Обновление выходов

Если период еще не закончен, процессор выполняет свой рабочий цикл до конца периода внутренней обработки. Если время выполнения операций становится более длительным, чем выделено для периода, контроллер указывает на то, что период был превышен, установкой системного бита %S19 в 1. Процесс продолжается до полного завершения. Он, однако, не должен превышать временной предел сторожевого таймера. Следующие сканирование подсоединяется после полной записи выходов в текущем сканировании.

Контроллер в режиме ОСТАНОВКА, процессор выполняет:

- Внутреннюю обработку
- Чтение входов

Иллюстрация На следующем рисунке показаны рабочие циклы.



Проверка цикла Выполняется две проверки:

- Контроль переполнение периода
- При помощи сторожевого таймера

Контроль времени сканирования

Введение

Время выполнения задачи контролируется сторожевым таймером Tmax (максимальная длительность цикла задачи). Он позволяет выявить ошибки приложения (бесконечные циклы и т. д.) и гарантирует максимальную продолжительность обновления выходов.

Программный сторожевой таймер (периодическое или циклическое выполнение)

При периодическом или циклическом выполнении срабатывание сторожевого таймера приводит к программной ошибке. Приложение переходит в режим ОСТАНОВ и устанавливает системный бит %S11 в 1. Перед перезапуском задачи необходимо при помощи Twido Soft проанализировать причины ошибки, исправить ошибку в приложении и вернуть программу в режим РАБОТА.

Примечание: Состояние ОСТАНОВ - это состояние, при котором выполнение приложения немедленно прекращается из-за программной ошибки, такой как переполнение при сканировании. Текущие значения данных сохраняются, что позволяет проанализировать причину ошибки. Программа останавливается на текущей инструкции. Связь с контроллером возможна.

Проверка при периодическом выполнении

При периодическом выполнении используется дополнительная проверка для обнаружения переполнения периода:

- **%S19** показывает переполнение периода. Он устанавливается:
  - в 1 системой, когда время сканирования больше периода задачи,
  - в 0 пользователем.
- **%SW0** содержит значение периода (0-150 мс). Он:
  - Оно инициализируется при "холодном" пуске значением, определенным во время конфигурации,
  - Может быть изменено пользователем.

Использование времени выполнения мастер-задачи

Системные слова, предоставляющие информацию о времени сканирования:

- **%SW11** содержит максимальное время сторожевого таймера (10 - 500 мс).
- **%SW30** содержит время выполнения последнего сканирования.
- **%SW31** содержит время выполнения самого продолжительного сканирования с момента последнего “холодного” пуска.
- **%SW32** содержит время выполнения самого короткого сканирования с момента последнего “холодного” пуска.

Примечание: Эта информация также может быть получена из редактора конфигураций.

Режимы работы

Введение

Twido Soft принимает в расчет три основные группы режимов работы:
- Проверка
- Работа или производство
- Остановка

Запуск при помощи Grafcet

Эти режимы работы можно достичь либо начиная с, либо используя следующие методы Grafcet:
- Инициализация Grafcet
- Предустановка шагов
- Поддержание ситуации
- “Замораживание” диаграмм

Предварительная обработка и использование системных битов обеспечивают эффективное управление рабочим режимом без усложнения и перегрузки программы пользователя.

Системные биты Grafcet

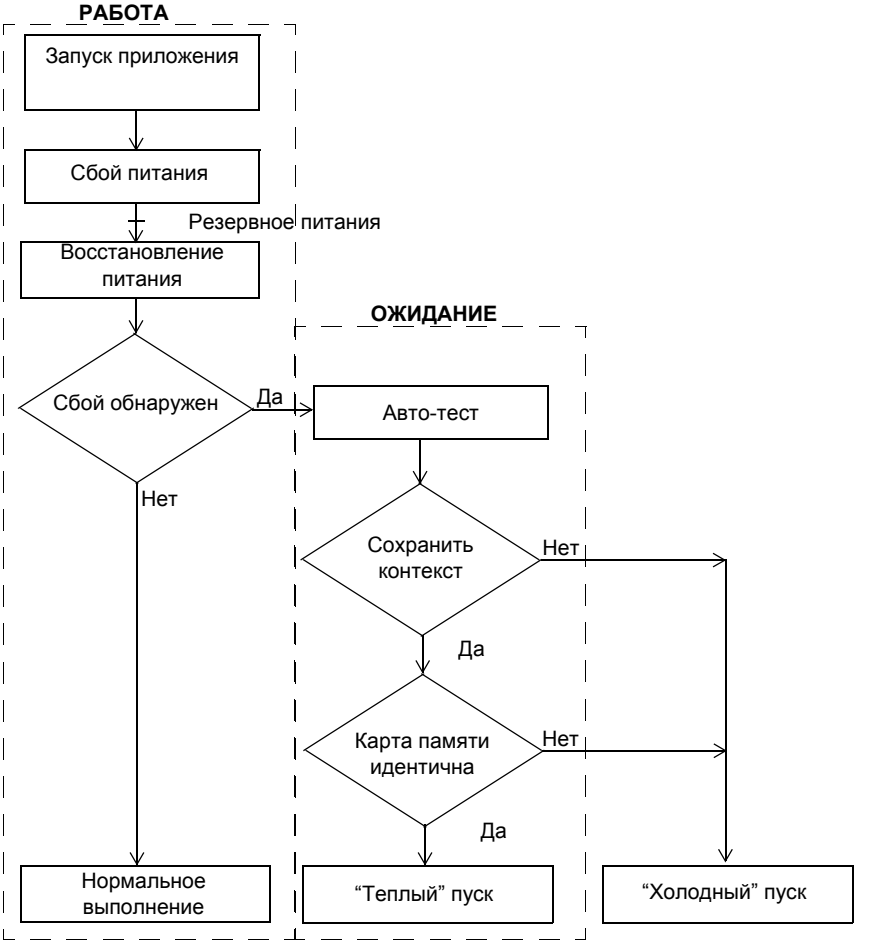
Биты %S21, %S22 и %S23 зарезервированы для предварительной обработки. Эти биты автоматически сбрасываются системой. Они должны записываться только при помощи инструкции установки **S**.
В следующей таблице представлены связанные с Grafcet системные биты:

Бит	Функция	Описание
%S21	Инициализация GRAFCET	Обычно установлен в 0, устанавливается в 1: ● При “холодном” пуске, **%S0=1**; ● Пользователем в части предобработки при помощи инструкции **S %S21** или катушки **-(S)- %S21**. Результат: ● Дезактивация всех активных шагов. ● Активация всех начальных шагов.
%S22	Сброс GRAFCET	Обычно установлен в 0, может быть установлен в 1 только программой при предварительной обработке. Результат: ● Дезактивация всех активных шагов. ● Останавливается выполнение последовательной обработки.
%S23	Предустановка и замораживание GRAFCET	Обычно = 0, можно уст. в 1 только программой при предобработке. ● Предустановка установкой бита %S22 в 1. ● Предустановка шагов, активируемых набором инструкций S Xi. ● Разрешение предустановки установкой бита %S23 в 1. Замораживание состояния: ● В начальной ситуации: поддерживанием бита %S21 в 1 программой. ● В "пустой" ситуации: поддерживанием бита %S22 в 1 прогр. ● В ситуации, обусловленной поддерживанием бита %S23 в 1.

Поведение при отключении и включении питания

Иллюстрация

На следующем рисунке показаны различные перезапуски питания, определяемые системой. Если время отсутствия питания меньше, чем время фильтрации (около 10 мс для источника питания переменного тока или 1 мс для источника питания постоянного тока), то сбой не замечается программой, которая продолжает нормально выполняться.



Примечание: Контекст сохраняется в резервной оперативной памяти батареи . При запуске система проверяет состояние батареи и сохраненный контекст, чтобы решить, возможен ли “теплый” пуск.

Сравнение бита входа РАБОТА/ ОСТАНОВКА и опции автоматического запуска

У бита входа РАБОТА/ОСТАНОВКА приоритет больше, чем у опции "Автоматический запуск при старте", которая доступна из диалогового окна “Режим Сканирования” (Scan Mode). Если установлен бит РАБОТА/ОСТАНОВКА , тогда контроллер запустится в режиме РАБОТА при восстановлении питания. Режим контроллера определяется следующим образом:

Бит РАБОТА/ОСТАНОВКА	Автоматический запуск при старте	Результат
Ноль	Ноль	ОСТАНОВКА
Ноль	Единица	ОСТАНОВКА
Фронт сигнала	Не имеет значения	РАБОТА
Единица	Не имеет значения	РАБОТА
Не сконфигурирован	Ноль	ОСТАНОВКА
Не сконфигурирован	Единица	РАБОТА

Примечание: Для всех компактных контроллеров с ПО версии 1.0 действительно следующее: если в момент сбоя питания контроллер был в режиме РАБОТА, и флаг “Автоматический запуск при старте” не был установлен из диалогового окна Scan Mode, контроллер перезагрузится в режиме ОСТАНОВКА. В остальных случаях будет произведен “холодный” перезапуск.

Примечание: Для всех модульных и компактных контроллеров с ПО версии 1.11 действительно следующее: если батарея контроллера нормально работала при сбое питания, контроллер перезагрузится в том режиме, который был в момент сбоя. Флаг “Автоматический запуск при старте” не повлияет на режим при восстановлении питания.

Работа

В следующей таблице описаны фазы работы при сбое питания.

Фаза	Описание
1	При отключении питания система сохраняет контекст приложения и время сбоя.
2	Все выходы устанавливаются в аварийное состояние (0).
3	При восстановлении питания сохраненный контекст сравнивается с текущим, что определяет тип пуска: ● Если контекст приложения изменился (потеря системного контекста или запуск нового приложения), контроллер инициализирует приложение: “холодный” перезапуск (обычно для компактных контроллеров). ● Если контекст приложения не изменился, контроллер перезагружается без инициализации данных: “теплый” перезапуск.

Поведение при “теплом” перезапуске

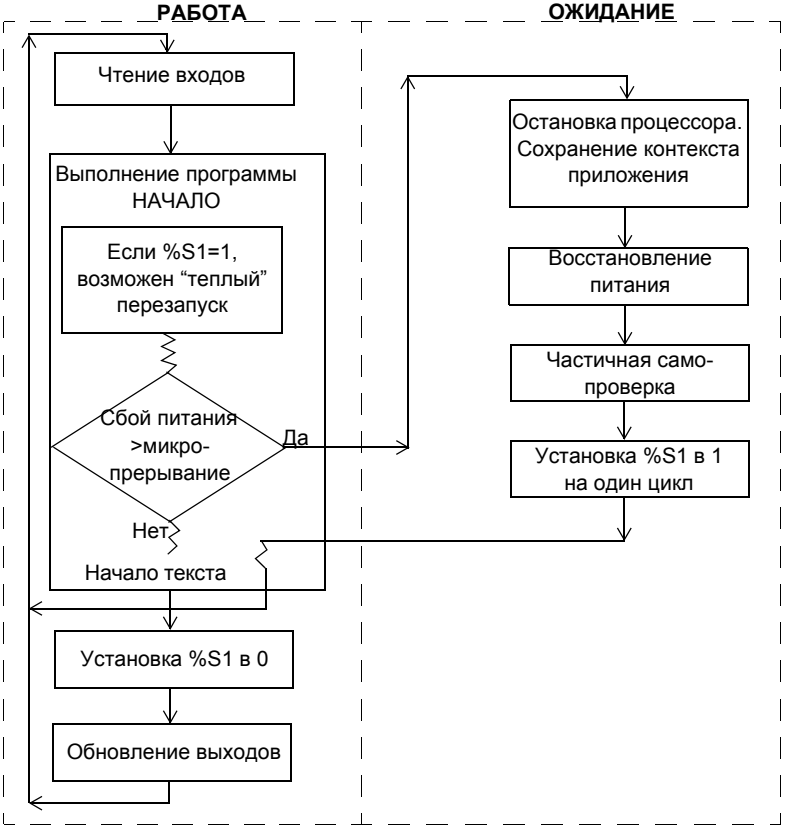
Причины “теплого” перезапуска

“Теплый” перезапуск может произойти:

- Когда питание восстановлено без потери контекста приложения,
- Когда бит **%S1** установлен в 1 программой,
- Из операторского дисплея, когда контроллер находится в режиме ОСТАНОВКА

Иллюстрация

На следующем рисунке описан процесс “теплого” перезапуска в режиме РАБОТА.



Восстановление выполнения программы

В следующей таблице описываются фазы запуска программы после “теплого” перезапуска.

Фаза	Описание
1	Выполнение программы перезапускается с элемента, на котором происходит отключение питания, без обновления выходов. Примечание: перезапускается только код пользователя. Системный код (например, для обновления выходов) не запускается.
2	В конце цикла восстановления система: ● Освобождает приложение, если оно было зарезервировано (и вызывает приложение ОСТАНОВКА в случае отладки) ● Повторно инициализирует сообщения.
3	Система выполняет цикл перезагрузки, в котором она: ● Запускает задания, устанавливая **%S1** (индикатор “теплого” пуска) и **%S13** (первый цикл при РАБОТЕ) в 1 ● Сбрасывает биты **%S1** и **%S13** в 0 в конце первого цикла задания.

Процедура “теплого” пуска

В случае “теплого” пуска, если требуется особый процесс приложения, должен быть проверен бит **%S1** в начале цикла задачи и вызвана соответствующая задача.

Состояние выходов после отключения питания

Если отсутствие питания определено, выходы устанавливаются (по умолчанию) в аварийное состояние 0.
При восстановлении питания выходы находятся в последнем состоянии, пока не будут обновлены программой.

Поведение при “холодном” запуске

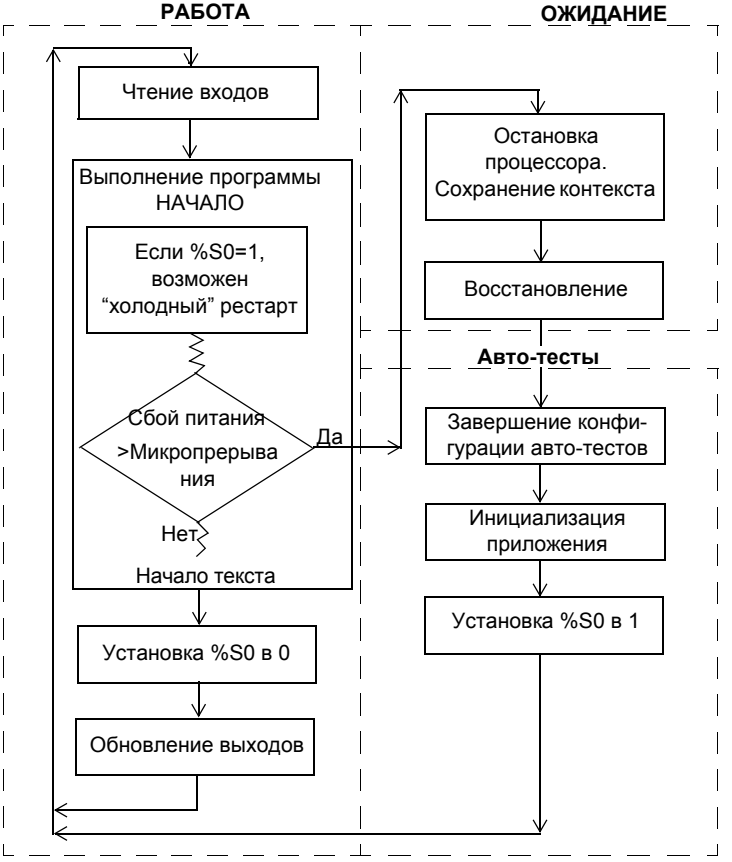
Причины “холодного” запуска

“Холодный” запуск может произойти:

- При загрузке нового приложения в ОЗУ,
- Когда питание восстановлено с потерей контекста приложения,
- Когда бит **%S0** установлен в 1 программой,
- Из операторского дисплея, когда контроллер находится в режиме ОСТАНОВКА

Иллюстрация

На следующем рисунке описана операция “холодного” перезапуска в режиме РАБОТА.



Работа

В следующей таблице описываются фазы запуска программы после “холодного” перезапуска.

Фаза	Описание
1	При запуске контроллер находится в режиме РАБОТА. При “холодном” перезапуске после остановки из-за ошибки система вмешивается в процесс “холодного” перезапуска. Выполнение программы начинается с начала цикла.
2	Система: ● Сбрасывает внутренние биты и слова и отображения вх/вых в 0 ● Инициализирует системные биты и слова ● Инициализирует функциональные блоки из конфигурационных данных
3	Для первого цикла после перезапуска система: ● Запускает задания, устанавливая **%S0** (индикатор “холодного” пуска) и **%S13** (первый цикл при РАБОТЕ) в 1 ● Сбрасывает биты **%S0** и **%S13** в 0 в конце первого цикла задания ● Сбрасывает биты **%S31**, **%S38** и **%S39** (индикаторы контроля событий), и слова **%SW48** (число выполненных событий).

Процедура “холодного” пуска

В случае холодного пуска, если требуется выполнение отдельного приложения, бит **%S0** (который =1) должен быть проверен во время первого цикла задания.

Состояние выходов после отключения питания

Если отсутствие питания определено, выходы устанавливаются (по умолчанию) в аварийное состояние 0.
При восстановлении питания, выходы находятся в состоянии 0, пока не будут обновлены задачей.

Инициализация объектов

Введение

Контроллеры могут инициализироваться при помощи Twido Soft установкой системных битов **%S0** (“холодный” перезапуск) и **%S1** (“теплый” перезапуск).

Инициализация “холодного” пуска

Для инициализации “холодного” перезапуска системный бит **%S0** должен быть установлен в 1.

Инициализация объектов (аналогично “холодному” пуску) при подключении питания, используя %S0 и %S1

Для инициализации объектов при включении питания системные биты **%S1** и **%S0** должны быть установлены 1.

Следующий пример показывает, как запрограммировать инициализацию объектов при “теплом” перезапуске, используя системные биты.


```
LD  %S1   Если%S1 = 1 (“теплый” перезапуск), установить %S0 в 1.
ST  %S0   Эти два бита сбрасываются системой в 0 в конце текущего прохода.
ST  %S9   Этот бит используется для инициализации выходов.
```

Примечание: Не устанавливайте %S0 в 1больше чем на один интервал сканирования контроллера.

Управление задачами, запускаемыми по событию

5

Вкратце...

Обзор

В этой главе описаны задачи, запускаемые по событию, и их выполнение контроллером.

Примечание: Этими задачами не может управлять контроллер TWDLCAA10DRF.

Содержание главы

Эта глава содержит следующие темы:

Обзор задач, запускаемых по событию

Введение

В предыдущей главе были представлены периодические (См. *Периодическое выполнение, стр. 62*) и циклические (См. *Циклическое выполнение, стр. 60*) задачи, в которых объекты обновляются в начале и в конце задачи. Источники событий могут вызвать останов определенных задач для выполнения более приоритетных задач, запускаемых по событию, что позволяет быстрее обновлять объекты.

Задача, запускаемая по событию:

- является частью программы, выполняемой в момент выполнения условия (источник события),
- имеет больший приоритет, чем основная программа,
- гарантирует быстрое время отклика, позволяя уменьшить общее время отклика системы.

Описание события

Событие состоит из:

- источника события, которое может быть определено, как программное или аппаратное условие, прерывающее основную программу (См. *Описание различных источников событий, стр. 77*),
- независимого программируемого объекта, относящегося к событию,
- очереди событий, которая может использоваться для хранения списка событий до их выполнения,
- уровня приоритета, который определяет порядок обработки событий.

Описание различных источников событий

Обзор различных источников событий

Необходимо использовать программное обеспечение для обработки источников событий, чтобы убедиться, что основная программа была корректно прервана событием, и чтобы вызвать раздел программы, связанный с событием. Время сканирования приложения не влияет на выполнение событий.
Разрешены следующие 9 источников событий:

- 4 условия, связанные с порогами функционального блока VFC (2 события на %VFC),
- 4 условия, связанные с физическими входами базы,
- 1 периодическое условие.

Источник события может связываться только с одним событием, и должен сразу определяться TwidoSoft. После определения источника программное обеспечение выполняет раздел программы, связанный с событием: каждое событие связано с подпрограммой с меткой **SRi:** определенной при конфигурации источников событий.

События, связанные с физическими входами базового контроллера

Входы %I0.2, %I0.3, %I0.4 и %I0.5 могут использоваться в качестве источников событий, если они не заперты и если события разрешены при конфигурации. Обработка событий может вызываться входами (со 2 по 5) базового контроллера (позиция 0) по фронту и спаду сигнала.
Для получения подробной информации по конфигурированию этого события обратитесь к разделу "Аппаратная конфигурация -> Конфигурация входов" ("Hardware Configuration -> Input Configuration") в on-line справочнике "Использование TwidoSoft" ("TwidoSoft Operation Guide").

Выходное событие функциональ-ного блока %VFC

Выходы TH0 и TH1 функционального блока %VFC являются источниками событий. Выходы TH0 и TH1 установлены:

- в 1, когда напряжение больше порогового S0 и порогового S1, соответственно,
- в 0, когда напряжение меньше порогового S0 и порогового S1, соответственно.

Обработка событий может активироваться по фронту и спаду сигналов на этих выходах.
Для получения подробной информации по конфигурированию этого события обратитесь к разделу "Программная конфигурация -> Очень быстрые счетчики" ("Software Configuration -> Very Fast Counters") в on-line справочнике "Использование TwidoSoft" ("TwidoSoft Operation Guide").

Периодическое событие

Это событие периодически запускает выполнение одной программной секции. У этой задачи приоритет больше, чем у основной задачи (master-задачи). Однако, у этого источника событий приоритет меньше, чем у других источников событий.
Периодичность этого задания устанавливается при конфигурации от 5 до 255 мс. Может использоваться только одно периодическое событие.
Для получения подробной информации по конфигурированию этого события обратитесь к разделу "Конфигурирование программных параметров -> Режим сканирования" ("Configuring Program Parameters -> Scan Mode") в on-line справочнике "Использование TwidoSoft" ("TwidoSoft Operation Guide").

Управление событиями

Очередь событий и приоритет событий

У событий есть 2 возможных приоритета: высокий и низкий. Только **один** тип события (один источник события) может иметь высокий приоритет, поэтому остальные события имеют низкий приоритет, и порядок их выполнения зависит от порядка, в котором они были обнаружены.
Существует две очереди задач для управления порядком выполнения задач, вызываемых событием:

- в одной может хранится до 16 событий с высоким приоритетом (от одного источника событий),
- в другой может хранится до 16 событий с низким приоритетом (от других источников событий).

Эти очереди работают по принципу FIFO: первое хранящееся событие будет обработано первым. Очереди могут хранить только 16 событий, поэтому все остальные события теряются.
События с низким приоритетом будут выполняться только тогда, когда очередь событий с высоким приоритетом будет пуста.

Управление очередями событий

Следующая последовательность выполняется каждый раз при появлении прерывания (связанного с источником события):

Шаг	Описание
1	Управление прерываниями: ● распознавание физического прерывания, ● событие сохраняется в подходящую очередь, ● проверка того, что нет незаконченного события того же приоритета (если такое событие есть, новое событие ожидает в очереди).
2	Сохранение контекста.
3	Выполнение программной секции (подпрограмма с меткой SRi:), связанной с событием.
4	Обновление выходов
5	Восстановление контекста

Перед восстановлением контекста должны быть выполнены все события в очереди.

Проверка событий

Для проверки событий используются системные биты и слова (См. *Системные биты и слова, стр. 433*):

- %S31: используется для выполнения или задержки события,
- %S38: используется для решения, поместить событие в очередь или нет,
- %S39: используется для выяснения, пропало ли событие или ожидает,

- %SW48: показывает, сколько событий обработано с момента последнего “холодного” пуска.

Значение битов и слов сбрасывается в ноль после “холодного” пуска или загрузки приложения, но не изменяется после “теплого” перезапуска. В обоих случаях очереди событий сбрасываются.

Специальные функции

II

Обзор

Тема этой части

В этой части описаны коммуникации, встроенные аналоговые функции, управление модулями аналоговых вх/вых и установка шины AS-Interface V2 для контроллеров Twido.

Содержание этой части

Эта часть содержит следующие главы:

Коммуникации

6

Обзор

Предмет

В этой главе представлен обзор конфигурирования, программирования и управления коммуникациями контроллеров Twido.

Содержание главы

Эта глава содержит следующие темы:

Представление различных типов коммуникаций

Обзор

Контроллеры Twido имеют один или два порта последовательной связи, которые используются для связи с удаленными контроллерами вх/вых, одноранговыми контроллерами или общими устройствами. Любой свободный порт может использоваться для всех видов связи, за исключением связи с Twido Soft, которая может производиться только через первый порт. На каждом контроллере Twido поддерживается три различных базовых протокола: дистанционная связь, ASCII и Modbus (modbus master или modbus slave).

Дистанционная связь

Протокол дистанционной связи является высокоскоростной шиной master/slave, предназначенной для передачи небольшого количества данных между Master контроллером и удаленными Slave контроллерами (до семи контроллеров). Приложение или вх/вых данные передаются в зависимости от конфигурации удаленного контроллера. Допустимо комбинирование типов контроллеров, когда некоторые контроллеры являются удаленными, а некоторые одноранговыми.

ASCII

Протокол ASCII - простой полудуплексный протокол, использующийся для передачи и/или приема текстовых строк в/из простые устройства (принтер или терминал). Этот протокол поддерживается только инструкцией "EXCH".

Modbus

Протокол Modbus является master/slave протоколом, который позволяет одному и только одному главному (master) контроллеру запрашивать ответы от подчиненных (slave) или действовать на основании запросов. Главный контроллер может адресовать послания подчиненным контроллерам индивидуально или может инициировать широковещательную передачу всем подчиненным контроллерам. Подчиненные контроллеры возвращают отклики только на послания, которые адресовались им индивидуально. Ответы на широковещательные запросы от главного контроллера не посылаются. Поддерживаются Modbus ASCII и дистанционный терминал (RTU).
Режим Modbus Master - режим, позволяющий контроллеру инициировать передачу запроса с ожиданием отклика от подчиненного контроллера. Этот режим поддерживается только инструкцией "EXCH". **Режим Modbus Slave** - режим, позволяющий контроллеру отвечать на запросы от главного контроллера. Этот режим устанавливается по умолчанию, если не сконфигурирован другой тип связи. Контроллер Twido поддерживает стандартные функции данных и управления Modbus и дополнительные услуги по доступу к объектам.

Примечание: В сети RS-485 могут быть установлены 32 устройства (без повторителей) (1 master и до 31 slave), адрес которых может быть от 1 до 247.

Связь TwidoSoft и контроллера

Обзор

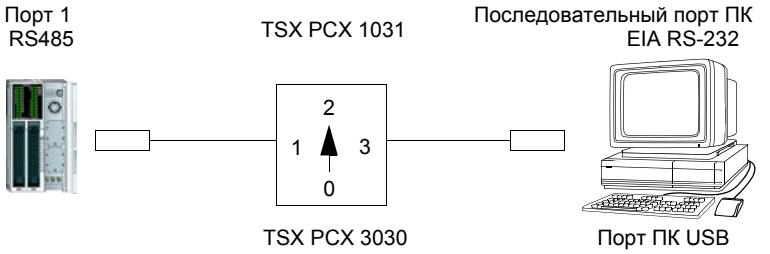
Каждый контроллер имеет встроенный в порт 1 терминальный порт EIA RS-485. У этого порта свой источник питания. Порт 1 должен использоваться для связи с программным обеспечением TwidoSoft.
Никакой опциональный картридж или коммуникационный модуль не может использоваться для этого порта. Модем, однако, может использовать этот порт.
Существует несколько способов соединения ПК с контроллером:

- При помощи кабеля TSX PCX,
- По телефонной линии: модемное соединение.

	**ОСТОРОЖНО** **ПОВРЕЖДЕНИЕ ОБОРУДОВАНИЯ** TwidoSoft может не заметить разъединения, если физически вытащить TSX PCX1031 или TSX PCX 3030 коммуникационный кабель из первого контроллера и вставить его во второй. Используйте TwidoSoft для разъединения перед перемещением кабеля, чтобы избежать такой ситуации. **Несоблюдение этих правил может привести к ущербу или повреждению оборудования.**

Кабельное соединение TSX PCX

Порт EIA RS-232C или USB-порт вашего персонального компьютера соединяется с портом 1 контроллера при помощи многофункционального коммуникационного кабеля TSX PCX1031 или TSX PCX 3030. Этот кабель конвертирует сигналы между EIA RS-232 и EIA RS-485 для TSX PCX 1031и между USB и EIA RS-485 для TSX PCX 3030. Этот кабель оснащается 4-позиционным вращающимся переключателем для выбора различных режимов. Переключатель обозначает четыре положения как "0-3", положение 2 подходит для связи TwidoSoft и контроллера Twido.
Это соединение изображено ниже:



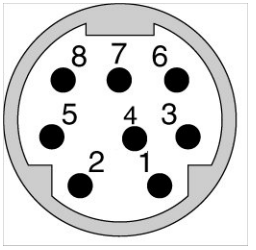
Примечание: Для этого кабеля сигнал DPT на 5-м выводе не привязан к 0В. Это показывает контроллеру, что текущее соединение - соединение с TwidoSoft. Сигнал изменяется изнутри, информируя исполнительные программно-аппаратные средства, что это соединение с TwidoSoft.

Выводы вилок и розеток разъемов

На следующем рисунке показаны выводы вилки 8-ми штыревого разъема и терминала:

Mini DIN

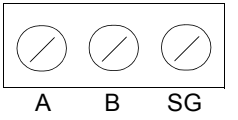
TWD NAC232D, TWD NAC485D
TWD NOZ485D, TWD NOZ232D



Pin outs	Base RS485	RS485 option	RS232-C
1	A (+)	A (+)	RTS
2	B (-)	B (-)	DTR
3	NC	NC	TXD
4	/DE	NC	RXD
5	/DPT	NC	DSR
6	NC	NC	GND
7	0 V	0 V	GND
8	5 V	5 V	5 V

Терминальный блок

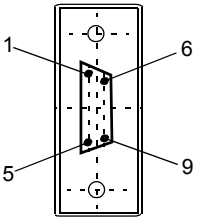
TWD NAC485T
TWD NOZ485T



Pin outs	RS485
A	A(+)
B	B(-)
SG	0V

Примечание: максимальное общее потребление тока для режима 5В (вывод 8): 180мА

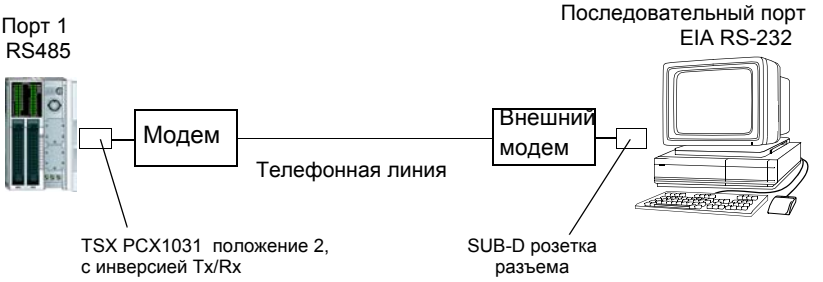
На следующем рисунке показаны выводы 9-ти штыревой розетки SubD разъема для TSX PCX 1031.



Pin outs	RS232
1	DCD
2	RX
3	TX
4	DTR
5	SG
6	NC
7	RTS
8	CTS
9	NC

Соединение по телефонной линии

Модемное соединение (См. *Связь TwidoSoft и модема, стр. 88*) позволяет программировать и связываться с контроллером при помощи телефонной линии.
Модем, ассоциирующийся с контроллером, это **принимающий** модем, связанный с портом 1 контроллера. Модем, ассоциирующийся с ПК, может быть внутренним или внешним и соединен с последовательным портом COM. Это соединение изображено ниже.



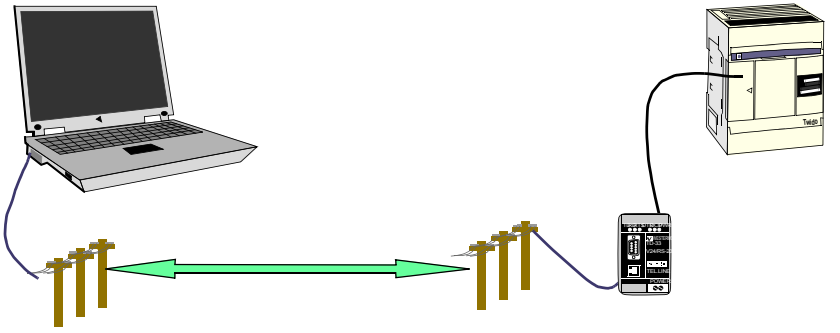
Примечание: Только один модем может быть подключен к порту 1 контроллера.

Примечание: Осторожно. Не забудьте установить программное обеспечение, поставляемое с модемом, потому что TwidoSoft учитывает только установленные модемы.

Связь TwidoSoft и модема

Общие понятия

Компьютер с Twidosoft может быть соединен с контроллером для пересылки приложений, анимационных объектов и выполняющихся команд рабочего режима. Также возможно соединить контроллер Twido с другими устройствами, такими как второй контроллер Twido, для установки связи с процессом приложения.

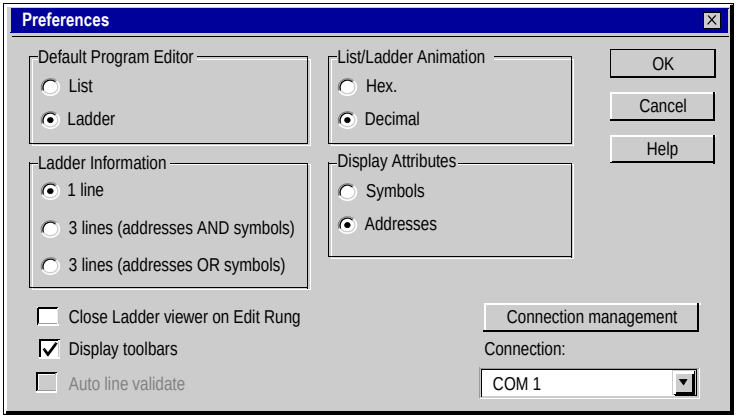


Установка модема

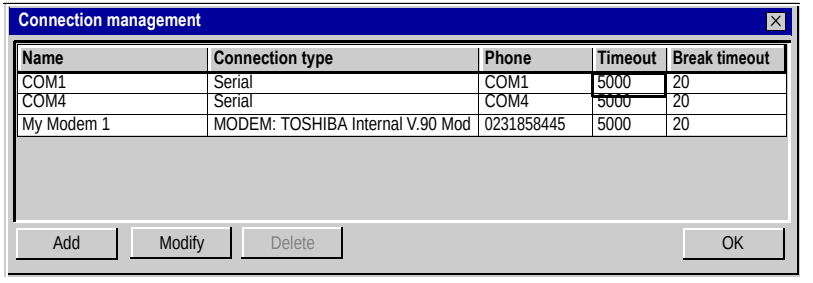
Все модемы, которые пользователь хочет использовать с Twidosoft, должны быть установлены под Windows с компьютера.
Для установки модемов под Windows, следуйте инструкциям Windows.
Эта установка не имеет отношения к Twidosoft.

Установка соединения

Соединение по умолчанию между Twidosoft и контроллером Twido происходит через последовательный порт при помощи кабеля TSX PCX 1031 и скрещивающего адаптера (См. *Приложение 1, стр. 97*).
Если для связи с ПК используется модем, это должно быть отражено в Twidosoft.
Для выбора соединения в Twidosoft, щелкните "file"(“файл”), затем "preferences"(“настройка”).



На этом экране Вы можете выбрать соединение или управлять соединением (создание, модификация и т. д.).
Для того, чтобы использовать существующее соединение, выберите его в выпадающем меню.
Если Вы хотите добавить, изменить или удалить соединение, нажмите на кнопку "Connection management" (Управление соединениями”); в открывшемся окне будет отображен список соединений и их свойства.

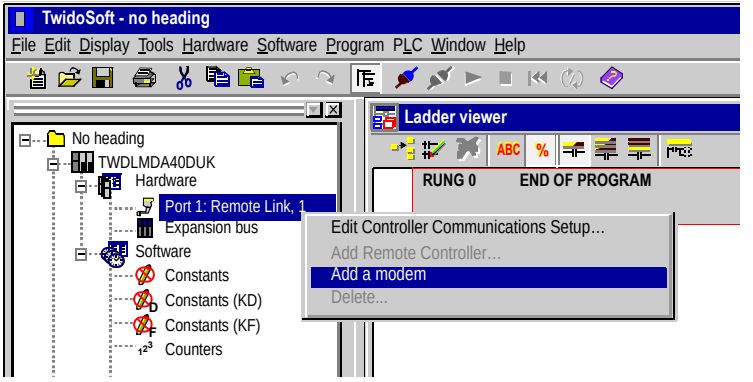


В этом примере отображаются оба последовательных порта и модемное соединение. Используется модем TOSHIBA V.90, настроенный для набора номера: 0231858445 (местный звонок).

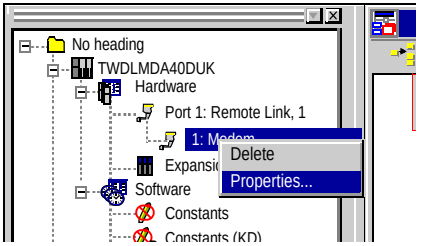
Вы можете поменять название каждого соединения в целях поддержки приложения (COM1 и COM4 не могут быть изменены).
Это касалось того, как определить и выбрать соединение, которое вы хотите использовать для связи ПК с модемом.
Однако, это только часть процесса создания полного соединения компьютера и контроллера Twido.
Следующий шаг касается контроллера Twido. Удаленный контроллер Twido должен быть соединен с модемом.
Все модемы должны быть проинициализированы, чтобы установить соединение. Контроллеры Twido, содержащие программно-аппаратные средства версии 2.0 и выше, способны при запуске послать адаптирующую строку модему, если он сконфигурирован в приложении.

Конфигурирование модема

Процедура конфигурирования модема в контроллере Twido следующая:

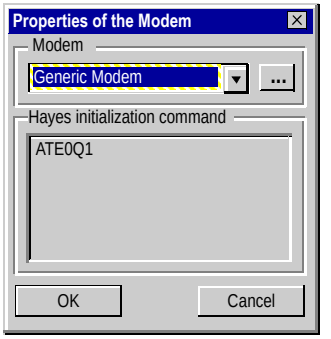


После того как модем сконфигурирован на порт 1, необходимо задать его свойства. Щелкните правой кнопкой мыши на модеме, чтобы открыть доступ к пунктам "delete" (“удаление”) и "properties" (“свойства”). Выбор пункта “Свойства” позволит выбрать известный модем, создать новый модем или изменить свойства модема.

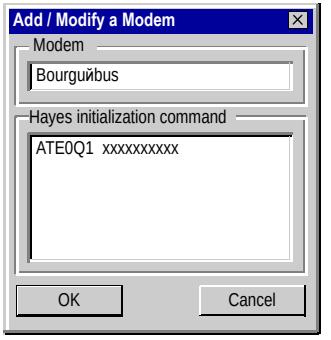


Примечание: Модем полностью управляется контроллером Twido через порт 1. Это означает, что Вы можете подключить модем к порту 2, но в этом случае настройка всех режимов работы модема и их инициализирующая последовательность должны быть выполнены вручную. Это не может быть произведено так же, как при подключении к порту 1.

Следующий шаг, выберите "properties", затем:



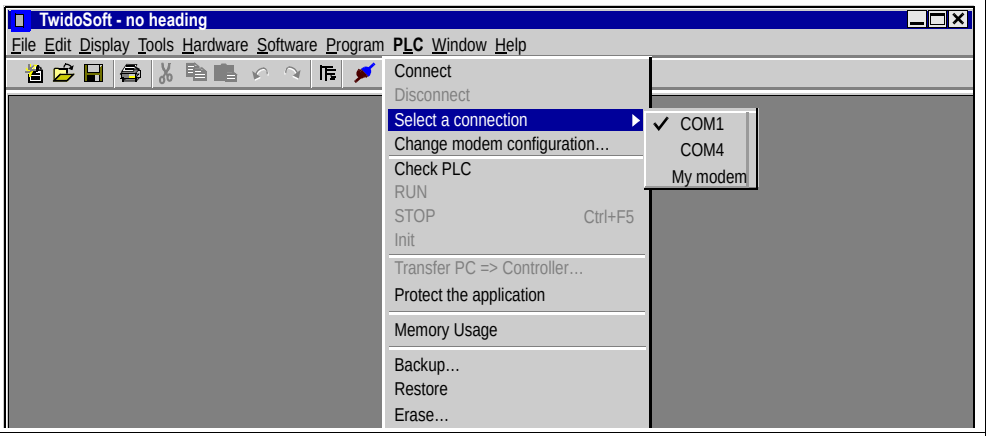
Вы можете выбрать ранее определенный модем или создать новый, нажав "..." .



Затем назовите новый профиль и завершите стандартные команды модема, как это описано в документации к модему.
На рисунке "xxxxxx" отображает инициализирующую последовательность, которую Вы должны ввести для подходящего соединения, например, скорость передачи (в бодах), контроль по четности, стоп бит и режим приема.
Для завершения последовательности, пожалуйста, обратитесь к документации модема.
Максимальная длина строки: 127 символов.
Когда ваше приложение будет завершено, или по крайней мере коммуникационный порт 1 будет полностью описан, перешлите приложение используя прямое соединение.
Контроллер Twido теперь готов к соединению с ПК через модем.

Установка соединения

После подготовки Twidosoft и контроллера Twido, установите соединение следующим образом:

Шаг	Действие
1	Включите контроллер Twido и модем.
2	Включите компьютер и запустите Twidosoft.
3	Выберите меню "PLC" (ПЛК), затем "Select a connection" (выбор соединения) и выберите "My modem" (мой модем) (или название, которое вы дали модемному соединению – см. "установка соединения":) 
4	Присоедините TwidoSoft

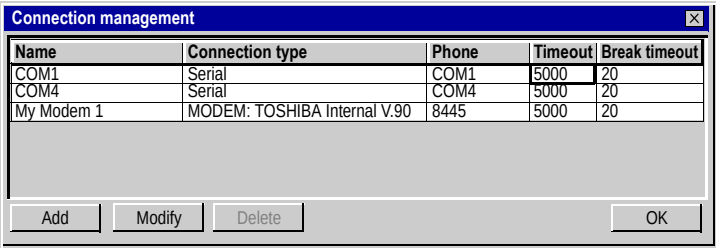
Примечание: Если вы хотите использовать это модемное соединение постоянно, кликните на "file", "preferences"и выберите "my modem" (или название, которое вы ему дали). Twidosoft запомнит эту настройку.

Режимы работы

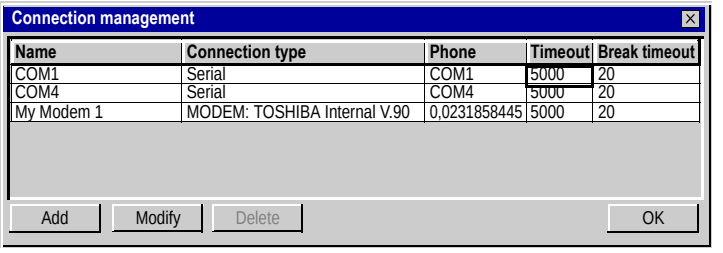
Контроллер Twido посылает инициализирующую строку подсоединенному, включенному модему. Когда модем сконфигурирован в приложении Twido, контроллер сначала посылает команду "FF", чтобы установить, подключен ли модем. Если контроллер получает ответ, инициализирующая строка посылается в модем.

Внутренние, внешние и международные звонки

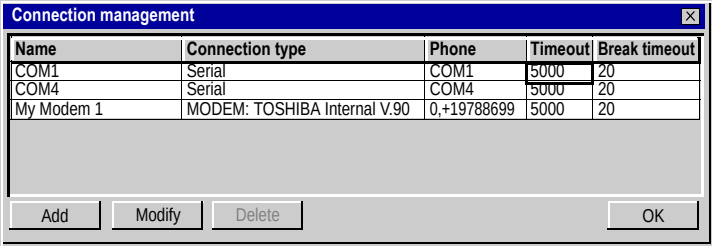
Если Вы связываетесь с контроллером, который находится на территории компании, Вы можете использовать только добавочный код, необходимый для звонка, например: 8445



Если Вы используете внутренний коммутатор, чтобы позвонить по номерам за пределами компании и Вам нужно сначала набрать “0” или “9”, используйте этот синтаксис: 0,0231858445 или 9,0231858445

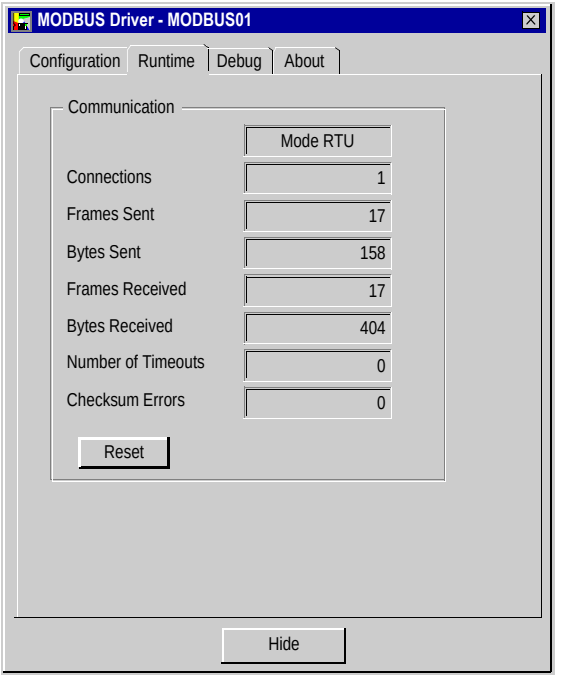


Для международных звонков синтаксис следующий: +19788699001. Если Вы используете коммутатор, то синтаксис: 0,+ 19788699001



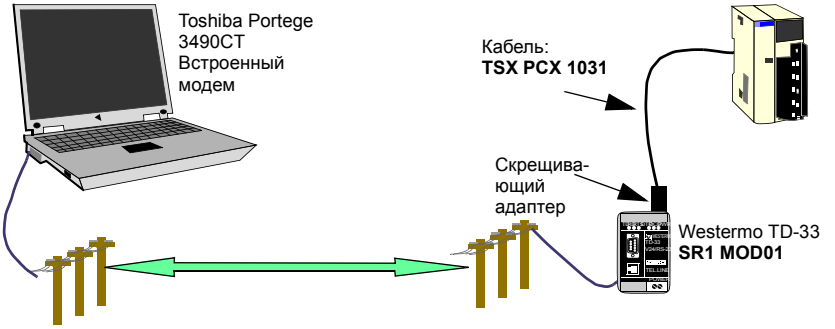
Часто задаваемые вопросы

Когда связь установлена, в течении нескольких минут могут возникать ошибки. В этом случае вам следует настроить параметры коммуникации. Twidosoft использует драйвер modbus для связи через последовательные порты или внутренние модемы. В начале соединения драйвер modbus виден на панели задач. Двойной щелчок на иконке драйвера открывает окно. Теперь у Вас есть доступ к параметрам драйвера modbus; вкладка "runtime" предоставляет информацию о фреймах, которыми модем обменялся с удаленным контроллером.
Если "Number of timeouts" (число тайм-аутов) возрастает или отлично от 0, измените значение, используя "Connection management" (выберите в Twidosof меню "File", затем "Preferences" и "Connection management"). Кликните на поле "timeout", затем нажмите на кнопке изменений (modification) и введите новое, большее значение. Значение по умолчанию 5000 мс. Попробуйте новое соединение. Настраивайте значение, пока соединение не стабилизируется.



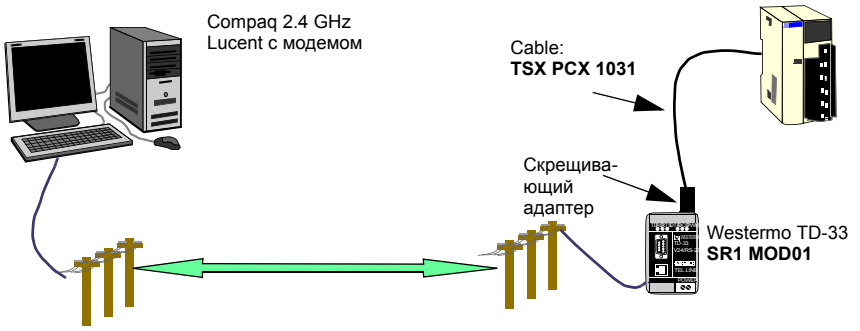
Примеры

- **Пример 1**: Twidosoft соединен с TWD LMDA 20DRT (Windows 98 SE).
  - ПК: Toshiba Portege 3490CT под Windows 98,
  - Модем (внутренний в ПК): Toshiba internal V.90 modem,
  - Контроллер Twido: TWD LMDA 20DRT версия 2.0,
  - Модем (соединенный с Twido): Type Westermo TD-33 / V.90, шифр SR1 MOD01, доступен из нового каталога Twido (Сентябрь 03) (См. *Приложение 2, стр. 98*),
  - Кабель: TSX PCX 1031, соединен с коммуникационным портом 1 Twido, и адаптер: вилка 9-выводов/вилка 9-выводов, чтобы поменять Rx и Tx при соединении модема Westermo и контроллера Twido (См. *Приложение 1, стр. 97*). Вы также можете использовать кабель TSX PCX 1130 (RS485/ 232 преобразование и скрещивание Rx/Tx).



Первый тест включает использование двух аналоговых телефонных линий внутри компании и использование не целого телефонного номера, а добавочного расширения (только 4 цифры номера для внутреннего модема Toshiba V.90).
Для этого теста параметры соединения (Twidosoft, меню "preferences", затем "Connection management") были установлены в значения по умолчанию, тайм-аут = 5000 и прерывающий тайм-аут = 20.

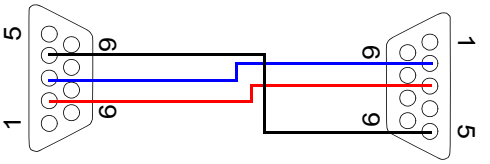
- **Пример 2**: Twidosoft соединен с TWD LMDA 20DRT (windows XP Pro)
  - ПК: Compaq Pentium 4, 2.4GHz,
  - Модем: Lucent Win modem, PCI bus,
  - Контроллер Twido: TWD LMDA 20DRT версия 2.0,
  - Модем (соединенный с Twido): Type WESTERMO TD-33 / V.90, шифр SR1 MOD01, доступен из нового каталога Twido (Сентябрь 03) (См. *Приложение 2, стр. 98*),
  - Кабель: TSX PCX 1031, соединен с коммуникационным портом 1 Twido, и адаптер: вилка 9-выводов/вилка 9-выводов, чтобы поменять Rx и Tx при соединении модема Westermo и контроллера Twido (См. *Приложение 1, стр. 97*). Вы также можете использовать кабель TSX PCX 1130 (RS485/ 232 преобразование и скрещивание Rx/Tx).



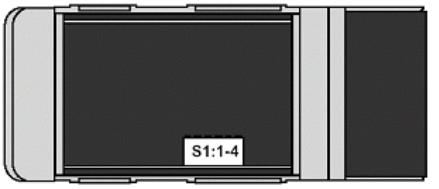
Первый тест включает использование двух аналоговых телефонных линий внутри компании и использование не целого телефонного номера, а добавочного расширения (только 4 цифры номера для внутреннего модема Toshiba V.90).
Для этого теста параметры соединения (Twidosoft, меню "preferences", затем "Connection management") были установлены в значения по умолчанию, тайм-аут = 5000 и прерывающий тайм-аут = 20.

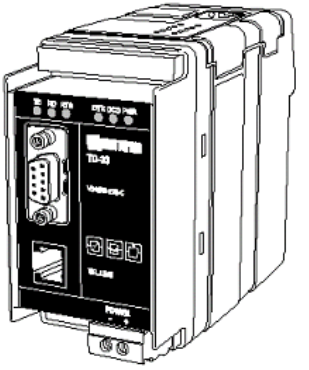
Приложение 1

Скрещивающий адаптер для кабеля TSX PCX 1031 и модема Westermo TD-33 (**SR1 MOD01**):

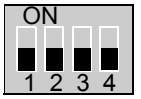


Приложение 2 Модем Westermo TD-33, Schneider шифр **SR1 MOD01**. Этот модем управляет четырьмя переключателями DIP, которые должны быть установлены в **OFF**:





Заводские настройки



Использует сохраненную конфигурацию (скорость & формат и т. д.)
Запрет DTR Hotcall, Auto Band

Приложение 3 Модем Wavecom WMOD2B, Schneider шифр **SR1 MOD02** двух диапазонный (900/1800гц):

Приложение 4 Шифры изделий, упомянутых в этом документе:

- Twido: TWD LMDA 20DRT,
- Twidosoft ПО: TWD SPU 1002 V10M,
- TSX PCX 1031 кабель,
- TSX PCX 1130 кабель,
- Модем RTU: Westermo TD-33 / V90 **SR1 MOD01**,
- Модем GSM: Wavecom WMOD2B **SR1 MOD02**.

Дистанционная связь

Введение

Протокол дистанционной связи является высокоскоростной шиной master/slave, предназначенной для передачи небольшого количества данных между Master контроллером и удаленными Slave контроллерами (до семи контроллеров). Приложение или вх/вых данные передаются в зависимости от конфигурации удаленного контроллера. Допустимо комбинирование типов контроллеров, когда некоторые контроллеры являются удаленными, а некоторые одноранговыми.

Примечание: Master контроллер содержит информацию об адресе удаленных вх/вых. Он не знает, какие конкретно контроллеры находятся по этим адресам. Поэтому, master -контроллер не может подтвердить, что все удаленные входы и выходы, использующиеся в приложении пользователя, действительно существуют.

Примечание: Использование шины удаленных вх/вых и протокола запатентовано и сторонние устройства в сеть не допускаются .

ОСТОРОЖНО

НЕПРЕДСКАЗУЕМАЯ РАБОТА ОБОРУДОВАНИЯ

- Убедитесь, что есть только один master контроллер дистанционной связи, и адреса slave контроллеров уникальны. Несоблюдение этого правила может привести к порче данных или неожиданному и неопределенному результату.
- Убедитесь, что все slave контроллеры имеют уникальные адреса. Никакие два slave контроллера не должны иметь одинаковые адреса. Несоблюдение этого правила может привести к порче данных или неожиданному и неопределенному результату.

Несоблюдение этих правил может привести к ущербу или повреждению оборудования.

Примечание: Дистанционная связь требует соединения EIA RS-485 и одновременно может выполняться только на одном порте.

Аппаратная конфигурация

Дистанционная связь должна использовать минимум 3-х проводной EIA RS-485 порт. Она может быть сконфигурирована на использование либо первого, либо второго опционального порта, в случае его присутствия.

Примечание: Только один коммуникационный порт одновременно может быть сконфигурирован для дистанционной связи.

В следующей таблице перечислены используемые устройства:

Удаленное устр-во	Порт	Спецификации
TWDLCAA10/16/24DRF, TWDLMDA20/40DUK, TWDLMDA20/40DTK, TWDLMDA20DRT	1	Базовый контроллер, оборудованный 3-х проводным EIA RS-485 портом с коннектором miniDIN.
TWDNOZ485D	2	Коммуникационный модуль, оборудованный 3-х проводным EIA RS-485 портом с коннектором miniDIN. **Примечание:** Этот модуль доступен только для модульных контроллеров. Когда этот модуль установлен, у контроллера не может быть модуля дисплея.
TWDNOZ485T	2	Коммуникационный модуль, оборудованный 3-х проводным EIA RS-485 портом с терминалом. **Примечание:** Этот модуль доступен только для модульных контроллеров. Когда этот модуль установлен, у контроллера не может быть модуля дисплея.
TWDNAC485D	2	Коммуникационный адаптер, оборудованный 3-х проводным EIA RS-485 портом с коннектором miniDIN. **Примечание:** Этот адаптер доступен только для компактных контроллеров Compact 16 и Compact 24 и модуля дисплея.
TWDNAC485T	2	Коммуникационный адаптер, оборудованный 3-х проводным EIA RS-485 портом с терминалом. **Примечание: Этот адаптер доступен только для компактных контроллеров Compact 16 и Compact 24 и модуля дисплея.**
TWDXCPODM	2	Дисплей, оборудованный 3-х проводным EIA RS-485 портом с коннектором miniDIN или 3-х проводным EIA RS-485 портом с терминалом. **Примечание:** Этот модуль доступен только для модульных контроллеров. Когда этот модуль установлен, у контроллера не может быть коммуникационного модуля расширения.

Примечание: Вы можете только проверить наличие и конфигурацию (RS232 или RS485) порта 2 при включении или сбросить состояние при помощи встроенной программы.

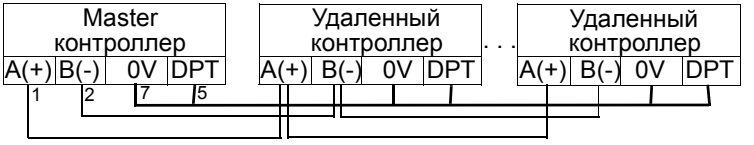
Соединение кабелей для каждого устройства

Примечание: Сигнал DPT на выводе 5 должен быть привязан к 0V на выводе 7, чтобы обозначить использование дистанционной связи. Когда этот сигнал не привязан к земле, контроллер Twido, master или slave, будет по умолчанию пытаться установить соединение с TwidoSoft.

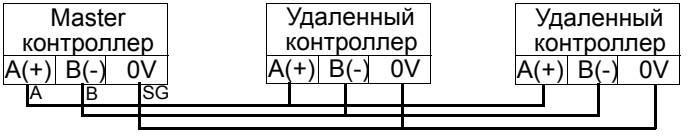
Примечание: Соединение DPT и 0V необходимо только тогда, когда Вы соединяетесь с базовым контроллером через порт 1.

Соединение кабелей для каждого удаленного устройства изображено ниже.

Соединение Mini-DIN



Соединение терминального блока



Программная конфигурация

Должен быть только один master контроллер, определенный для дистанционной связи. Кроме того, у каждого удаленного контроллера должен быть уникальный адрес slave. Несколько master или slave контроллеров, использующих идентичные адреса, могут привести к порче данных или к неопределенности.

ОСТОРОЖНО

НЕПРЕДСКАЗУЕМАЯ РАБОТА ОБОРУДОВАНИЯ

- Убедитесь, что есть только один master контроллер дистанционной связи, и адреса slave контроллеров уникальны. Несоблюдение этого правила может привести к порче данных или неожиданному и неопределенному результату.

Несоблюдение этих правил может привести к ущербу или повреждению оборудования.

Конфигурация master контроллера

Master контроллер конфигурируется при помощи TwidoSoft для управления до 7 удаленными контроллерами дистанционной связи. Эти семь контроллеров могут быть сконфигурированы, как удаленные вх/вых или как равноправные контроллеры. Адрес master контроллера, сконфигурированный при помощи TwidoSoft, соответствует адресу 0. Чтобы сконфигурировать контроллер, как master контроллер, при помощи TwidoSoft сконфигурируйте порт 1 или порт 2 для дистанционной связи и выберите адрес 0 (Master). Затем из окна "Add remote controller"(добавить удаленный контроллер) Вы можете установить, будут ли slave контроллеры удаленными вх/вых или равноправными контроллерами, а также задать их адреса.

Конфигурация удаленного контроллера

Удаленный контроллер конфигурируется в TwidoSoft, установкой порта 1 или 2 для дистанционной связи или присваиванием порту адреса от 1 до 7.
В следующей таблице приведены различия и ограничения каждой из конфигураций удаленного контроллера:

Тип	Прикладная программа	Доступ к данным
Удаленный вх/вых	Нет. Даже для простого оператора "END". Режим РАБОТА связан с режимом Master.	%I и %Q. Доступны только локальные вх/вых контроллера (а не вх/вых расширения).
Равно-правный контроллер	Да. Режим РАБОТА не связан с режимом Master.	%INW и %QNW. Максимум 4 входных слова и 4 выходных слова могут быть переданы в/из узла.

Синхронизация сканирования удаленного контроллера

Цикл обновления удаленной связи не синхронизирован с циклом сканирования master контроллера. Связь с удаленным контроллером управляется прерыванием и происходит, как фоновая задача, параллельно выполнению сканирования master контроллера. В конце цикла сканирования самые новые значения считываются в приложение, чтобы использоваться при следующем выполнении программы. Этот процесс одинаков для удаленных вх/вых и равноправных контроллеров. Любой контроллер может проверить общую активность связи, используя системный бит %S111. Для достижения синхронизации master или равноправный контроллер должны использовать системный бит %S110. Этот бит устанавливается в 1, когда произошел полный цикл обновления. Прикладная программа отвечает за установку этого бита в 0. Master контроллер может разрешить или запретить удаленную связь, используя системный бит %S112. Контроллеры могут проверить правильную конфигурацию и работу дистанционной связи, используя %S113. Сигнал DPT порта 1 (использующийся для определения, подсоединен ли TwidoSoft) считывается и выводится в %S100.
Это обобщено в следующей таблице:

Бит	Статус	Индикация
%S100	0	master/slave: DPT не активен (TwidoSoft кабель НЕ подключен)
	1	master/slave: DPT активен (TwidoSoft кабель подключен)
%S110	0	master/slave: устанавливается в 0приложением
	1	master: все обмены по дистанционной связи закончены (только для удаленных вх/вых) slave: обмен с master закончен
%S111	0	master: отдельный обмен по дистанционной связи закончен slave: отдельный обмен по дистанционной связи определен
	1	master: происходит отдельный обмен по дистанционной связи slave: отдельный обмен по дистанционной связи определен
%S112	0	master: дистанционная связь запрещена
	1	master: дистанционная связь разрешена
%S113	0	master/slave: конфигурация/функционирование дистанционной связи в порядке
	1	master: ошибка конфигурации/функционирования дистанционной связи slave: ошибка функционирования дистанционной связи

Перезапуск master контроллера

При перезапуске master контроллера происходит одно из следующих событий:
- “Холодный” пуск (%S0 = 1) приводит к реинициализации коммуникаций.
- “Теплый” пуск (%S1 = 1) приводит к реинициализации коммуникаций.
- В режиме ОСТАНОВКА, master контроллер продолжает связь с slave контроллерами.

Перезапуск slave контроллера

При перезапуске slave контроллера происходит одно из следующих событий:
- “Холодный” пуск (%S0 = 1) приводит к реинициализации коммуникаций.
- “Теплый” пуск (%S1 = 1) приводит к реинициализации коммуникаций.
- В режиме ОСТАНОВКА, slave контроллер продолжает связь с master контроллерами. Если master в состоянии ОСТАНОВКА:
  - Удаленные вх/вых принимают состояние ОСТАНОВКА.
  - Равноправный контроллер сохраняет свое состояние.

Остановка master контроллера

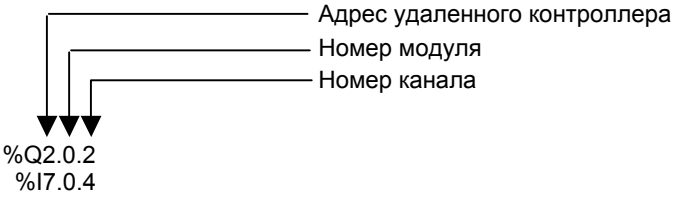
Когда master контроллер входит в режим ОСТАНОВКА, все slave устройства продолжают общаться master. Когда master указывает, что требуется ОСТАНОВКА, удаленный контроллер вх/вых перейдет в режим ОСТАНОВКА, но равноправный контроллер сохранит текущий режим РАБОТА или ОСТАНОВКА.

Доступ к данным удаленного вх/вых

Удаленный контроллер, сконфигурированный как удаленный вх/вых, не имеет и не исполняет собственной прикладной программы. Цифровые входы и выходы удаленного контроллера являются выводами расширения master контроллера. Приложение должно использовать предоставляемый механизм полной адресации тремя цифрами.

Примечание: Для удаленных вх/вых номер модуля всегда =0.

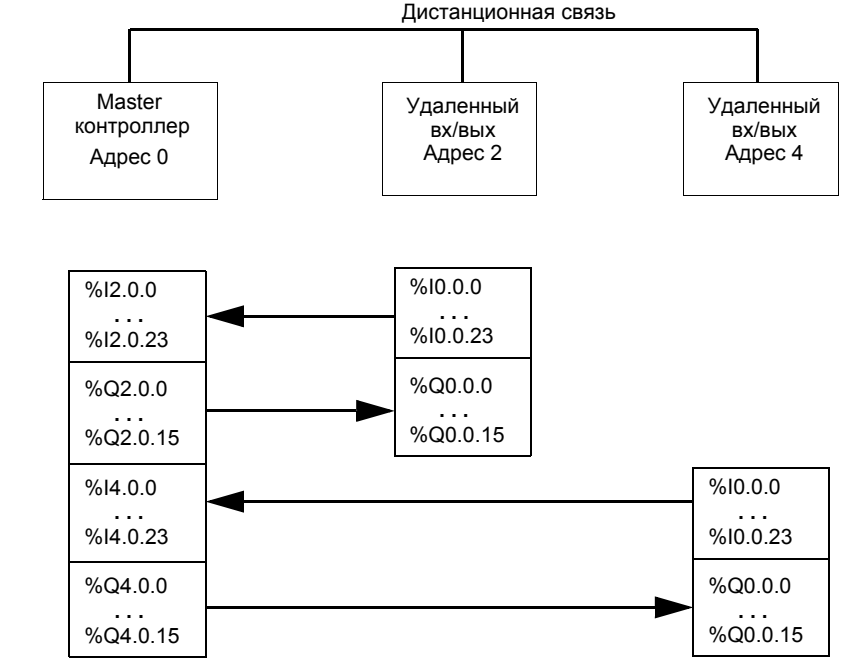
Иллюстрация:



Для связи с удаленным вх/вых master контроллер использует стандартную запись входов и выходов %I и %Q. Для доступа к третьему выходному биту удаленного вх/вых с адресом 2, используется инструкция %Q2.0.2. Аналогично, чтобы прочитать пятый входной бит удаленного вх/вых с адресом 7, используется инструкция %I7.0.4.

Примечание: Master контроллер может получить доступ только к цифровым вх/вых, которые являются частью локальных вх/вых удаленного контроллера. Нельзя обмениваться информацией ни с аналоговыми вх/вых, ни с выходами расширения, пока Вы используете равноправное соединение.

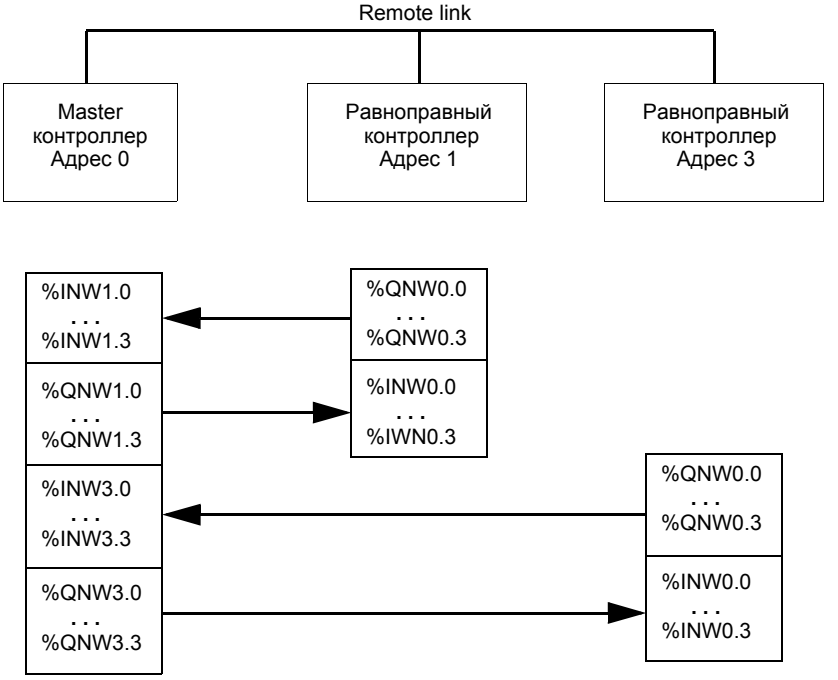
Иллюстрация



Доступ к данным равноправного контроллера

Для обмена данными с равноправными контроллерами master контроллер использует сетевые слова %INW и %QNW. К каждому равноправному контроллеру в сети можно получить доступ по его удаленному адресу "j", используя слова %INWj.k и %QNWj.k. Каждый равноправный контроллер в сети использует слова с %INW0.0 по %INW0.3 и с %QNW0.0 по %QNW0.3 доступа к данным master контроллера. Сетевые слова обновляются автоматически, когда контроллеры находятся в режиме РАБОТА или ОСТАНОВКА.

Следующий пример показывает обмен между master контроллерами и двумя равноправными контроллерами.



В протоколе дистанционной связи нет прямой связи между равноправными узлами. Можно использовать прикладную программу на master контроллере для управления сетевыми словами, чтобы передавать информацию между удаленными контроллерами. Master контроллер выступает в роли моста.

Информация о состоянии

Кроме рассмотренных ранее системных битов, master контроллер поддерживает присутствие и сконфигурированное состояние удаленных контроллеров. Это действие выполняется при помощи системных слов %SW111 и %SW113. Либо удаленный, либо master контроллер может запросить значение последней ошибки, произошедшей при дистанционной связи. Значение хранится в системном слове %SW112.

Систем-ные слова	**Использование**	
%SW111	Статус дистанционной связи: два бита на каждый удаленный контроллер (только master)	
	x0-6	0-Удаленный контроллер 1-7 отсутствует
		1-Удаленный контроллер 1-7 присутствует
	x8-14	0-Удаленный вх/вых определен на удаленном контроллере 1-7
		1-Равноправный вх/вых определен на удаленном контроллере 1-7
%SW112	Код ошибки конфигурации/функционирования дистанционной связи	
		0 - операции успешны
		1 - обнаружен тайм-аут (slave)
		2 - обнаружена ошибка контрольной суммы (slave)
		3 - несоответствие конфигурации (slave)
%SW113	Конфигурация дистанционной связи: два бита на каждый удаленный контроллер (только master)	
	x0-6	0-Удаленный контроллер 1-7 не сконфигурирован
		1-Удаленный контроллер 1-7 сконфигурирован
	x8-14	0-Удаленный вх/вых сконфигурирован, как удаленный контроллер 1-7
		1-Равноправный контроллер сконфигурирован, как удаленный контроллер 1-7

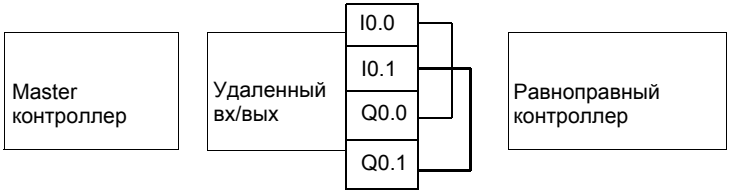
Пример дистанционной связи

Для конфигурации дистанционной связи Вы должны:

1. Сконфигурировать аппаратные средства.
2. Соединить контроллеры.
3. Соединить коммуникационным кабелем ПК и контроллеры.
4. Сконфигурировать аппаратные средства.
5. Написать приложения.

Следующие рисунки иллюстрируют использование дистанционной связи с удаленными вх/вых и равноправными контроллерами.

Шаг 1: Сконфигурировать аппаратные средства:



Аппаратная конфигурация: три базовых контроллера любого типа. Порт 1 используется для двух режимов коммуникации. Первый режим используется для конфигурации и пересылки прикладной программы TwidoSoft. Второй режим используется для сети дистанционной связи. При наличии опционального порта 2 на любом контроллере, он может использоваться, но контроллер поддерживает только одиночную дистанционную связь.

Примечание: в этом примере первые два входа контроллера удаленных вх/вых жестко соединены с первыми двумя выходами.

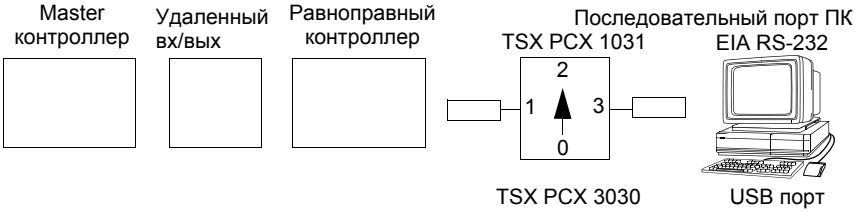
Шаг 2: Соединить контроллеры

Соединение Mini-DIN

Соединение терминального блока

Соедините сигнальные провода A(+) и B(-). На каждом контроллере сигнал DPT соедините с землей. Это хорошая практика, хотя привязка сигналов к земле не требуется при использовании дистанционной связи через порт 2 (опциональный картридж или коммуникационный модуль).

Шаг 3: Соединить коммуникационным кабелем ПК и контроллеры:
:



Многофункциональный программируемый кабель TSX PCX1031 или TSX PCX 3030 используется для связи с каждым из трех контроллеров. Убедитесь, что кабель переключен в положение 2. Вместо программирования каждого контроллера необходимо установить одноранговую связь с каждым контроллером. Для установления этого соединения: соедините ПК с портом 1 первого контроллера, перешлите конфигурационные и прикладные данные и переведите контроллер в режим РАБОТА. Повторите эту процедуру для каждого контроллера.

Примечание: необходимо перемещать кабель после каждой пересылки конфигурационных и прикладных данных**.**

Шаг 4: Сконфигурировать аппаратные средства:
каждый из трех контроллеров использует TwidoSoft для создания конфигурации и, если необходимо, прикладной программы. Для master контроллера, отредактируйте настройки коммуникаций контроллера, чтобы установить протокол "Remote Link" (дистанционная связь) и адрес "0 (Master)".

Controller comm. settings
Type: Remote link
Address: 0 (Master)

Сконфигурируйте удаленный контроллер на master контроллере, добавив "Remote I/O"(удал. вх/вых) по адресу "1" и "Peer PLC"(равнопр. ПЛК) по адресу "2".

Add Remote Controllers

Controller Usage: Remote I/O
Remote Address: 1

Controller Usage: Peer controller
Remote Address: 2

Для контроллера, сконфигурированного как удаленные вх/вых, убедитесь, что он настроен для удаленной связи ("Remote Link"), и адрес установлен в "1".

```
Controller comm. settings
Type:     Remote link
Address: 1
```

Для контроллера, сконфигурированного как равноправный, убедитесь, что он настроен для удаленной связи ("Remote Link"), и адрес установлен в "2".

```
Controller comm. settings
Type:     Remote link
Address: 2
```

Шаг 5: Написать приложения:
Для Master контроллера напишите следующую программу приложения:

```
LD 1

[%MW0 := %MW0 +1]
[%QNW2.0 := %MW0]
[%MW1 := %INW2.0]

LD %I0.0
ST %Q1.00.0
LD %I1.0.0
ST %Q0.0

LD %I0.1
ST %Q1.0.1
LD %I1.0.1
ST %Q0.1
```

Для удаленных вх/вых не пишите никакой программы приложения.
Для равноправного контроллера напишите следующую программу приложения::

```
LD 1
[%QNW0.0 := %INW0.0]
```

В этом примере приложение master контроллера увеличивает слово внутренней памяти и передает его равноправному контроллеру, используя одиночное слово сетевого обмена. Равноправный контроллер получает слово и отправляет его обратно. В master контроллере другое слово памяти получает и хранит это переданное слово.
Для связи с контроллером удаленных вх/вых, master посылает значения локальных входов на удаленные выходы. Используя жесткое соединение внешних вх/вых, сигналы возвращаются и извлекаются master контроллером.

Коммуникации ASCII

Введение

Протокол ASCII предоставляет контроллерам Twido простой полудуплексный протокол, использующийся для передачи и/или приема текстовых строк в/из простые устройства. Этот протокол поддерживается инструкцией EXCH и контролируется при помощи функционального блока %MSGx.
ASCII протокол поддерживает три типа передачи:

- Только передача
- Передача/прием
- Только прием

Максимальный размер фрейма, переданного и/или полученного при помощи инструкции EXCHx, составляет 256 байт.

Аппаратная конфигурация

Связь ASCII (См. системные биты %S103 и %S104 (См. *Системные биты (%S), стр. 434*)) может быть установлена через порт EIA RS-232 или порт EIA RS-485 и может осуществляться одновременно через два коммуникационных порта.
В следующей таблице перечислены используемые устройства:

Удаленное устр-во	Порт	Спецификации
TWDLCAA10/16/24DRF, TWDLMDA20/40DUK, TWDLMDA20/40DTK, TWDLMDA20DRT	1	Базовый контроллер, оборудованный 3-х проводным EIA RS-485 портом с коннектором miniDIN.
TWDNOZ232D	2	Коммуникационный модуль, оборудованный 3-х проводным EIA RS-232 портом с коннектором miniDIN. **Примечание:** Этот модуль доступен только для модульных контроллеров. Когда этот модуль установлен, у контроллера не может быть модуля дисплея.
TWDNOZ485D	2	Коммуникационный модуль, оборудованный 3-х проводным EIA RS-485 портом с коннектором miniDIN. **Примечание:** Этот модуль доступен только для модульных контроллеров. Когда этот модуль установлен, у контроллера не может быть модуля дисплея.
TWDNOZ485T	2	Коммуникационный модуль, оборудованный 3-х проводным EIA RS-485 портом с терминалом. **Примечание:** Этот модуль доступен только для модульных контроллеров. Когда этот модуль установлен, у контроллера не может быть модуля дисплея.

Удаленное устр-во	Порт	Спецификации
TWDNAC232D	2	Коммуникационный адаптер, оборудованный 3-х проводным EIA RS-232 портом с коннектором miniDIN. **Примечание: Этот адаптер доступен только для компактных контроллеров Compact 16 и Compact 24 и модуля дисплея.**
TWDNAC485D	2	Коммуникационный адаптер, оборудованный 3-х проводным EIA RS-485 портом с коннектором miniDIN. **Примечание:** Этот адаптер доступен только для компактных контроллеров Compact 16 и Compact 24 и модуля дисплея.
TWDNAC485T	2	Коммуникационный адаптер, оборудованный 3-х проводным EIA RS-485 портом с терминалом. **Примечание: Этот адаптер доступен только для компактных контроллеров Compact 16 и Compact 24 и модуля дисплея.**
TWDXCPODM	2	Дисплей, оборудованный 3-х проводным EIA RS-232 портом с коннектором miniDIN, 3-х проводным EIA RS-485 портом с коннектором miniDIN и 3-х проводным EIA RS-485 портом с терминалом. **Примечание:** Этот модуль доступен только для модульных контроллеров. Когда этот модуль установлен, у контроллера не может быть коммуникационного модуля расширения.

Примечание: Вы можете только проверить наличие и конфигурацию (RS232 или RS485) порта 2 при включении или сбросить состояние при помощи встроенной программы.

Номинальные соединения кабелей

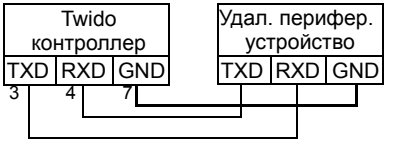
Номинальные соединения кабелей типов EIA RS-232 и EIA RS-485 изображены на следующем рисунке.

Примечание: Если порт 1 контроллера используется, сигнал DPT на выводе 5 должен быть привязан к 0V на выводе 7. Это обозначает, что связь через порт 1 это ASCII, а не протокол для связи с TwidoSoft.

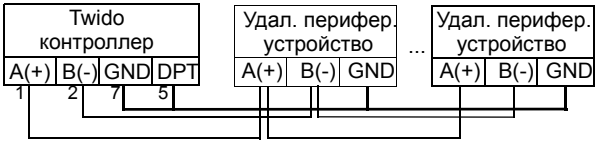
На рисунке показано соединения кабеля с каждым устройством.

Соединение Mini-DIN

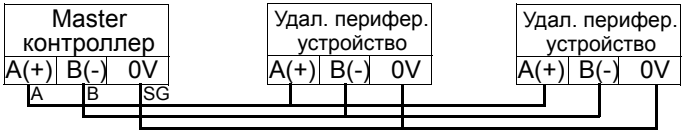
RS-232 EIA кабель



RS-485 EIA кабель



Соединение с терминальным блоком



Программная конфигурация

Для того, чтобы настроить контроллер для использования последовательного порта для отправки и получения символов при помощи протокола ASCII, Вы должны:

Шаг	Описание
1	Сконфигурировать последовательный порт для ASCII при помощи TwidoSoft.
2	Создать в приложении таблицу отправки/получения, которая будет использоваться инструкцией EXCHx.

Конфигурирование порта

Контроллер Twido может использовать основной порт 1 или опциональный порт 2 для протокола ASCII. Чтобы сконфигурировать последовательный порт для ASCII:

Шаг	Действие
1	Определите любые дополнительные коммуникационные адаптеры или модули сконфигурированные для базы.
2	Щелкните правой кнопкой на порте, щелкните “Edit Controller Comm Setup...” и измените тип последовательного порта на "ASCII".
3	Установите коммуникационные параметры.

Конфигурирование таблицы отправки/ получения для режима ASCII

Максимальный размер переданного и/или полученного фрейма равен 256 байтам. Таблица слов, связанная с инструкцией EXCHx, состоит из управляющих таблиц отправки и получения.

	Старший значимый байт	Младший значимый байт
Управляющая таблица	Команда	Длина (отправка/получение)
	Зарезервирован (0)	Зарезервирован (0)
Таблица отправки	Отправленный байт 1	Отправленный байт 2
	...	...
	...	Отправленный байт n
	Отправленный байт n+1	
Таблица получения	Полученный байт 1	Полученный байт 2
	...	...
	...	Полученный байт p
	Полученный байт p+1	

Управляющая таблица

Байт **Длина** содержит длину таблицы пересылки в байтах (250 макс.). В него записывается число полученных символов в конце получения, если получение требуется.

Байт **Команда** должен содержать одно значение из следующих:

- 0: Только передача
- 1: Передача/прием
- 2: Только прием

Таблицы отправки/ получения

В режиме “Только передача” управляющие таблицы и таблицы пересылки заполняются перед выполнением инструкции EXCHx , и могут быть типа %KW и %MW. В этом режиме не требуется свободного места для приема символов. После передачи всех байтов %MSGx.D устанавливается в 1 и может выполняться новая инструкция EXCHx.
В режиме “Передача/прием” управляющие таблицы и таблицы пересылки заполняются перед выполнением инструкции EXCHx , и должны быть типа %MW. Требуется свободное место в конце таблицы отсылки для приема до 256 байт. После передачи всех байтов, контроллер Twido переключается в режим получения и ожидает получения байтов.
В режиме “Только получение” управляющие таблицы и таблицы пересылки заполняются перед выполнением инструкции EXCHx , и должны быть типа %MW. Требуется свободное место в конце управляющей таблицы для приема до 256 байт. Контроллер Twido сразу переходит в режим приема и ожидает получения байтов.
Получение заканчивается, когда получен байт “конец фрейма” или таблица получения заполнена. В этом случае ошибка (переполнение таблицы получения) выставляется в слова %SW63 и %SW64. Если сконфигурирован тайм-аут не равный нулю, получение заканчивается, когда время вышло. Если тайм-аут равен 0, нет тайм-аута на получение, поэтому для окончания получения должен быть активирован вход %MSGx.R.

Обмен сообщениями

Язык предоставляет две службы обмена сообщениями:

- **Инструкция EXCHx :** для отправки/получения сообщений
- **Функциональный блок %MSGx:** для контроля обмена.

Контроллер Twido использует протокол порта при выполнении инструкции EXCHx.

Примечание: Каждый коммуникационный порт может быть сконфигурирован для разных или одинаковых протоколов. Инструкция EXCHx или функциональный блок %MSGx для каждого коммуникационного порта достигается прибавлением номера порта (1 или 2).

Инструкция EXCHx

Инструкция EXCHx позволяет контроллеру Twido посылать и/или получать информацию к/от устройств ASCII. Пользователь задает таблицу слов (%MWi:L или %KWi:L), содержащую управляющую информацию или данные для отсылки и/или приема (до 256 байт в каждом направлении). Формат таблицы был описан ранее.
Обмен сообщениями происходит посредством инструкции EXCHx:

Синтаксис: [EXCHx %MWi:L] или [EXCHx %KWi:L]

где: x = номер порта (1 или 2)
L = число слов в таблицах управления, отсылки и получения

Контроллер Twido должен завершить обмен по первой инструкции EXCHx, прежде чем запускать следующую инструкцию. При отправке нескольких сообщений должен использоваться функциональный блок %MSGx. Выполнение инструкции EXCHx начинается немедленно, передача по прерыванию (получение данных тоже происходит по прерыванию) происходит в фоновом режиме.

Функциональный блок %MSGx

Использование функционального блока %MSGx является необязательным; он может использоваться для управления обменом данными. У функционального блока %MSGx есть три назначения:

- **Проверка ошибок коммуникаций**
  Проверка ошибок проверяет, чтобы параметр L (длина таблицы слов) заданный в инструкции EXCHx, достаточен для хранения длины передаваемых сообщений. Это значение сравнивается с длиной младшего значимого байта первого слова в таблице слов.
- **Согласование передачи нескольких сообщений**
  Для обеспечения координации при посылке нескольких сообщений, функциональный блок %MSGx обеспечивает информацию, требуемую, чтобы определить, что передача предыдущего сообщения завершена.
- **Передача приоритетных сообщений**
  Функциональный блок %MSGx позволяет остановить текущую передачу сообщений, чтобы позволить немедленную передачу срочного сообщения.

Функциональный блок %MSGx имеет один вход и два выхода:

Вход/Выход	Определение	Описание
R	Вход сброса	Установлен в 1: реинициализация связи или сброс блока (%MSGx.E = 0 и %MSGx.D = 1).
%MSGx.D	Связь завершена	0: происходит запрос. 1: связь завершена, если передача закончилась, получен символ завершения, ошибка или сброс блока.
%MSGx.E	Ошибка	0: длина сообщения и связь в порядке. 1: неправильная команда, или конфигурация таблицы, получен неправильный символ (скорость, четность и т. д.) или переполнение таблицы получения.

Ограничения

Важно отметить следующие ограничения:

- Доступность и тип порта 2 (см %SW7) проверяется при пуске и перезагрузке
- Все сообщения через порт 1прерываются, когда подключается TwidoSoft
- Нельзы выполнить EXCHx или %MSG на порте дистанционной связи
- EXCHx прерывает активное выполнение Modbus Slave
- Выполнение инструкции EXCHx не повторяется в случае ошибки
- Вход сброса (R) может использоваться для сброса выполнения инструкции получения EXCHx
- Инструкция EXCHx может конфигурироваться с тайм-аутом для прекращения получения
- Множественные сообщения контролируются через %MSGx.D

Ошибка и условия рабочего режима

Если во время выполнения инструкции EXCHx происходит ошибка, биты %MSGx.D и %MSGx.E устанавливаются в 1, системное слово %SW63 содержит код ошибки для порта 1, системное слово %SW64 содержит код ошибки для порта 2.

Системные слова	Использование
%SW63	Код ошибки EXCH1: 0 - выполнение было успешно 1 - число байтов для пересылки слишком большое (> 250) 2 - таблица пересылки слишком маленькая 3 - таблица слов слишком маленькая 4 - переполнение таблицы получения 5 - тайм-аут 6 - ошибка пересылки 7 - неправильная команда в таблице 8 - выбранный порт не сконфигурирован/не доступен 9 - ошибка получения (только режим ASCII) 10 - при получении невозможно использовать %KW 11 - смещение пересылки больше, чем таблицы пересылки 12 - смещение получения больше, чем таблицы получения 13 - контроллер остановил выполнение EXCH
%SW64	Код ошибки EXCH2: см. %SW63.

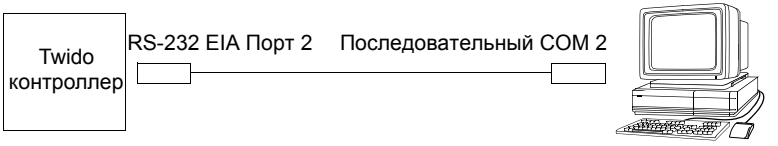
Последствие перезапуска контроллера во время связи

При перезапуске контроллера происходит одно из следующих событий:
- “Холодный” старт (%S0 = 1) приводит к реинициализации коммуникаций.
- “Теплый” старт (%S1 = 1) приводит к реинициализации коммуникаций.
- В режиме ОСТАНОВКА контроллер останавливает все коммуникации ASCII.

Пример связи ASCII

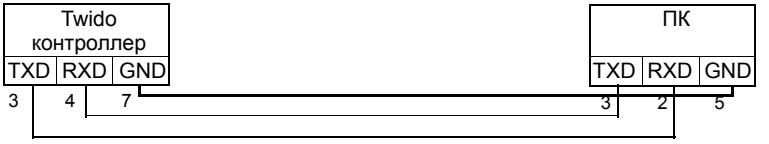
Для конфигурирования связи ASCII, Вы должны:
1. Сконфигурировать аппаратные средства.
2. Соединить коммуникационный кабель ASCII.
3. Сконфигурировать порт.
4. Написать приложение.
5. Инициализировать Редактор анимационных таблиц.
На следующем рисунке показано использование ASCII-связи с эмулятором терминала на ПК.

Шаг 1: Сконфигурировать аппаратные средства:



Аппаратная конфигурация: два последовательных соединения ПК и опционального EIA RS-232 порта 2 контроллера Twido. Для модульного контроллера опциональный порт 2 это TWDNOZ232D или TWDNAC232D для TWDXCPODM. Для компактного контроллера опциональный порт 2 это TWDNAC232D.
Для конфигурирования контроллера, соедините кабель TSX PCX1031 (не изображен) к порту 1 контроллера Twido. Затем соедините кабель к порту COM 1 компьютера. Убедитесь, что переключатель в положении 2. В конце соедините порт COM 2 компьютера с опциональным портом 2 EIA RS-232 контроллера Twido. Монтажная схема представлена в следующем шаге.

Шаг 2: Монтажная схема коммуникационного кабеля ASCII (EIA RS-232):



Минимальное количество проводов, используемых при связи ASCII, 3 провода. Пересечение сигналов приема и передачи.

Примечание: Могут понадобиться дополнительные соединения кабеля со стороны ПК (такие как Data Terminal Ready или Data Set Ready) для обеспечения квитирования. Для контроллера Twido дополнительные соединения не требуются.

Шаг 3: Сконфигурировать порт:

Hardware -> Add Option TWDNOZ232D

Hardware => Adjust Controller Comm. Setting	
Port:	2
Type:	ASCII
Baud Rate:	19200
Data:	8 Bit
Parity:	None
Stop:	1 Bit
End of Frame:	65
Response Timeout:	100 x 100 ms

Terminal Emulator on a PC	
Port:	COM2
Baud Rate:	19200
Data:	8 Bit
Parity:	None
Stop:	1 Bit
Flow control:	None

Используйте простой эмулятор терминала на ПК, чтобы сконфигурировать порт COM2 и убедиться, что нет управляющей логики.
Используйте TwidoSoft, чтобы сконфигурировать порт контроллера. Во-первых, сконфигурируйте аппаратные опции. В этом примере TWDNOZ232D добавляется к базовому модульному контроллеру.
Во-вторых, инициализируйте Controller Communication Setup теми же параметрами настройки, как у эмулятора терминала на ПК. В этом примере заглавная буква "A" выбрана в качестве символа "Конец фрейма", для завершения приема символов. В качестве параметра “тайм-аут ответа” выбран 10 секундный тайм-аут. Только один из этих двух параметров будет задействован, в зависимости от того, какое событие произойдет первым.

Шаг 4: Напишите приложение:

```
LD 1
[%MW10 := 16#0104 ]
[%MW11 := 16#0000 ]
[%MW12 := 16#4F4B ]
[%MW13 := 16#0A0D ]
LD 1
AND %MSG2.D
[EXCH2 %MW10:8]
LD %MSG2.E
ST %Q0.0
END
```

Используйте TwidoSoft для создания прикладной программы, состоящей из трех основных частей. Во-первых, инициализируйте таблицу управления и передачи для использования с инструкцией EXCH. В этом примере команда установлена для режима приема и пересылки данных. Количество данных для передачи 4 байта и данные инициализированы: "O", "K", CR, LF.

Затем, проверьте бит статуса, ассоциированный с %MSG2, и выдайте инструкцию EXCH2, если порт готов. Для инструкции EXCH2 определено значение 8 слов: два управляющих слова (%MW10 и %MW11), два слова для передачи информации (%MW12 и %MW13) и четыре слова для получения данных (с %MW14 по %MW17).
Наконец, статус ошибки %MSG2 считывается и хранится в первом выходном бите локальных базовых вх/вых контроллера. Также можно добавить проверку ошибок при помощи %SW64.

Шаг 5: Инициализировать Редактор анимационных таблиц:

Address	Current	Retained	Format
1	%MW10	0104	Hexadecimal
2	%MW11	0000	Hexadecimal
3	%MW12	4F4B	Hexadecimal
4	%MW13	0A0D	Hexadecimal
5	%MW14	TW	ASCII
6	%MW15	ID	ASCII
7	%MW16	O	ASCII
8	%MW17	A	ASCII

В последнем шаге приложение загружается в контроллер и запускается. Инициализируйте Редактор анимационных таблиц, чтобы анимировать и отображать слова с %MW10 по %MW17. В эмуляторе терминала отображаются символы "O"-"K"-CR-LF. Символы "O"-"K"-CR-LF могут быть отображены столько раз, сколько раз заканчивается тайм-аут ответа блока EXCH и запускается новый блок EXCH. В эмуляторе терминала напечатайте "T"-"W"-"I"-"D"-"O"-" "-"A". Эти данные обмениваются с контроллером Twido и отображаются в Редакторе анимационных таблиц.

Коммуникации Modbus

Введение

Протокол Modbus является master/slave протоколом, который позволяет одному и только одному главному (master) контроллеру запрашивать ответы от подчиненных (slave) или действовать на основании запросов. Главный контроллер может адресовать послания подчиненным контроллерам индивидуально или может инициировать широковещательную передачу всем подчиненным контроллерам. Подчиненные контроллеры возвращают отклики только на послания, которые адресовались им индивидуально. Ответы на широковещательные запросы от главного контроллера не посылаются.

Аппаратная конфигурация

Связь Modbus может быть установлена через порт EIA RS-232 или порт EIA RS-485 и выполняться одновременно на двух портах. Каждому порту может быть присвоен свой адрес Modbus, используя системный бит %S101 (См. *SCистемные биты (%S), стр. 434*)
В следующей таблице перечислены используемые устройства::

Удаленное устр-во	Порт	Спецификации
TWDLCAA10/16/24DRF, TWDLMDA20/40DUK, TWDLMDA20/40DTK, TWDLMDA20DRT	1	Базовый контроллер, оборудованный 3-х проводным EIA RS-485 портом с коннектором miniDIN.
TWDNOZ232D	2	Коммуникационный модуль, оборудованный 3-х проводным EIA RS-232 портом с коннектором miniDIN. **Примечание:** Этот модуль доступен только для модульных контроллеров. Когда этот модуль установлен, у контроллера не может быть модуля дисплея.
TWDNOZ485D	2	Коммуникационный модуль, оборудованный 3-х проводным EIA RS-485 портом с коннектором miniDIN. **Примечание:** Этот модуль доступен только для модульных контроллеров. Когда этот модуль установлен, у контроллера не может быть модуля дисплея.
TWDNOZ485T	2	Коммуникационный модуль, оборудованный 3-х проводным EIA RS-485 портом с терминалом. **Примечание:** Этот модуль доступен только для модульных контроллеров. Когда этот модуль установлен, у контроллера не может быть модуля дисплея.

Удаленное устр-во	Порт	Спецификации
TWDNAC232D	2	Коммуникационный адаптер, оборудованный 3-х проводным EIA RS-232 портом с коннектором miniDIN. **Примечание: Этот адаптер доступен только для компактных контроллеров Compact 16 и Compact 24 и модуля дисплея.**
TWDNAC485D	2	Коммуникационный адаптер, оборудованный 3-х проводным EIA RS-485 портом с коннектором miniDIN. **Примечание:** Этот адаптер доступен только для компактных контроллеров Compact 16 и Compact 24 и модуля дисплея.
TWDNAC485T	2	Коммуникационный адаптер, оборудованный 3-х проводным EIA RS-485 портом с терминалом. **Примечание: Этот адаптер доступен только для компактных контроллеров Compact 16 и Compact 24 и модуля дисплея.**
TWDXCPODM	2	Дисплей, оборудованный 3-х проводным EIA RS-232 портом с коннектором miniDIN, 3-х проводным EIA RS-485 портом с коннектором miniDIN и 3-х проводным EIA RS-485 портом с терминалом. **Примечание:** Этот модуль доступен только для модульных контроллеров. Когда этот модуль установлен, у контроллера не может быть коммуникационного модуля расширения.

Примечание: Вы можете только проверить конфигурацию порта 2 (доступность и тип) при включении или сбросить состояние при помощи встроенной программы.

Номинальные соединения кабелей

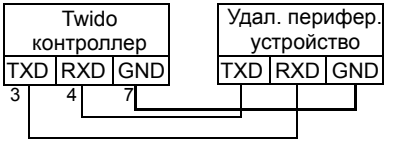
Номинальные соединения кабелей типов EIA RS-232 и EIA RS-485 изображены на следующем рисунке.

Примечание: Если порт 1 контроллера используется, сигнал DPT на выводе 5 должен быть привязан к 0V на выводе 7. Это обозначает, что связь через порт 1 это Modbus, а не протокол для связи с TwidoSoft.

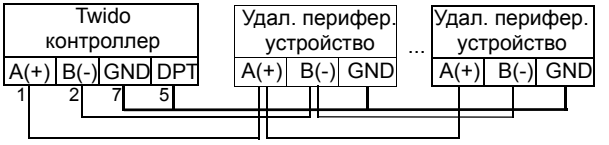
На рисунке показано соединения кабеля с каждым устройством.

Соединение Mini-DIN

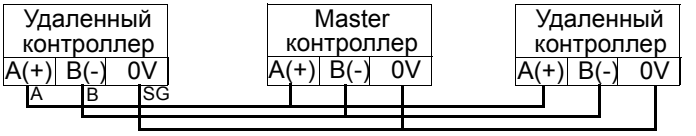
RS-232 EIA cable



RS-485 EIA cable



Соединение с терминальным блоком



Программная конфигурация

Для того, чтобы настроить контроллер для использования последовательного порта для отправки и получения символов при помощи протокола Modbus, Вы должны:

Шаг	Описание
1	Сконфигурировать последовательный порт для Modbus при помощи TwidoSoft.
2	Создать в приложении таблицу отправки/получения, которая будет использоваться инструкцией EXCHx.

Конфигурирование порта

Контроллер Twido может использовать основной порт 1 или опциональный порт 2 для протокола Modbus. Чтобы сконфигурировать последовательный порт для Modbus:

Шаг	Действие
1	Определите любые дополнительные коммуникационные адаптеры или модули, сконфигурированные для базы.
2	Щелкните правой кнпкой на порте, щелкните “ Edit Controller Comm Setup...” и измените тип последовательного порта на "Modbus".
3	Установите коммуникационные параметры.

Modbus Master

Режим modbus master позволяет контроллеру инициировать передачу запроса с ожиданием отклика от подчиненного контроллера. Этот режим поддерживается только инструкцией "EXCH".Поддерживаются Modbus ASCII и дистанционный терминал (RTU).
Максимальный размер переданных и/или полученных фреймов равен 256 байтам. таблица слов, связанная с инструкцией EXCHx состоит из управляющей таблицы, таблицы отправки и таблицы получения.

	Старший значимый байт	**Младший значимый байт**
Управляющая таблица	Команда	Длина (отправка/получение)
	Смещение получения	Смещение отправки
Таблица отправки	Отправленный байт 1	Отправленный байт 2
	...	...
	...	Отправленный байт n
	Отправленный байт n+1	
Таблица получения	Полученный байт 1	Полученный байт 2
	...	...
	...	Полученный байт p
	Полученный байт p+1	

Управляющая таблица

Байт **Длина** содержит длину таблицы пересылки (макс 250 байт). В него записывается число полученных символов в конце получения, если получение требуется.
Этот параметр - длина таблицы пересылки. Если параметр смещения Tx равен 0, этот параметр равен длине фрейма передачи. Если параметр смещения Tx не равен 0, один байт из таблицы передачи (указанный величиной смещения) не будет передан и этот параметр равен длине самого фрейма плюс 1.
Байт **Команда** в случае запроса RTU (за исключением широковещания) должен всегда быть равен 1.
Байт **Смещение Tx** содержит ранг (1 для первого байта, 2 для второго байта и т. д.) в таблице пересылки байта, который надо проигнорировать при пересылке байтов. Это используется для обработки расхождений значений байтов/слов в протоколе Modbus. Например, если этот байт содержит 3, третий байт будет проигнорирован, четвертый байт таблицы станет третьим байтом для пересылки.
Байт **Смещение Tx** содержит ранг (1 для первого байта, 2 для второго байта и т. д.) в таблице приема байта, который надо добавить при пересылке пакета. Это используется для обработки расхождений значений байтов/слов в протоколе Modbus. Например, если этот байт содержит 3, в третий байт таблицы запишется 0, а третий принятый байт запишется в четвертый байт в таблице.

Таблицы отправки/ получения

При использовании любого режима (Modbus ASCII или Modbus RTU), таблица пересылки заполняется после выполнения инструкции EXCHx. Во время выполнения, чем является канальный уровень модели OSI и выполняет все необходимые преобразования для выполнения передачи и отклика. Стартовые, конечные и проверочные символы не хранятся в таблицах отправки/получения.
После того, как переданы все байты, контроллер переключается в режим приема и ожидает получения байтов.
Получение заканчивается одним из следующих способов:

- обнаружен тайм-аут символа или фрейма,
- получен символ “конец кадра” в режиме ASCII,
- таблица получения переполнена.

Отправленный байт X содержит данные протокола Modbus (кодировка RTU), которые необходимо передать. Если коммуникационный порт сконфигурирован для Modbus ASCII, к передаче присоединяются символы формирования фрейма. Первый байт содержит адрес устройства (отдельный или широковещательный), второй байт содержит код функции, остальные содержат информацию, связанную с кодом функции.

Примечание: Это типичное применение и не раскрывает всех возможностей. Не будет произведено никакой проверки переданных данных.

Полученный байт X содержит данные протокола Modbus (кодировка RTU), которые необходимо принять. Если коммуникационный порт сконфигурирован для Modbus ASCII, из ответа изымаются символы формирования фрейма. Первый байт содержит адрес устройства (отдельный или широковещательный), второй байт содержит код функции, остальные содержат информацию, связанную с кодом функции.

Примечание: Это типичное применение и не раскрывает всех возможностей. Не будет произведено никакой проверки полученных данных, кроме проверки контрольной суммы.

Modbus Slave

Режим modbus slave позволяет контроллеру отвечать на стандартные запросы Modbus от modbus master.
Когда к контроллеру присоединен кабель TSX PCX1031, устанавливается соединение с TwidoSoft через порт, временно запрещающее текущий режим связи.
Протокол Modbus поддерживает два формата канального уровня : ASCII и RTU. Каждый определяется реализацией физического уровня, ASCII использует 7 битов данных, а RTU использует 8 битов данных.
При использовании режима Modbus ASCII, каждый байт сообщения посылается как два символа ASCII. Фрейм Modbus ASCII начинается стартовым символом(':') и может закончиться двумя конечными символами (CR и LF).Символ конца фрейма равен по умолчанию 0x0A (перевод строки), пользователь может изменить значение этого байта при конфигурации.
Контрольное значение для фрейма Modbus ASCII это дополнительный код фрейма, исключая начальный и конечные символы.
Режим Modbus RTU не переформатирует сообщение перед отправкой; однако, он использует другой способ вычисления контрольной суммы CRC(контроль с помощью циклического избыточного кода).
Канальный уровень Modbus имеет следующие ограничения:

- Адрес 1-247
- Биты: 128 бит по требованию
- Слова: 125 слов из 16 бит по требованию

Обмен сообщениями

Язык предоставляет две службы обмена сообщениями:

- **Инструкция EXCHx : для отправки/получения сообщений**
- **Функциональный блок %MSGx:** для контроля обмена.

Контроллер Twido использует протокол порта при выполнении инструкции EXCHx.

Примечание: Каждый коммуникационный порт может быть сконфигурирован для разных или одинаковых протоколов. Инструкция EXCHx или функциональный блок %MSGx для каждого коммуникационного порта достигается прибавлением номера порта (1 или 2).

Инструкция EXCHx

Инструкция EXCHx позволяет контроллеру Twido посылать и/или получать информацию к/от устройств Modbus. Пользователь задает таблицу слов (%MWi:L), содержащую управляющую информацию или данные для отсылки и/или приема (до 128 байт в каждом направлении). Формат таблицы был описан ранее.
Обмен сообщениями происходит посредством инструкции EXCHx:

Синтаксис: [EXCHx %MWi:L] или [EXCHx %KWi:L]

где: x = номер порта (1 или 2)
L = число слов в таблицах управления, отсылки и получения

Контроллер Twido должен завершить обмен по первой инструкции EXCHx, прежде чем запускать следующую инструкцию. При отправке нескольких сообщений должен использоваться функциональный блок %MSGx. Выполнение инструкции EXCHx начинается немедленно, передача по прерыванию (получение данных тоже происходит по прерыванию) происходит в фоновом режиме.

Функциональный блок %MSGx

Использование функционального блока %MSGx является необязательным; он может использоваться для управления обменом данными. У функционального блока %MSGx есть три назначения:

- **Проверка ошибок коммуникаций**
  Проверка ошибок контролирует, что параметр L (длина таблицы слов) заданный в инструкции EXCHx, достаточен для хранения длины передаваемых сообщений. Это значение сравнивается с длиной младшего значимого байта первого слова в таблице слов.
- **Согласование передачи нескольких сообщений**
  Для обеспечения координации при посылке нескольких сообщений, функциональный блок %MSGx обеспечивает информацию, требуемую, чтобы определить, что передача предыдущего сообщения завершена.
- **Передача приоритетных сообщений**
  Функциональный блок %MSGx позволяет остановить текущую передачу сообщений, чтобы позволить немедленную передачу срочного сообщения.

Функциональный блок %MSGx имеет один вход и два выхода:

Вход/Выход	Определение	Описание
R	Вход сброса	Установлен в 1: реинициализация связи или сброс блока (%MSGx.E = 0 и %MSGx.D = 1).
%MSGx.D	Связь завершена	0: происходит запрос. 1: связь завершена, если передача закончилась, получен символ завершения, ошибка или сброс блока.
%MSGx.E	Ошибка	0: длина сообщения и связь в порядке. 1: неправильная команда, или конфигурация таблицы, получен неправильный символ (скорость, четность и т. д.) или переполнение таблицы получения.

Ограничения

Важно отметить следующие ограничения:

- Доступность и тип порта 2 (см %SW7) проверяется при пуске и перезагрузке
- Все сообщения через порт 1прерываются, когда подключается TwidoSoft
- Нельзя выполнить EXCHx или %MSG на порте дистанционной связи
- EXCHx прерывает активное выполнение Modbus Slave
- Выполнение инструкции EXCHx не повторяется в случае ошибки
- Вход сброса (R) может использоваться для сброса выполнения инструкции получения EXCHx
- Инструкция EXCHx может конфигурироваться с тайм-аутом для прекращения получения
- Множественные сообщения контролируются через %MSGx.D

Ошибка и условия рабочего режима

Если во время выполнения инструкции EXCHx происходит ошибка, биты %MSGx.D и %MSGx.E устанавливаются в 1, системное слово %SW63 содержит код ошибки для порта 1, системное слово %SW64 содержит код ошибки для порта 2.

Систем-ные слова	Использование
%SW63	Код ошибки EXCH1: 0 - выполнение было успешно 1 – число байтов для пересылки слишком большое (> 250) 2 - таблица пересылки слишком маленькая 3 - таблица слов слишком маленькая 4 - переполнение таблицы получения 5 - тайм-аут 6 - ошибка пересылки 7 - неправильная команда в таблице 8 - выбранный порт не сконфигурирован/не доступен 9 - ошибка получения (только режим ASCII) 10 - при получении невозможно использовать%KW 11 - смещение пересылки больше, чем таблицы пересылки 12 - смещение получения больше, чем таблицы получения 13 - контроллер остановил выполнение EXCH
%SW64	Код ошибки EXCH2: см. %SW63.

Перезапуск master контроллера

При перезапуске master/slave контроллера происходит одно из следующих событий:

- “Холодный” старт (%S0 = 1) приводит к реинициализации коммуникаций.
- “Теплый” старт (%S1 = 1) приводит к реинициализации коммуникаций.
- В режиме ОСТАНОВКА контроллер останавливает все коммуникации ASCII.

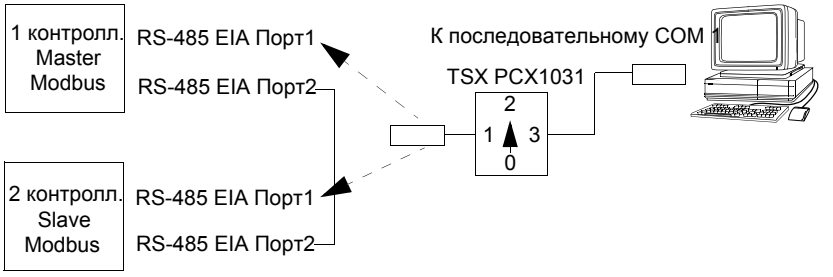
Связь Modbus. Пример 1.

Для конфигурирования связи Modbus, Вы должны:

1. Сконфигурировать аппаратные средства.
2. Соединить коммуникационный кабель Modbus.
3. Сконфигурировать порт.
4. Написать приложение.
5. Инициализировать Редактор анимационных таблиц.

На следующем рисунке показано использование кода запроса 3 для чтения выходных слов. В этом примере используется два контроллера Twido.

Шаг 1: Сконфигурировать аппаратные средства:



Аппаратная конфигурация: два контроллера Twido. Один будет сконфигурирован как Modbus Master, другой как Modbus Slave.

Примечание: В этом примере каждый контроллер конфигурируется для использования EIA RS-485 порта 1 и опционального EIA RS-485 порта 2. Для модульного контроллера опциональный порт 2 может бытьTWDNOZ485D или TWDNOZ485T, или, если Вы используете TWDXCPODM, он может бытьTWDNAC485D или TWDNAC485T. Для компактного контроллера опциональный порт 2 может быть TWDNAC485D или TWDNAC485T.

Для конфигурирования каждого контроллера, подключите кабель TSX PCX1031 к первому порту контроллера.

Примечание: TSX PCX1031 может быть одновременно присоединен только к одному контроллеру только к порту 1 RS-485 EIA .

Затем присоедините кабель к порту COM 1 компьютера. Убедитесь, что переключатель в положении 2. Загрузите и отобразите приложение. Повторите процедуру для второго контроллера.

Шаг 2: Соединить коммуникационный кабель Modbus:
Соединение Mini-DIN

Соединение с терминальным блоком

Монтажная схема в этом примере представляет собой простое прямое соединение. Три сигнала A(+), B(-) и 0V монтируются в соответствии с рисунком. При использовании порта 1контроллера Twido, сигнал DPT (вывод 5) должен быть привязан к 0V (вывод 7). Это согласование DPT определяет, присоединен ли TwidoSoft. При привязке к земле контроллер использует конфигурацию порта, установленную в приложении, для определения типа связи.

Шаг 3: Конфигурация порта:

Hardware -> Add Option TWDNOZ485-
Hardware => Controller Comm. Setting Port: 2 Type: Modbus Address: 1 Baud Rate: 19200 Data: 8 Bit Parity: None Stop: 1 Bit End of Frame: 65 Response Timeout: 10 x 100 ms Frame Timeout: 10 ms

Hardware -> Add Option TWDNOZ485-
Hardware => Controller Comm. Setting Port: 2 Type: Modbus Address: 2 Baud Rate: 19200 Data: 8 Bit Parity: None Stop: 1 Bit End of Frame: 65 Response Timeout: 100 x 100 ms Frame Timeout: 10 ms

В обоих приложениях master и slave сконфигурированы опциональные порты EIA RS-485. Убедитесь, что что коммуникационные параметры контроллеров изменены для протокола Modbus и находятся по разным адресам.
В этом примере, для master адрес равен 1, для slave адрес равен 2. Число битов установлено в 8, что показывает использование режима Modbus RTU . Если бы число было установлено в 7, мы бы использовали режим Modbus-ASCII. Последнее изменение значения по умолчанию - увеличение тайм-аута ответа до 1 секунды.

Примечание: После выбора режима Modbus RTU, параметр "End of Frame"(конец фрейма) игнорируется.

Шаг 4: Напишите приложение:

```
LD 1
[%MW0 := 16#0106 ]
[%MW1 := 16#0300 ]
[%MW2 := 16#0203 ]
[%MW3 := 16#0000 ]
[%MW4 := 16#0004 ]
LD 1
AND %MSG2.D
[EXCH2 %MW0:11]
LD %MSG2.E
ST %Q0.0
END
```

```
LD 1
[%MW0 := 16#6566 ]
[%MW1 := 16#6768 ]
[%MW2 := 16#6970 ]
[%MW3 := 16#7172 ]
END
```

Прикладная программа написана используя TwidoSoft для master и slave. Для slave, мы просто записываем в некоторые слова памяти набор известных значений. Для master, таблица слов для инструкции EXCHx инициализируется для чтения 4 слов из slave по Modbus адресу 2, начиная с положения %MW0.

Приложение: Обратите внимание, что смещение RX установлено на %MW1 Modbus master. Смещение на 3 добавит байт (значение = 0) на третью позицию таблицы в области приема. Это выравнивает слова в master так, что они попадают в границы слов. Без этого смещения каждое слово данных разбивалось бы между двумя словами в блоке обмена. Это смещение используется для удобства.

Перед исполнением инструкции EXCH2, приложение проверяет коммуникационный бит, связанный с %MSG2. В конце, статус ошибки %MSG2 считывается и хранится в первом выходном бите локальных базовых вх/вых контроллера. Также можно добавить проверку ошибок при помощи %SW64.

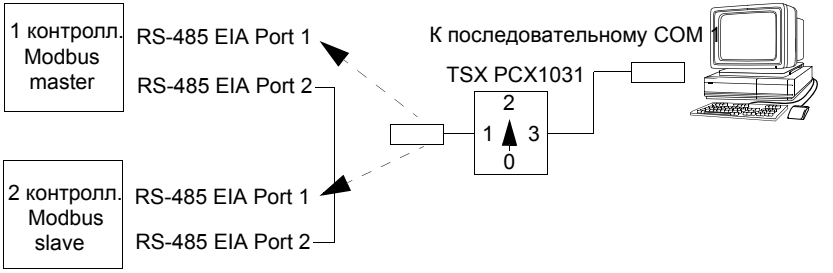
Шаг 5: Инициализировать Редактор анимационных таблиц на master:

	Address	Current	Retained	Format
1	%MW5	0203	0000	Hexadecimal
2	%MW6	0008	0000	Hexadecimal
3	%MW7	6566	0000	Hexadecimal
4	%MW8	6768	0000	Hexadecimal
5	%MW9	6970	0000	Hexadecimal
6	%MW10	7172	0000	Hexadecimal

После загрузки и настройки каждого контроллера на запуск, откройте анимационную таблицу на master. Проверьте в секции отклика таблицы, что код отклика = 3 и что было прочитано верное количество байтов. Также в этом примере обратите внимание, что слова, считанные с slave (начиная с %MW7) правильно выровнены по границам слов в master.

Связь Modbus. Пример 2.

На следующем рисунке показано использование запроса 16 Modbus на запись выходных слов в slave. В этом примере использовано 2 контроллера Twido.
Шаг 1: Сконфигурируйте аппаратные средства:



Аппаратная конфигурация идентична предыдущему примеру.
Шаг 2: Соедините коммуникационный кабель Modbus (RS-485):

Соединение Mini-DIN

Соединение с терминальным блоком

Соединение коммуникационных кабелей Modbus идентично предыдущему примеру.
Шаг 3: Конфигурация порта:

Hardware -> Add Option TWDNOZ485-
Hardware => Controller Comm. Setting Port: 2 Type: Modbus Address: 1 Baud Rate: 19200 Data: 8 Bit Parity: None Stop: 1 Bit End of Frame: 65 Response Timeout: 10 x 100 ms Frame Timeout: 10 ms

Hardware -> Add Option TWDNOZ485-
Hardware => Controller Comm. Setting Port: 2 Type: Modbus Address: 2 Baud Rate: 19200 Data: 8 Bit Parity: None Stop: 1 Bit End of Frame: 65 Response Timeout: 100 x 100 ms Frame Timeout: 10 ms

Конфигурации портов идентичны предыдущему примеру.
Шаг 4: Напишите приложение:

```
LD 1
[%MW0 := 16#010C ]
[%MW1 := 16#0007 ]
[%MW2 := 16#0210 ]
[%MW3 := 16#0010 ]
[%MW4 := 16#0002 ]
[%MW5 := 16#0004 ]
[%MW6 := 16#6566 ]
[%MW7 := 16#6768 ]
LD 1
AND %MSG2.D
[EXCH2 %MW0:11]
LD %MSG2.E
ST %Q0.0
END
```

```
LD 1
[%MW18 := 16#FFFF ]
END
```

Прикладная программа написана используя TwidoSoft для master и slave. Для slave, мы записываем одинарное слово памяти %MW18. Это выделит на slave свободное место для адресов памяти с %MW0 по %MW18. Без выделения памяти, запрос Modbus будет пытаться записать в несуществующее на slave место. Для master, таблица слов для инструкции EXCH2 инициализируется для чтения 4 слов в slave по Modbus адресу 2 в адрес %MW16 (10 шестнадцатеричное).

Приложение: Обратите внимание, что смещение TX установлено на %MW1 приложения Modbus master. Смещение на 7 удалит старший байт в шестом слове (шестнадцатеричное значение 00 в %MW5). Это выравнивает слова так, что они попадают в границы слов.

Перед исполнением инструкции EXCH2, приложение проверяет коммуникационный бит, связанный с %MSG2. В конце, статус ошибки %MSG2 считывается и хранится в первом выходном бите локальных базовых вх/вых контроллера. Также можно добавить проверку ошибок при помощи %SW64.

Шаг 5: Инициализируйте Редактор анимационных таблиц:

Создайте следующую анимационную таблицу на master:

Address	Current	Retained	Format
1 %MW0	010C	0000	Hexadecimal
2 %MW1	0007	0000	Hexadecimal
3 %MW2	0210	0000	Hexadecimal
4 %MW3	0010	0000	Hexadecimal
5 %MW4	0002	0000	Hexadecimal
6 %MW5	0004	0000	Hexadecimal
7 %MW6	6566	0000	Hexadecimal
8 %MW7	6768	0000	Hexadecimal
9 %MW8	0210	0000	Hexadecimal
10 %MW9	0010	0000	Hexadecimal
11 %MW10	0004	0000	Hexadecimal

Создайте следующую анимационную таблицу на slave:

Address	Current	Retained	Format
1 %MW16	6566	0000	Hexadecimal
2 %MW17	6768	0000	Hexadecimal

После загрузки и настройки каждого контроллера на запуск, откройте анимационную таблицу на master. Два значения в %MW16 и %MW17 записаны в slave. На master анимационная таблица может использоваться для проверки таблицы получения. Данные обмена отображают адрес slave, код отклика, первое записанное слово, число слов, записанных с %MW8 в примере.

Стандартные запросы Modbus

Введение

Эти запросы используются для обмена словами или битами памяти между удаленными устройствами. Формат таблицы одинаков для режимов RTU и ASCII.

Формат	**Шифр**
Бит	%Mi
Слово	%MWi

Modbus Master: Чтение N битов

В следующей таблице представлены запросы 01 и 02.

	Индекс в табли-це	**Старший значащий байт**	**Младший значащий байт**
Управляющая таблица	0	01 (Отправка/прием)	06 (Длина отправки) (*)
	1	00 (Смещение приема)	00 (Смещение отправки)
Таблица отправки	2	Slave@(1..247)	01 or 02 (Код запроса)
	3	Номер первого бита для чтения	
	4	N = Число битов для чтения	
Таблица приема (после отклика)	5	Slave@(1..247)	01 (Код отклика)
	6	Число переданных байтов (1 байт на бит)	
	7	Первый прочитанный байт (значение = 00 или 01)	Второй прочитанный байт (если N>1)
	8	Третий прочитанный байт (если N>1)	
	...		
	(N/2)+6	N-й прочитанный байт (если N>1)	

(*) В этот байт также записывается длина переданной после отклика строки.

Modbus Master: Чтение N слов

В следующей таблице представлены запросы 03 и 04.

	Индекс в табли-це	**Старший значащий байт**	**Младший значащий байт**
Управляющая таблица	0	01 (Отправка/прием)	06 (Длина отправки) (*)
	1	03(Смещение приема)	00 (Смещение отправки)
Таблица отправки	2	Slave@(1..247)	03 или 04 (Код запроса)
	3	Номер первого слова для чтения	
	4	N = Число слов для чтения	
Таблица приема (после отклика)	5	Slave@(1..247)	03 (Код отклика)
	6	00 (байт добавленный действием смещения Rx)	2*N (число байтов для чтения)
	7	Первое прочитанное слово	
	8	Второе прочитанное слово(если N>1)	
	...		
	N+6	N-е прочитанное слово (если N>2)	

(*) В этот байт также записывается длина переданной после отклика строки.

Примечание: Смещение Rx на 3 добавит байт (значение = 0) на третью позицию таблицы в области приема. Это обеспечивает правильное положение числа прочитанных байтов и значений слов в таблице.

Modbus Master: Запись бита

В таблице представлен запрос 05.

	Индекс в таблице	Старший значащий байт	Младший значащий байт
Управляющая таблица	0	01 (Отправка/прием)	06 (Длина отправки) (*)
	1	00 (Смещение приема)	00 (Смещение отправки)
Таблица отправки	2	Slave@(1..247)	05 (Код запроса)
	3	Номер бита для записи	
	4	Значение бита для записи	
Таблица приема (после отклика)	5	Slave@(1..247)	05 (Код отклика)
	6	Номер записанного бита	
	7	Записанное значение	

(*) В этот байт также записывается длина переданной после отклика строки.

Примечание:
- Для этого запроса не нужно использовать смещение.
- Фрейм ответа такой же, как фрейм запроса (в нормальном случае).
- Для записи бита=1,связанное слово в таблице пересылки должно содержать значение FF00H, и 0 для записи бита=0.

Modbus Master: Запись слова

В таблице представлен запрос 06.

	Индекс в таблице	**Старший значащий байт**	**Младший значащий байт**
Управляющая таблица	0	01 (Отправка/прием)	06 (Длина отправки) (*)
	1	00 (Смещение приема)	00 (Смещение отправки)
Таблица отправки	2	Slave@(1..247)	05 (Код запроса)
	3	Номер слова для записи	
	4	Значение слова для записи	
Таблица приема (после отклика)	5	Slave@(1..247)	05 (Код отклика)
	6	Номер записанного слова	
	7	Записанное значение	

(*) В этот байт также записывается длина переданной после отклика строки

Примечание:
- Для этого запроса не нужно использовать смещение.
- Фрейм ответа такой же, как фрейм запроса (в нормальном случае).

Modbus Master: Запись N битов

В таблице представлен запрос 15.

	Индекс в таблице	**Старший значащий байт**	**Младший значащий байт**
Управляющая таблица	0	01 (Отправка/прием)	8 + число байтов (отправка)
	1	00 (Смещение приема)	07 (Смещение отправки)
Таблица отправки	2	Slave@(1..247)	15 (Код запроса)
	3	Номер первого бита для записи	
	4	N_1 = Число битов для записи	
	5	00 (байт не посылается, эффект смешения)	N_2 =Число байтов данных для записи
	6	Значение второго байта	Значение второго байта
Управляющая таблица	7	Значение третьего байта	Значение четвертого байта
	...		
Таблица отправки	6+(N_2/2)	Значение N_2-го байта	
Таблица приема (после отклика)		Slave@(1..247)	15 (Код отклика)
		Номер первого записанного байта	
		Число записанных байтов (= N_1)	

Примечание:
- Смещение Tx на 7 удалит старший байт в посланном фрейме. Это обеспечивает соответствие значений слов в таблице пересылки.

Modbus Master: Запись N слов

В таблице представлен запрос 16.

	Индекс в таблице	**Старший значащий байт**	**Младший значащий байт**
Управляющая таблица	0	01 (Отправка/прием)	8 + (2*N) (длина отправки)
	1	00 (Смещение приема)	07 (Смещение отправки)
Таблица отправки	2	Slave@(1..247)	16 (Код запроса)
	3	Номер первого слова для записи	
	4	N = Число слов для записи	
	5	00 (байт не посылается, эффект смешения)	2*N =Число байтов данных для записи
	6	Значение первого слова для записи	
	7	Значение второго слова для записи	
	...		
	N+5	Значение N для записи	
Таблица приема (после отклика)	N+6	Slave@(1..247)	16 (Код отклика)
	N+7	Номер первого записанного слова	
	N+8	Число записанных слов(= N)	

Примечание: Смещение Tx = 7 удалит пятый MMSB байт в посланном фрейме. Это обеспечивает соответствие значений слов в таблице пересылки.

Встроенные аналоговые функции

7

Обзор

Предмет

В главе описано, как управлять встроенным аналоговым каналом и потенциометрами.

Содержание главы

Глава содержит следующие темы:

Аналоговый потенциометр

Введение

У контроллеров Twido есть:
- аналоговый потенциометр у контроллеров TWDLCAA10DRF, TWDLCAA16DRF и у всех модульных контроллеров (TWDLMDA20DTK, TWDLMDA20DUK, TWDLMDA20DRT, TWDLMDA40DTK и TWDLMDA40DUK,
- два потенциометра у контроллера TWDLCAA24DRF.

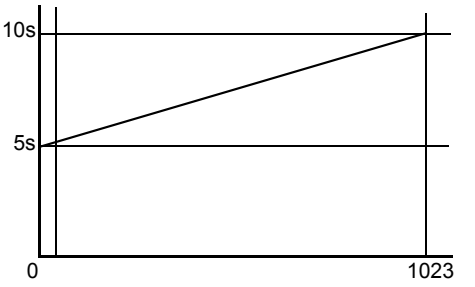
Программирование

Численные значения, от 0 до 1023 для аналогового потенциометра 1, и от 0 до 511 для аналогового потенциометра 2, соответствующие аналоговым значениям, представленным этими потенциометрами, содержатся в двух входных словах:
- %IW0.0.0 для аналогового потенциометра 1 (слева)
- %IW0.0.1 для аналогового потенциометра 2 (справа)

Эти слова можно использовать в арифметических операциях. Они могут использоваться для всех типов регулировки, например, при предустановке задержки времени или значения счетчика они согласовываются с частотой генератора импульсов или времени предварительного нагрева машины.

Пример

Корректировка продолжительности задержки с 5 до 10 сек., используя аналоговый потенциометр 1:

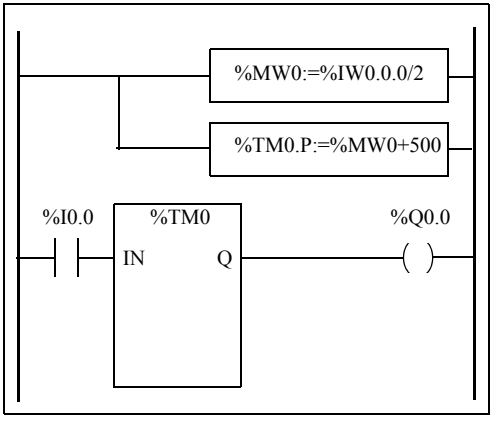
Для этой корректировки используется практически весь диапазон аналогового потенциометра 1 (с 1 по 1023) .	

Следующие параметры выбираются при конфигурации блока задержки %TM0:

- Type (Тип) TON
- Развертка: 10 мс

Предустановленное время задержки вычисляется в зависимости от корректирующего значения потенциометра по формуле %TM0.P := (%IW0.0.0/2)+500.

Код программы для рассмотренного выше примера:




```
LD      1
[%MW0:=%IW0.0.0/2]
[%TM0.P:=%MW0+500]
BLK     %TM0
LD      %I0.0
IN
OUT_BLK
LD      Q
ST      %Q0.0
END_BLK
.....
```

Аналоговый канал

Введение

Все модульные контроллеры (TWDLMDA20DTK, TWDLMDA20DUK, TWDLMDA20DRT, TWDLMDA40DTK и TWDLMDA40DUK)имеют встроенный аналоговый канал. Входное напряжение изменяется от 0 до 10 В, цифровой сигнал от 0 до 511. у аналогового канала есть преимущества по сравнению с простыми усредняющими схемами.

Принцип работы

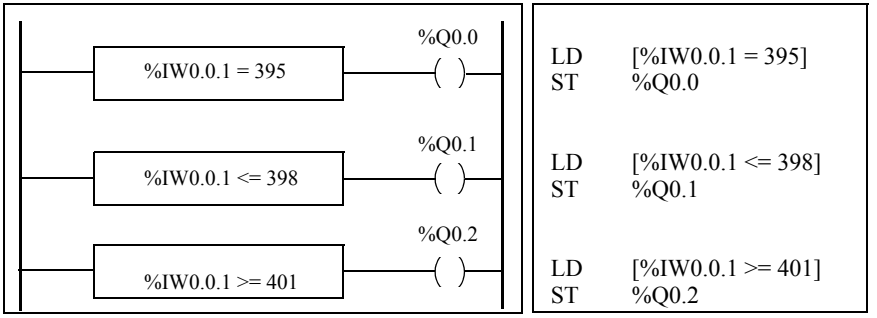
Аналогово-цифровой преобразователь конвертирует входное напряжение от 0 до 10 В в цифровое значение от 0 до 511. Это значение хранится в системном слове %IW0.0.1. Значение линейно на всем диапазоне, каждое увеличение приблизительно 20 мВ (10 В/512). Когда система обнаруживает значение 511, канал считается наполненным.

Пример программирования

Контролирование температуры духовки: Температура приготовления установлена в 350°C. Колебание +/- 2.5°C приводит к отключению выходов %Q0.0 и %Q0.2 соответственно. Практически все возможные диапазоны настройки аналогового канала от 0 до 511 используются в этом примере. Аналоговые настройки для точек температуры:

Температура (°C)	Напряжение	Системное слово %IW0.0.1
0	0	0
347.5	7.72	395
350	7.77	398
352.5	7.83	401
450	10	511

Код программы для рассмотренного выше примера:



Управление аналоговыми модулями

8

Обзор

Предмет

В этой главе представлен обзор управления аналоговыми модулями контроллеров Twido.

Содержание главы

Глава содержит следующие темы:

Обзор аналогового модуля

Введение

Кроме встроенного 10-битного потенциометра и 9-битного аналогового канала, все контроллеры Twido, которые поддерживают расширение вх/вых, могут также конфигурировать и связываться с модулями аналоговых вх/вых. Этими аналоговыми модулями являются:

Название	Выводы	Уровень сигнала	Кодирование
TWDAMI2HT	2 входа	0 - 10 В или 4 - 20 мА	12 бит
TWDAM01HT	1 выход	0 - 10 В или 4 - 20 мА	12 бит
TWDAMM3HT	2 входа, 1 выход	0 - 10 В или 4 - 20 мА	12 бит
TWDALM3LT	2 входа, 1 выход	0 - 10 В, Входы Th или PT100, выходы 4 - 20 мА	12 бит

Работа аналоговых модулей

Входные и выходные слова (%IW и %QW) используются для обмена данными между приложением пользователя и любым из аналоговых каналов. Обновление этих слов происходит синхронно со сканированием контроллера в режиме РАБОТА.

ОСТОРОЖНО

Неожиданное включение устройства

Когда контроллер в режиме ОСТАНОВКА, аналоговый вход устанавливается в аварийное состояние. Как и в случае с цифровым входом, аварийное состояние - ноль.

Несоблюдение этого правила может привести к ущербу или повреждению оборудования.

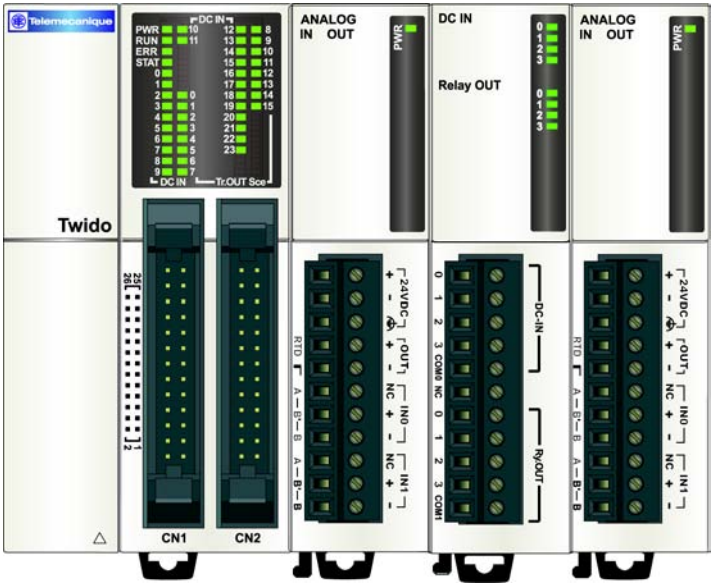
Адресация аналоговых входов и выходов

Введение

Адреса присваиваются аналоговым каналам в зависимости от их положения на шине расширения.

Пример адресации аналоговых вх/вых

В этом примере TWDLMDA40DUK имеет встроенный аналоговый скорректированный 10-битный потенциометр, 9-битный встроенный аналоговый канал. На шине расширения находятся: аналоговый модуль TWDAMM3HT, модуль цифровых релейных вх/вых TWDDMM8DRT и второй аналоговый модуль TWDAMM3HT.



База Модуль 1 Модуль 2 Модуль 3

В таблице приведена подробная информация по адресации выходов.

Описание	База	Модуль 1	Модуль 2	Модуль 3
Потенциометр 1	%IW0.0.0			
Встр. аналоговый канал	%IW0.0.1			
Аналоговый вх. канал 1		%IW0.1.0		%IW0.3.0
Аналоговый вых. канал 2		%IW0.1.1		%IW0.3.1
Аналоговый вых. канал 1		%QW0.1.0		%QW0.3.0
Цифровые вх. каналы			%I0.2.0 - %I0.2.3	
Цифровые вых. каналы			%Q0.2.0 -%Q0.2.3	

Конфигурирование аналоговых входов и выходов

Введение

В этой части приведена информация по конфигурированию аналоговых входов и выходов.

Конфигурирование аналоговых вх/вых

Диалоговое окно конфигурирования модуля (Configure Module) используется для управления параметрами аналоговых модулей.

Примечание: Вы можете изменять параметры только в режиме offline, когда Вы не подключены к контроллеру.

Адреса присваиваются аналоговым каналам в зависимости от их положения на шине расширения. В целях программирования Вы можете также присвоить предварительно определенные символы, чтобы манипулировать данными в Вашем пользовательском приложении.
Вы можете сконфигурировать тип одинарного выходного канала для TWDAM01HT, TWDAMM3HT и TWDALM3LT:

- Не используется
- 0 - 10 В
- 4 – 20 мА

Вы можете сконфигурировать тип двух входных каналов для TWDAMI2HT и TWDAMM3HT:

- Не используется
- 0 - 10 В
- 4 – 20 мА

ОСТОРОЖНО

Повреждение оборудования

Если Вы смонтировали вход для измерения напряжения и сконфигурировали TwidoSoft для тока, Вы можете навсегда повредить аналоговый модуль. Убедитесь, что монтажные схемы находятся в соответствии с конфигурацией TwidoSoft.

Несоблюдение этого правила может привести к ущербу или повреждению оборудования.

Вы можете сконфигурировать тип двух входных каналов для TWDALM3LT :

- Не используется
- Термопара K
- Термопара J
- Термопара T
- PT 100

При конфигурировании канала Вы можете выбрать присваиваемые единицы и установить диапазон входов в соответствии с таблицей:

Диапазон	**Единицы**	**Описание**
Нормальный	Нет	Фиксированный диапазон от 0 до 4095.
Пользовательский	Нет	Определяется пользователем с минимумом не меньше -32768 и максимумом не больше 32767.
Цельсий	0.1ЧС	Международная шкала температур. Доступна только для входных каналов TWDALM3LT.
Фаренгейт	0.1ЧF	Шкала температур, где точка закипания воды 212ЧF (100ЧС) и точка замерзания 32ЧF (0ЧС). Доступна только для входных каналов TWDALM3LT.

Информация о состоянии аналогового модуля

Таблица состояния

В следующей таблице содержится информация, необходимая для контролирования состояния модулей аналоговых вх/вых.

Системное слово	Функция	Описание
%SW80	Статус базовых вх/вых	Бит [0] Каналы работают нормально (все каналы) Бит [1] Модуль инициализируется (или инициализация информации всех каналов) Бит [2] Сбой аппаратных средств (сбой внешнего источника питания, общий для всех каналов) Бит [3] Ошибка конфигурации модуля Бит [4] Происходит конвертирование данных входного канала 0 Бит [5] Происходит конвертирование данных входного канала 1 Бит [6] Входная термопара канала 0 не сконфигурирована Бит [7] Входная термопара канала 1 не сконфигурирована Бит [8] Не используется Бит [9] Не используется Бит [10] Превышение диапазона аналоговых входных данных канала 0 Бит [11] Превышение диапазона аналоговых входных данных канала 1 Бит [12] Неправильный монтаж (аналоговые входные данные канала 0 меньше минимального допустимого значения, открыта токовая петля) Бит [13] Неправильный монтаж (аналоговые входные данные канала 1 меньше минимального допустимого значения, открыта токовая петля) Бит [14] Не используется Бит [15] Выходной канал недоступен
%SW81	Статус модуля расширения вх/вых 1: описание аналогично %SW80	
%SW82	Статус модуля расширения вх/вых 2: описание аналогично %SW80	
%SW83	Статус модуля расширения вх/вых 3: описание аналогично %SW80	
%SW84	Статус модуля расширения вх/вых 4: описание аналогично %SW80	
%SW85	Статус модуля расширения вх/вых 5: описание аналогично %SW80	
%SW86	Статус модуля расширения вх/вых 6: описание аналогично %SW80	
%SW87	Статус модуля расширения вх/вых 7: описание аналогично %SW80	

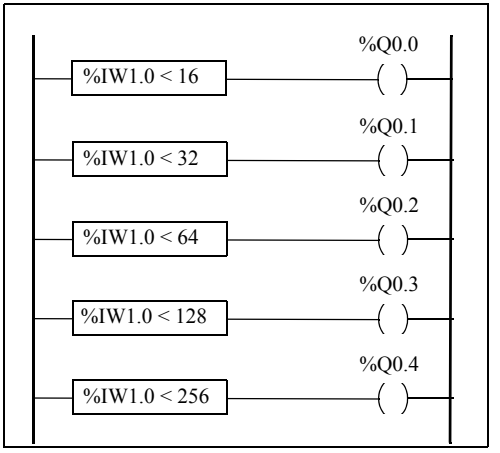
Пример использования аналоговых модулей

Введение

В этой части представлен пример использования аналоговых модулей, доступных для Twido.

Пример: аналоговый вход

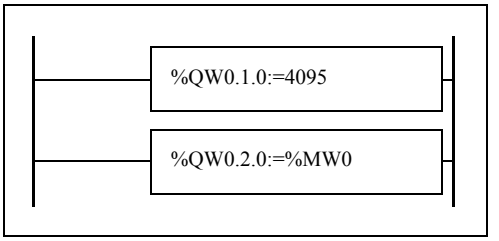
В этом примере аналоговый входной сигнал сравнивается с пятью отдельными пороговыми значениями. Производится сравнение аналогового входа и бит на базовом контроллере устанавливается, если значение меньше или равно пороговому.




```
LD [%IW1.0 < 16]
ST %Q0.0

LD [%IW1.0 < 32]
ST %Q0.1

LD [%IW1.0 < 64]
ST %Q0.2

LD [%IW1.0 < 128]
ST %Q0.3

LD [%IW1.0 < 256]
ST %Q0.4
```

Пример: аналоговый выход

Следующая программа использует аналоговую карту в слоте 1 и 2. Карта в слоте 1 имеет 10-вольтный выход с “нормальным” диапазоном:




```
LD   1
[%QW0.1.0:=4095
LD   1
[%QW0.2.0:=%MW0
```

- Пример выходных значений для %QW1.0=4095 (нормальный случай):

В следующей таблице показано значение выходного напряжения, соответствующее максимальному значению, присвоенному %QW1.0:

	цифровое значение	**аналоговое значение (В)**
Минимум	0	0
Максимум	4095	10
Значение 1	100	0.244
Значение 2	2460	6

- Пример выходных значений для диапазона пользователя (минимум = 0, максимум = 1000):

В следующей таблице показано значение выходного напряжения, соответствующее максимальному значению, присвоенному %QW1.0:

	цифровое значение	**аналоговое значение (В)**
Минимум	0	0
Максимум	1000	10
Значение 1	100	1
Значение 2	600	6

Установка шины AS-Interface V2

9

Обзор

Предмет

В этой главе представлена информация по программной инсталляции главного (Master) модуля TWDNOI10M3 и подчиненных модулей AS-Interface.

Содержание главы

Глава содержит следующие темы:

Представление шины AS-Interface V2

Введение

Шина AS-Interface (интерфейс сенсоров и исполнительных устройств) позволяет соединение одиночного кабеля сенсорных устройств/ исполнительных механизмов на нижнем уровне автоматизации.
Эти сенсоры/исполнительные устройства будут обозначаться в документе, как подчиненные устройства (**slave)**.

Для выполнения приложения AS-Interface Вам необходимо определить физический контекст приложения, в которое оно будет интегрироваться (шина расширения, питание, процессор, модули, подчиненные устройства AS-Interface, подсоединенные к шине) и обеспечить программную интеграцию.

Второй пункт будет осуществлен из различных редакторов TwidoSoft:
- либо в локальном режиме,
- либо в оперативном режиме (online).

Шина AS-Interface V2

Главный модуль **TWDNOI10M3** AS-Interface имеет следующие функциональные возможности:
- M3 профиль: Этот профиль включает все функциональные возможности, определенные стандартом AS-Interface V2, но не поддерживает аналоговые профили S7-4
- Один канал AS-Interface на модуль
- Автоматическая адресация slave с адресом 0
- Управление профилями и параметрами
- Защита от противоположной полярности на входах шины.

Шина AS-Interface позволяет:
- До 31slave со стандартным адресом и до 62 с расширенным адресом
- До 248 входов и 186 выходов
- До 7 аналоговых slave (максимум четыре вх/вых на slave)
- Максимальную продолжительность цикла 10 мс

Максимум два главных модуля AS-Interface могут быть подсоединены к модульному контроллеру Twido или компактному контролеру LCAA24DRF.

Общее функциональное описание

Общее введение

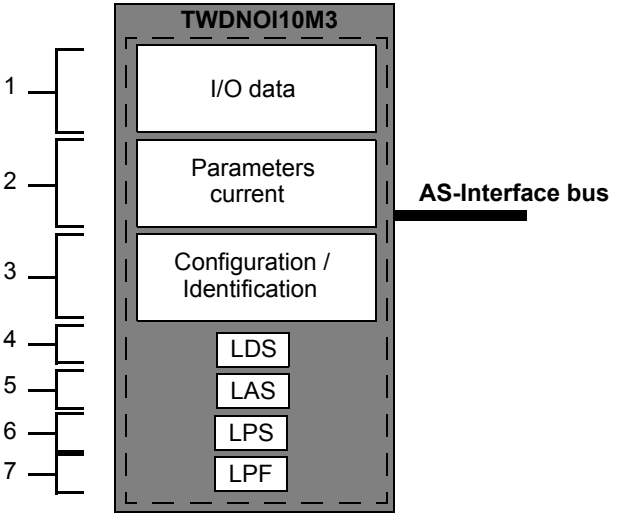
Для конфигурации AS-Interface, TwidoSoft позволяет пользователю:

- Вручную конфигурировать шину (декларацию slave и присваивание адресов на шине)
- Адаптировать конфигурацию в соответствии с тем, что присутствует на шине
- Подтверждать параметры slave
- Контролировать статус шины

По этой причине все данные, поступающие из или отправляемые в главный модуль AS-Interface, хранятся в специальных объектах (словах или битах).

Структура Master AS-Interface

Модуль AS-Interface содержит поля данных, которые позволяют Вам управлять списками подчиненных устройств и отображениями входных/выходных данных. Информация хранится во временной памяти.
На следующем рисунке показана архитектура модуля **TWDNOI10M3**.



Пояснение:

Адрес	Элемент	Описание
1	I/O data (вх/вых данные) (IDI, ODI)	Отображения 248 входов и 186 выходов шины AS-Interface V2.
2	Current parameters (текущие параметры) (PI, PP)	Отображение параметров всех slave.
3	Configuration/ Identification (конфигурация/ идентификация) (CDI, PCD)	Это поле содержит все коды вх/вых и идентификационные коды всех определенных slave.
4	LDS	Список всех slave, определенных на шине.
5	LAS	Список всех slave, активированных на шине.
6	LPS	Список готовых slave на шине, сконфигурированных через TwidoSoft.
7	LPF	Список slave, имеющих ошибку устройства.

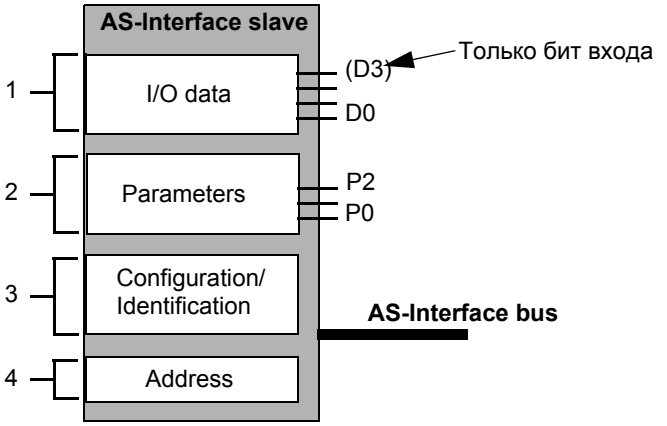
Структура подчиненных устройств

Каждый slave со стандартным адресом имеет:
- 4 бита вх/вых
- 4 бита параметров

Каждый slave с расширенным адресом имеет:
- 4 бита вх/вых (младший бит зарезервирован только для входа)
- 3 бита параметров

У каждого slave есть свой адрес, профиль и подпрофиль (определяет обмен переменными).

На следующем рисунке показана структура slave с расширенным адресом:



Пояснение:

Адрес	Элемент	Описание
1	Input/output data (вх/вых данные)	Входные данные хранятся slave и доступны для master AS-Interface. Выходные данные обновляются модулем master.
2	Parameters (параметры)	Параметры используются для контроля и переключения внутренних режимов работы на сенсор или исполняющее устройство.
3	Configuration/ Identification (конфигурация/ идентификация)	Это поле содержит: ● Код, соответствующий конфигурации вх/вых, ● Код идентификации (ID) slave, ● Коды идентификации (ID1 и ID2) slave.
4	Address (адрес)	Физический адрес slave.
Примечание: Рабочие параметры, адреса, конфигурационные и идентификационные данные сохраняются в постоянную память.		

Принципы установки программного обеспечения

Обзор

Для соблюдения философии, принятой в TwidoSoft, пользователь должен придерживаться пошагового подхода при создании приложения AS-Interface.

Принцип установки

Пользователь должен знать, как функционально конфигурировать его шину AS-Interface (См. *Как добавить подчиненное устройство в существующую конфигурацию AS-Interface V2, стр. 182*). В следующей таблице показаны различные фазы программы шины AS-Interface.

Режим	Фаза	Описание
Локальный	Объявление модуля	Выбор слота master модуля AS-Interface TWDNOI10M3 на шине расширения.
	Конфигурация канала модуля	Выбор режимов "master".
	Объявление slave устройств	Выбор для каждого устройства: ● номер слота на шине, ● тип расширенного или стандартного адреса slave.
	Подтверждение конфиг. параметров	Подтверждение на уровне slave.
	Общее подтверждение приложения	Подтверждение уровня приложения.
Локал. или подсоед.	Символизация (необяз.)	Символизация переменных, связанных с устройствами slave.
	Программирование	Программирование функций AS-Interface V2.
Подсоед.	Перенос	Перенос приложения на ПЛК.
	Отладка	Отладка программы с помощью: ● отладочного экрана, используемого для отображения slave (адрес, параметры) и для присвоения им требуемых адресов, ● диагностических экранов, позволяющих распознавание ошибок.

Примечание: Объявление и удаление master модуля AS-Interface на шине расширения такое же, как для других модулей расширения. Однако, после объявления двух master модулей AS-Interface на шине расширения, TwidoSoft не позволит объявить еще один.

Меры предосторожности перед соединением

Перед соединением (через программные средства) ПК с контроллером чтобы избежать проблем определения:

- Убедитесь, что нет slave, физически присутствующего на шине с адресом 0
- Убедитесь, что нет 2 slave, физически присутствующих на шине с одним адресом.

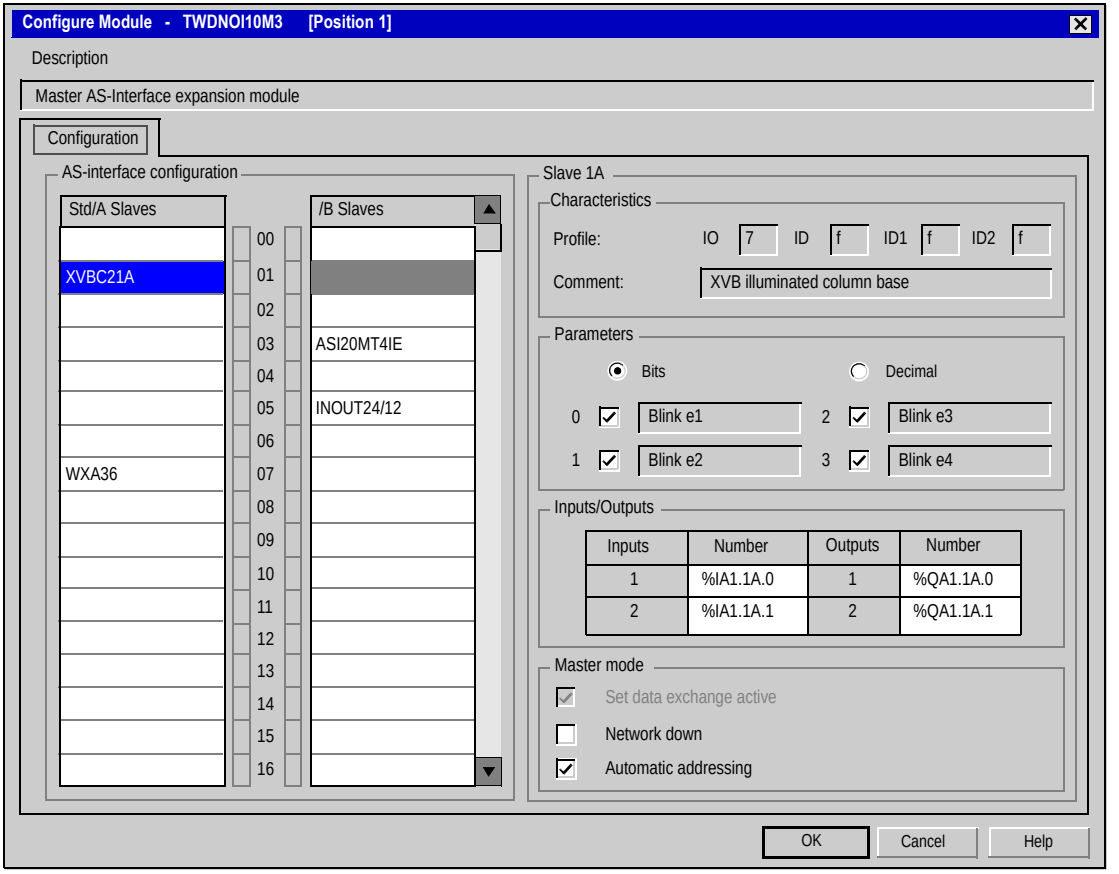
Описание экрана конфигурации для шины AS-Interface

Обзор

Экран конфигурации master модуля AS-Interface предоставляет доступ к параметрам, связанным с модулем и подчиненными устройствами. Он может использоваться для отображения и изменения параметров в режиме offline.

Иллюстрация режима Offline

Иллюстрация экрана конфигурации в режиме offline:



Описание экрана в режиме Offline

Этот экран группирует данные шины в три блока информации:

Блоки	Описание
AS-interface configuration (Конфигурация AS-interface)	Образ шины, желательный для пользователя: настройка ожидаемых на шине стандартных и расширенных адресов slave. Переместите курсор вниз по вертикальной полосе для доступа к следующим адресам. Серые адреса соответствуют адресам, не доступным здесь для конфигурации slave. Например, если объявлен новый slave со стандартным адресом 1A, адрес1B автоматически выделяется серым цветом.
Slave xxA/B	Конфигурация выбранного slave: ● Характеристики: IO код, ID код, ID1 и ID2 коды (профили) и комментарии для slave, ● Параметры: список параметров (модифицируемых), в двоичной (4 флага) или десятичной (1 флаг) форме, на усмотрение пользователя, ● Входы/выходы: список доступных вх/вых и их адреса.
Master mode (Режим master)	Активация или дезактивация возможна для двух функций, доступных для этого модуля AS-Interface (например, автоматическая адресация). Режим "Автоматическая адресация" выбран по умолчанию. Функция "Активация обмена данными" пока не доступна.

Этот экран также включает 3 кнопки:

Кнопки	Описание
OK	Используется для сохранения конфигурации шины AS-Interface, видимой на конфигурационном экране. Затем возврат на основной экран. Конфигурация может быть передана на контроллер Twido.
Cancel (Отмена)	Возврат на основной экран без подтверждения текущих изменений.
Help (Помощь)	Открывает окно справки.

Примечание: Изменения в экране конфигурации могут быть сделаны только в режиме offline.

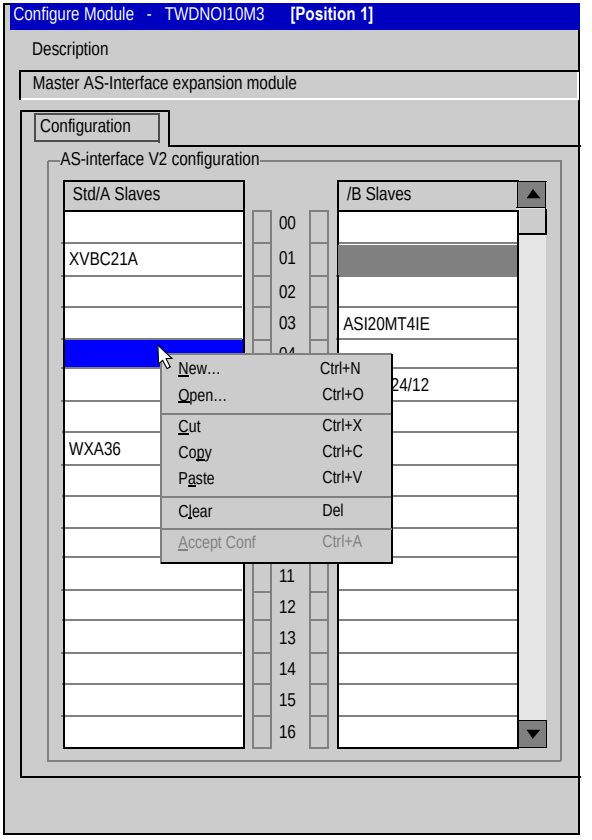
Конфигурация шины AS-Interface

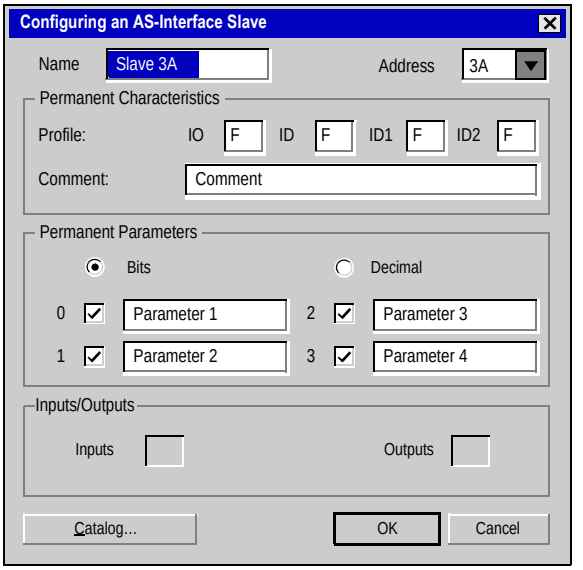
Введение

Конфигурация шины AS-Interface происходит в экране конфигурации в локальном режиме.
После выбора Master и его режимов AS-Interface, конфигурация шины AS-Interface состоит из конфигурирования подчиненных устройств.

Процедура для объявления и конфигурирования slave

Процедура для создания или изменения slave на шине AS-Interface V2:

Шаг	Действие
1	На желаемой ячейке адреса (не серой) на изображении шины: ● Двойной щелчок: доступ к шагу 3 ИЛИ ● Правый щелчок: Результат:  Примечание: Появляется меню быстрого вызова. Используется для: ● Конфигурирования новых slave на шине ● Изменения конфигурации требуемого slave ● Копирования (или Ctrl+C), вырезания (или Ctrl+X), вставки slave (или Ctrl+V) ● Удаления slave (или Del)

Шаг	Действие
2	В меню быстрого вызова выберите: ● "New" для создания нового slave: отображается экран конфигурации slave; поле "Address" показывает выбранный адрес, поля "Profile" установлены в F по умолчанию, все остальные поля на экране пустые. ● "Open" для создания нового slave или для изменения конфигурации выбранного slave. Для нового slave: отображается новый экран для конфигурирования slave, поле "Address" показывает выбранный адрес, поля "Profile" установлены в F по умолчанию, все остальные поля на экране пустые. Для изменения: отображается экран конфигурации slave с полями, содержащими предыдущие значения выбранного slave. Иллюстрация экрана конфигурации для нового slave: 
3	В появившемся экране конфигурации slave, введите или измените: ● название нового профиля (ограничено 13 символами), ● комментарий (необязательно). Или щелкните "Catalog..." и выберите slave из семейства предварительно сконфигурированных профилей AS-Interface.
4	Введите: ● Код IO (соответствует конфигурации вх/вых), ● Код ID (идентификатор), (плюс ID1 для расширенного типа). Примечание: Поля "Inputs" и "Outputs" показывают число входных и выходных каналов. Они автоматически выполняются, когда вводится код IO.

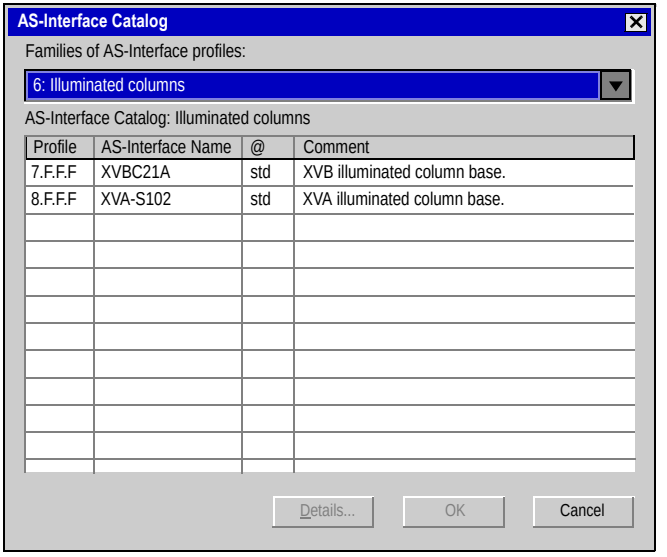
Шаг	Действие
5	Для каждого параметра определите: ● системное подтверждение (поставленный флажок при представлении "Bits", или десятичное значение между 0 и 15 при представлении "Decimal"), ● Название, более значащее, чем "Parameter X" (необязательно). Примечание: Выбранные параметры являются отображением постоянных параметров для представления главному устройству AS-Interface.
6	При необходимости измените "Address" (в пределах разрешенных адресов шины), кликнув на стрелках вверх/вниз слева от адреса (дается доступ к авторизированным адресам) или введите адрес с клавиатура.
7	Подтвердите конфигурацию slave, кликнув на кнопке "OK". Результат затем проверяется, чтобы убедиться, что: ● IO и ID авторизированы, ● адрес slave авторизирован (если используется ввод с клавиатуры) в соответствии с кодом ID ("bank" /B slaves доступны, только если код ID равен A). В случае возникновения ошибки, сообщение об ошибке предупреждает пользователя (например: "The slave cannot have this address" (“У slave не может быть такого адреса”) и снова отображается экран с начальными значениями (в профиле или адресе, в зависимости от ошибки).

Примечание: Программное обеспечение ограничивает число объявлений аналоговых slave до 7.

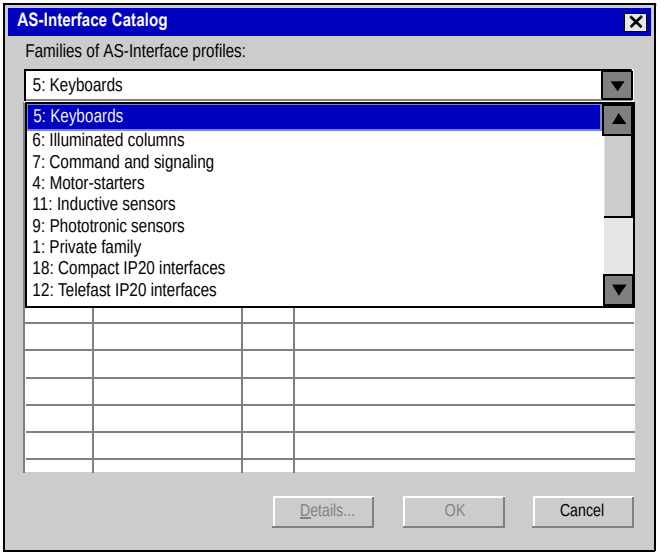
Примечание: О каталоге Schneider AS-Interface: когда Вы щелкаете на “Catalog”, Вы можете создавать и изменять slave в "Private family" (частном семействе) , отличные от содержащихся в каталоге Schneider AS-Interface.

Каталог AS-Interface

Кнопка Catalog может использоваться, чтобы облегчить конфигурацию slave на шине. Когда вы используете slave из семейства Schneider, используйте эту кнопку для упрощения и ускорения конфигурации.
При нажатии на кнопку "Catalog" в окне "Configure an AS-Interface slave" откроется следующее окно:



Выпадающее меню предоставляет доступ ко всем семействам каталога Schneider AS-Interface:



После выбора семейства появится список соответствующих slaves. Щелкните на требуемом slave и подтвердите "OK"

Примечание: Вы можете отобразить характеристики slave, щелкнув на "Details".

Примечание: Вы можете добавить и сконфигурировать slave, которые не являются часть каталога Schneider. Просто выделите частное семейство и сконфигурируйте новый slave.

Описание экрана отладчика

Обзор

Когда ПК **соединяется** с контроллером (после загрузки приложения в контроллер), вкладка "Debug" появляется справа от вкладки "Configuration"; она позволяет получить доступ к экрану отладчика.
Экран отладчика динамически обеспечивает отображение физической шины, которое включает:

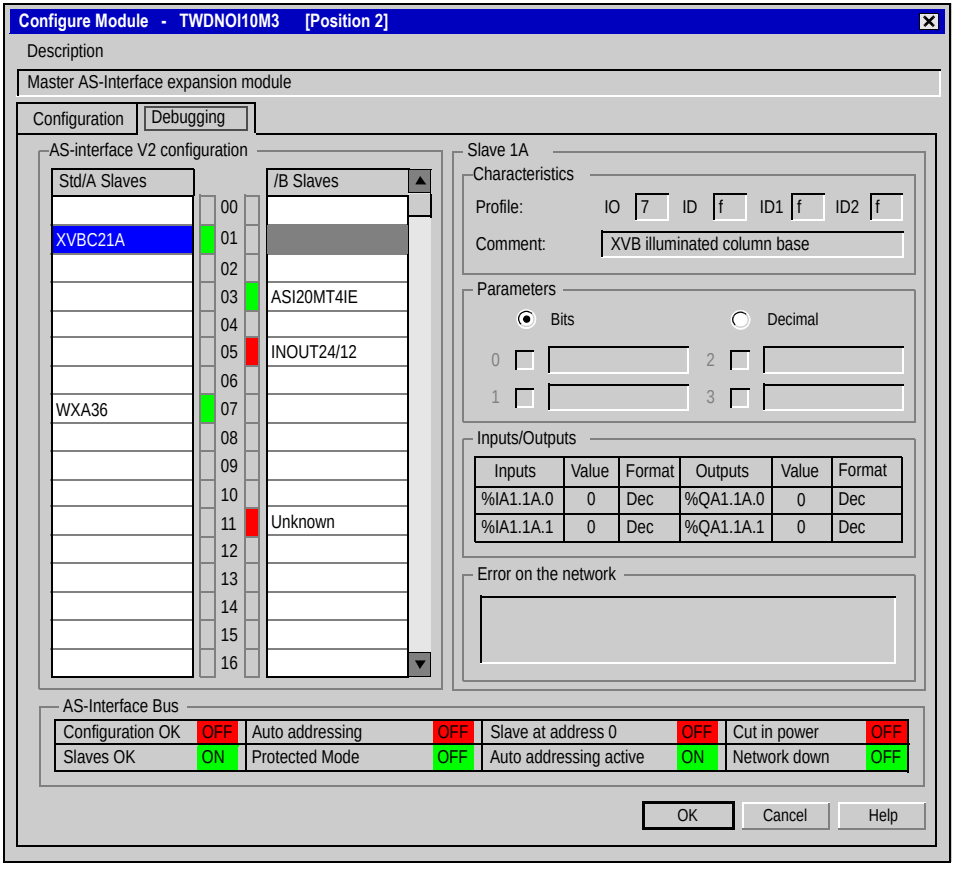
- Список ожидаемых slave (введенных) во время конфигурации с названиями, и список определенных slave (с неизвестными названиями, но ожидаемых),
- Статус модуля AS-Interface и подчиненных устройств,
- Отображение профиля, параметров и значений вх/вых выбранных slave.

Он также позволяет пользователю:

- Получить диагностики slave, на которых произошла ошибка (См. *Отображение статуса slave, стр. 173*),
- Изменить адрес slave в режиме online (См. *Изменение адреса slave , стр. 174*),
- Передать отображение slave в экран конфигурации (См. *Обновление конфигурации шины AS-Interface в оперативном режиме, стр. 176*),
- Адресовать все slave желаемыми адресами (при первой отладке).

Иллюстрация экрана отладчика

Иллюстрация экрана отладчика (только в режиме online) выглядит так:



Описание экрана отладчика

На экране отладчика представлена такая же информация, как на экране конфигурации. (См. *Описание экрана в режиме Offline, стр. 164*). Различия перечислены в следующей таблице:

Список	Описание
AS-interface V2 configuration (Конфигурация AS-interface V2)	Изображение физической шины. Включает статус slave: ● Зеленая индикаторная лампа: slave с этим адресом активен. ● Красная индикаторная лампа: произошла ошибка в slave с этим адресом, сообщение информирует Вас о типе ошибки в окне "Error on the network"(“Ошибка в сети”).
Slave xxA/B	Изображение конфигурации выбранного slave: ● Характеристики: изображение профиля (выделен серым, неизменяемый), ● Параметры: изображение параметров. Пользователь может выбрать только формат отображения параметров, ● Входы/Выходы: изображены значения входов/выходов. Неизменяемы.
Error on the network (Ошибка в сети)	Сообщает Вам тип ошибки, если на выбранном slave произошла ошибка.
AS-Interface Bus (Шина AS-interface)	Информация, полученная в результате неявной команды "Read Status"(“Прочитать статус”). ● Показывает статус шины: например, "Configuration OK = OFF" показывает, что указанная пользователем конфигурация не соответствует физической конфигурации шины, ● Показывает авторизированные функциональные возможности master модуля AS-Interface: например, "Automatic addressing active = ON" показывает, что режим автоматической адресации Master авторизирован.

Отображение статуса Slave

Когда индикаторная лампа, соответствующая адресу, красная, значит, произошла ошибка в slave, соответствующем этому адресу. Окно "Error on the network" предоставляет диагностику выбранного slave.
Описание ошибок:

- Профиль, определенный пользователем при конфигурации заданного адреса, не соответствует фактическому профилю, определенному для этого адреса на шине (диагностика: "Profile error"),
- Новый slave, не определенный при конфигурации, обнаружен на шине: а красная индикаторная лампа изображается для этого адреса и название slave ="Unknown" (диагностика: "Slave not projected"),
- Ошибка периферийного устройства, если определен slave, поддерживающий его (диагностика: "Peripheral fault"),
- Сконфигурированный профиль определен, но ни один slave не определен для этого адреса на шине (диагностика: "Slave not detected").

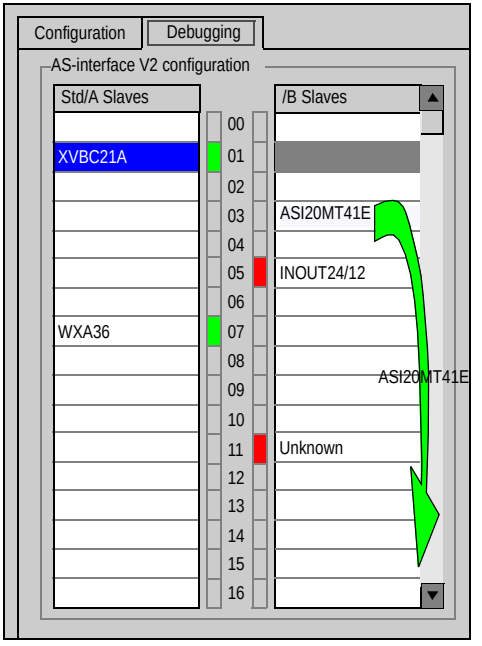
Модификация адреса slave

Обзор

Из экрана отладчика пользователь может изменить адрес slave в режиме online.

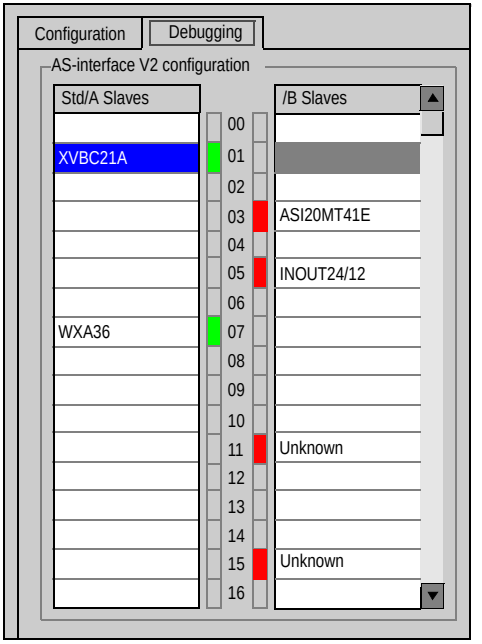
Модификация адреса slave

В следующей таблице показана процедура изменения адреса slave:

Шаг	Описание
1	Войдите к экран отладчика.
2	Выберите slave в зоне "AS-interface V2 Configuration".
3	Перетащите и бросьте slave в ячейку, соответствующую требуемому адресу. Иллюстрация: Перемещение slave 3B на адрес 15B методом drag and drop 

Шаг	Описание

Результат:
Все параметры slave автоматически проверяются, чтобы определить, возможна ли операция.
Иллюстрация результата:



После выполнения операции диагностика для slave по адресу 3B показывает "slave not detected", это означает, что slave, ожидаемого по этому адресу, больше там нет. При выборе адреса 15B, профиль и параметры перемещенного slave могут быть перемещены, но название slave остается неизвестным, т.к. он не ожидался по этому адресу.

Примечание: Профиль и параметры slave не ассоциируются с его названием. Несколько slaves с разными названиями могут иметь одинаковые профиль и параметры.

Обновление конфигурации шины AS-Interface в режиме online

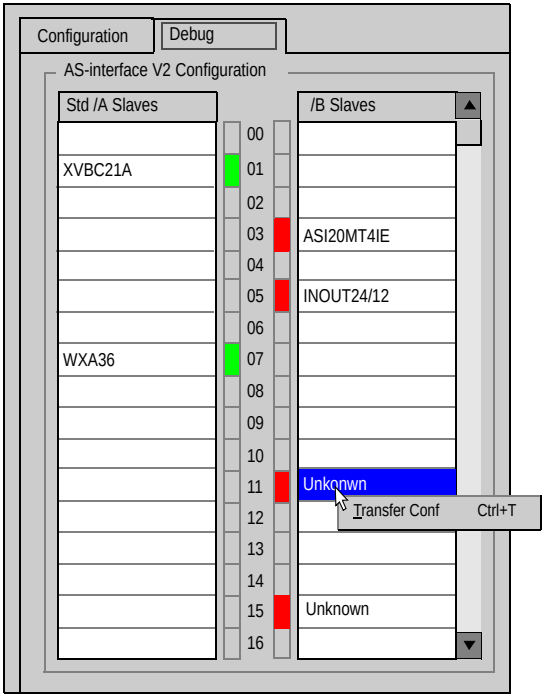
Обзор

В режиме online, никакие модификации экрана конфигурации не авторизируются и физическая конфигурация и программная конфигурация могут различаться. Любые различия в профиле и параметрах конфигурируемого и неконфигурируемого slave могут учитываться в экране конфигурации; в действительности, возможно передать любые изменения в экран конфигурации перед пересылкой нового приложения в контроллер. Процедура, необходимая для принятия в расчет физической конфигурации, следующая:

Шаг	Описание
1	Перенос желаемой конфигурации slave в экран конфигурации.
2	Прием конфигурации в экране конфигурации.
3	Подтверждение новой конфигурации.
4	Перенос приложения в модуль.

Перенос изображения Slave в экран конфигурации

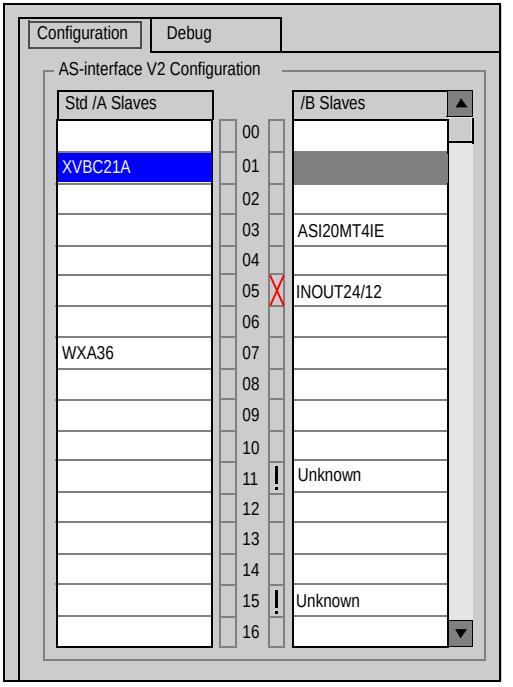
В случае, когда slave, не описанный в экране конфигурации, определяется на шине, появляется "Unknown" slave в зоне "AS-interface V2 Configuration zone" экрана отладчика для определенного адреса.
В следующей таблице описана процедура переноса изображения "Unknown" slave в экран конфигурации:

Шаг	Описание
1	Войдите к экран отладчика.
2	Выберите желаемый slave в зоне "AS-interface V2 Configuration".
3	Щелкните правой кнопкой мыши для выбора "Transfer Conf". Иллюстрация:  Результат: Изображение выбранного slave (изображение профиля и параметров) переносится в экран конфигурации.
4	Повторите операцию для каждого slave, чье изображение Вы хотите перенести в экран конфигурации.

Возврат в экран конфигурации

Когда пользователь возвращается в экран конфигурации, все новые перенесенные slave (неожиданные) становятся видимыми.

Иллюстрация экрана конфигурации после переноса всех slave:



Ключ:

- Крест показывает, что есть различия между изображением профиля, переданного slave, и профилем, изначально желаемым в экране конфигурации.
- Восклицательный знак показывает, что в экран конфигурации был добавлен новый профиль.

Пояснение:

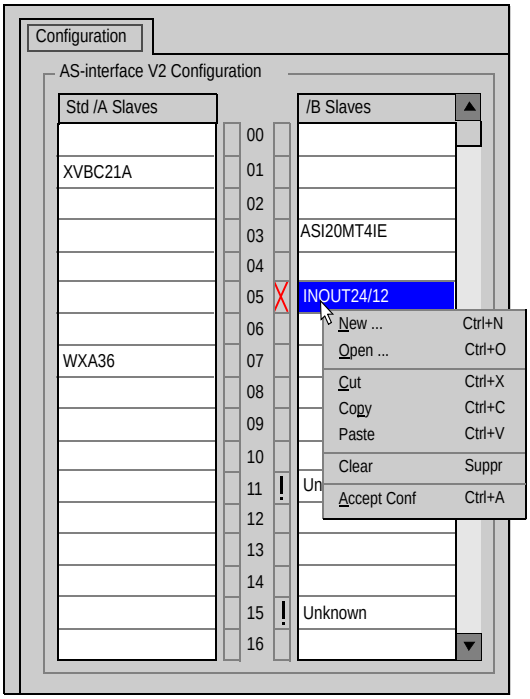
Экран конфигурации всегда показывает постоянное изображение желаемой конфигурации (поэтому slave все еще присутствует в 3B, несмотря на смену адреса (См. *Модификация адреса slave, стр. 174*)), завершенной текущим изображением шины.

Профили и параметры ожидаемых slave изображаются в соответствии с теми, которые ожидались. Профили и параметры неизвестных slave изображаются в соответствии с изображениями определенных slave.

Процедура пересылки окончательного приложения в модуль

Перед пересылкой нового приложения в модуль пользователь может для каждого slave принять определнный профиль и параметры (переданные в экран конфигурации) или изменить конфигурацию вручную (См. *Процедура объявления и конфигурирования slave, стр 166*).
В следующей таблице описаны шаги для подтверждения и передачи определенной конфигурации в модуль:

Шаг	Действие
1	Через ПО, отсоедините ПК от модуля. Примечание: Никакие изменения не могут быть произведены в экране конфигурации, если ПК соединен с модулем.
2	Щелчок правой кнопкой на желаемом slave.

Шаг	Действие
3	2 альтернативы: ● Выберите "Accept Conf" для принятия определенного профиля выбранного slave. Иллюстрация:  Для каждого slave помеченного крестом сообщение предупредит пользователя о том, что эта операция перепишет первоначальный профиль (изображены на экране) slave. ● Выберите другую альтернативу в меню правой кнопки, чтобы сконфигурировать выбранный slave вручную.
4	Повторите операцию для каждого желаемого slave в конфигурации.
5	Нажмите кнопку "OK", чтобы подтвердить и создать новое приложение. Результат: автоматический возврат в основной экран.
6	Перенесите приложение в модуль.

Автоматическая адресация подчиненного устройства AS-Interface V2

Обзор

Каждому slave на шине AS-Interface должен быть присвоен (через конфигурацию) уникальный физический адрес. Он должен быть таким же, как объявленный в TwidoSoft.

TwidoSoft предлагает утилиту автоматической адресации slave, чтобы не использовать консоль AS-Interface.

Утилита автоматической адресации используется для:

- замены ошибочного slave,
- вставки нового slave.

Процедура

В следующей таблице показана процедура установки параметра **Автоматической адресации**.

Шаг	Действие
1	Откройте экран конфигурации master модуля AS-Interface V2.
2	Щелкните на флажке автоматической адресации (**Automatic addressing)** в зоне режима **master (Master mode** zone). **Результат**: Утилита автоматической адресации будет активирована (флажок поставлен) или отключена (флажок не поставлен). **Примечание**: По умолчанию, параметр автоматической адресации выбран в экране конфигурации.

Как добавить подчиненное устройство в существующую конфигурацию AS-Interface V2

Обзор

Является возможным добавить устройство в существующую конфигурацию AS-Interface V2 без использования pocket programmer.
Эта операция возможна, когда:

- активна утилита **Автоматической адресации** режима конфигурации (См. *Автоматическая адресация подчиненного устройства AS-Interface V2, стр. 181*),
- один slave отсутствует в физической конфигурации,
- slave, который необходимо добавить, описан в экране конфигурации,
- slave имеет профиль, ожидаемый конфигурацией,
- slave имеет адрес 0 (A).

Модуль AS-Interface V2 автоматически присвоит slave значение, предопределенное при конфигурации.

Процедура

В следующей таблице показана процедура эффективного автоматического добавления нового slave.

Шаг	Действие
1	Добавьте новый slave в экране конфигурации в локальном режиме.
2	Выполните перенос конфигурации в ПЛК в подсоединенном режиме.
3	Физически свяжите новый slave с адресом 0 (A) с шиной AS-Interface V2.

Примечание: Приложение может быть модифицировано выполнением описанных выше манипуляций столько раз, сколько необходимо.

Автоматическая замена ошибочного подчиненного устройства AS-Interface V2

Принцип

Когда slave был объявлен с ошибкой, он может быть автоматически заменен slave того же типа.
Это происходит без останова шины AS-Interface V2 и без каких либо манипуляций, когда активна утилита режима конфигурации **Автоматическая адресация** (См. *Автоматическая адресация подчиненного устройства AS-Interface V2, стр. 181*).

Доступны две опции:

- Заменяемый slave программируется на тот же адрес используя pocket programmer, и имеет такой же профиль и подпрофиль, как у ошибочного slave. таким образом он автоматически вставляется в список определенных slave (LDS) и в список активных slave (LAS),
- Заменяемый slave пуст (адрес 0 (A), новый slave) и имеет такой же профиль, как у ошибочного slave. Он автоматически получает адрес замененного slave, и затем будет вставлен в список определенных slave (LDS) и список активных slave (LAS).

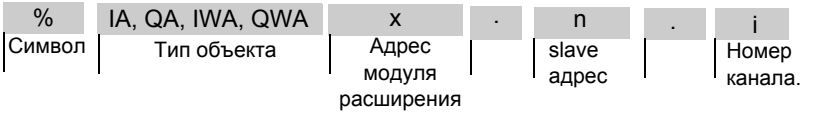
Адресация вх/вых, связанных с подчиненными устройствами, соединенными с шиной AS-Interface V2

Обзор

На этой странице представлена подробная информация, касающаяся адресации цифровых и аналоговых вх/вых подчиненных устройств.
Чтобы избежать путаницы с удаленными вх/вых, в синтаксисе AS-Interface доступны новые символы: %I**A**, например.

Иллюстрация

Рисунок напоминает принципы адресации:



Специальные значения

Следующая таблица содержит специальные значения подчиненных объектов AS-Interface V2:

Часть	Значения	Комментарий
IA	-	Отображение физического цифрового входа slave.
QA	-	Отображение физического цифрового выхода slave.
IWA	-	Отображение физического аналогового входа slave.
QWA	-	Отображение физического аналогового выхода slave.
x	1 до 7	Адрес модуля AS-Interface на шине расширения.
n	0A до 31B	Слот 0 не может быть сконфигурирован.
i	0 до 3	-

Примеры

В следующей таблице приведены некоторые примеры адресации вх/вых.:

Объекты вх/вых	Описание
%IWA4.1A.0	Аналоговый вход 0 slave 1A модуля AS-Interface, расположенного в позиции 4 на шине расширения.
%QA2.5B.1	Цифровой выход1 slave 5B модуля AS-Interface, расположенного в позиции 2 на шине расширения.
%IA1.12A.2	Цифровой вход2 slave 12A модуля AS-Interface, расположенного в позиции 1 на шине расширения.

Безусловный обмен

Объекты, описанные ниже, обмениваются безусловно, другими словами, они обмениваются автоматически при каждом цикле ПЛК.

Программирование и диагностика шины AS-Interface V2

Безусловный обмен

Объекты (слова и биты), связанные с шиной AS-Interface, предоставляют данные (например: операция шины, статус slave и т. д.) и дополнительные команды для выполнения улучшенного программирования функций AS-Interface.
Эти объекты безусловно обмениваются между контроллером Twido и AS-Interface Master по шине расширения:

- По запросу программы пользователя посредством инструкции: ASI_CMD (См. "Представление ASI_CMD" инструкции ниже)
- Через экран отладчика анимационной таблицы.

Зарезервированные специальные системные слова

Системные слова, зарезервированные в контроллере Twido для Master модулей AS-Interface, позволяют Вам определять состояние сети: %SW73 зарезервировано для первого модуля расширения AS-Interface и %SW74 для второго. Используются только первые 5 битов этих слов; они только для чтения.
В следующей таблице показаны используемые биты:

Системные слова	Бит	Описание
%SW73 и %SW74	0	статус системы (= 1 если конфигурация OK, иначе 0)
	1	обмен данными (= 1 обмен данными разрешен, иначе 0 (См. *Режимы работы интерфейсного модуля шины AS-Interface V2, стр. 189*))
	2	система остановлена (= 1 если режим Offline (См. *Режим Offline, стр. 189*)разрешен, иначе 0)
	3	прервана инструкция ASI_CMD (= 1 прервана, 0 в процессе)
	4	ошибка инструкции ASI_CMD (= 1 если в инструкции есть ошибка, иначе 0)

Пример использования (для первого модуля расширения AS-Interface):
Перед использованием инструкции ASI_CMD, бит %SW73:X3 должен быть проверен, чтобы убедиться, что инструкция не в процессе выполнения: проверка, что %SW73:X3 = 1.
Чтобы выяснить была ли инструкция выполнена правильно, проверьте, что бит %SW73:X4 равен 0.

Представление инструкции ASI_CMD

Для каждой программы пользователя инструкция ASI_CMD позволяет пользователю программировать сеть и получить диагностику slave.
Параметры инструкции передаются через внутренние слова (слова памяти) **%MW**x.
Синтаксис инструкции следующий:
ASI_CMDn **%MWx:**l
Легенда:

Символ	Описание
n	Адрес модуля расширения AS-Interface (1 по 7).
x	Номер первого внутреннего слова (слова памяти) , переданного как параметр (0 to 254).
l	Длина инструкции в количестве слов (2).

Использование инструкции ASI_CMD

В следующей таблице описано действие инструкции ASI_CMD в соответствии со значением параметров %MW(x) и %MW(x+1), когда необходимо. Для запросов диагностики slave, результат возвращается в слово %MW(x+1).

%MWx	%MWx+1	Действие
1	0	Выход из режима Offline.
1	1	Переключение в режим Offline.
2	0	Запрет обмена данными между Master и slave (вход в режим Data Exchange Off).
2	1	Авторизация обмена данными между Master и slaves (выход из режима Data Exchange Off).
3	Зарезерв.	-
4	Результат	Чтение списка активных slave (LAS таблица) с адресами от 0A до 15A (1 бит на slave).
5	Результат	Чтение списка активных slave (LAS таблица) с адресами от 16A до 31A (1 бит на slave).
6	Результат	Чтение списка активных slave (LAS таблица) с адресами от 0B до 15B (1 бит на slave).
7	Результат	Чтение списка активных slave (LAS таблица) с адресами от 16B до 31B (1 бит на slave).
8	Результат	Чтение списка определенных slave (LDS таблица) с адресами от 0A до 15A (1 бит на slave).
9	Результат	Чтение списка определенных slave (LDS таблица) с адресами от 16A до 31A (1 бит на slave).
10	Результат	Чтение списка определенных slave (LDS таблица) с адресами от 0B до 15B (1 бит на slave).

%MWx	%MWx+1	Действие
11	Результат	Чтение списка определенных slave (LDS таблица) с адресами от 16B до 31B (1 бит на slave).
12	Результат	Чтение списка периферийных ошибок slave (LPF таблица) с адресами от 0A до 15A (1 бит на slave).
13	Результат	Чтение списка периферийных ошибок slave (LPF таблица) с адресами от16A до 31A (1 бит на slave).
14	Результат	Чтение списка периферийных ошибок slave (LPF таблица) с адресами от 0B до 15B (1 бит на slave).
15	Результат	Чтение списка периферийных ошибок slave (LPF таблица) с адресами от 16B до 31B (1 бит на slave).
16	Результат	Чтение статуса шины. См. подробности о результате в следующем параграфе.

Примечание: Статус шины обновляется при каждом сканировании ПЛК, но результат инструкции чтения шины ASI_CMD доступен только в конце текущего сканирования ПЛК.

Подробности о результате инструкции ASI_CMD для чтения состояния шины

В случае, когда статус шины читается инструкцией ASI_CMD (значение параметра %MWx равно 16),формат результата в слове %MWx+1 следующий:

%MWx+1		Назначение (1=OK, 0=не OK)
младший значащий	бит 0	Конфигурация OK
	бит 1	LDS.0 (присутствует slave с адресом 0)
	бит 2	Автоадресация активна
	бит 3	Автоадресация доступна
	бит 4	режим конфигурации активен
	бит 5	Нормальное функционирование активно
	бит 6	APF (проблемы источника питания)
	бит 7	Offline готовность
старший значащий	бит 0	Периферийная ошибка
	бит 1	Обмен данными активен
	бит 2	Offline режим
	бит 3	Нормальный режим (1)
	бит 4	Ошибка коммуникации с AS-Interface Master
	бит 5	ASI_CMD инструкция в процессе
	бит 6	ошибка инструкции ASI_CMD

Подробности о результате инструкции ASI_CMD для чтения статуса slave

В случае диагностики slave инструкцией ASI_CMD (%MWx значВ следующей таблице содержатся подробности о результате, в соответствии со значением слова %MWx:

%MWx	%MWx+1															
значение	старший значащий байт								младший значащий байт							
	бит 7	бит	бит 5	бит 4	бит 3	бит 2	бит 1	бит 0	бит 7	бит 6	бит 5	бит 4	бит 3	бит 2	бит 1	бит 0
4, 8, 12	15A	14A	13A	12A	11A	10A	9A	8A	7A	6A	5A	4A	3A	2A	1A	0A
5, 9, 13	31A	30A	29A	28A	27A	26A	25A	24A	23A	22A	21A	20A	19A	18A	17A	16A
6, 10, 14	15B	14B	13B	12B	11B	10B	9B	8B	7B	6B	5B	4B	3B	2B	1B	0B
7, 11, 15	31B	30B	29B	28B	27B	26B	25B	24B	23B	22B	21B	20B	19B	18B	17B	16B

Чтобы прочитать, активен ли slave 20B, инструкция ASI_CMD должна быть выполнена с внутренним словом %MWx = 7. результат возвращается во внутреннее слово %MWx+1; статус slave 20Bзадается значением бита младшего значащего бита: если бит 4 равен 1, тогда slave 20B активен.

Примеры программирования для инструкции ASI_CMD

Чтобы заставить AS-Interface Master (позиция 1 на шине расширения)переключиться в режим Offline:

```
LD 1
[%MW0 := 16#0001 ]
[%MW1 := 16#0001 ]
LD %SW73:X3            //Если нет инструкций ASI_CMDв процессе, продолжить
[ASI_CMD1 %MW0:2]      //Чтобы заставить переключиться в режим Offline
```

Чтобы прочитать таблицу slave активных по адресам с 0A по 15A:

```
LD 1
[%MW0 := 16#0004 ]
[%MW1 := 16#0000           //необязательно]
LD %SW73:X3   //Если нет инструкций ASI_CMDв процессе, продолжить
[ASI_CMD1 %MW0:2]      //Чтобы прочитать таблицу LAS  для адресов с 0A по 15A
```

Режимы работы интерфейсного модуля шины AS-Interface V2

Обзор

У интерфейсного модуля TWDNOI10M3 шины AS-Interface есть три режима работы, each of which responds to particular needs. These modes are:

- Защищенный режим,
- Автономный режим,
- Режим без обмена данными.

Использование инструкции ASI_CMD (См. *Представление инструкции ASI_CMD, стр.186*) в программе пользователя позволяет входить или выходить из этих режимов.

Защищенный режим

Защищенный режим обычно используется для выполняющегося приложения. Предполагается, что модуль AS-Interface V2 сконфигурирован в TwidoSoft. Он:

- непрерывно проверяет, что список определенных slave тот же, что и список ожидаемых slave,
- контролирует источник питания.

В этом режиме slave будет активирован только при условии, что он был объявлен в конфигурации и был определен.

При загрузке или в фазе конфигурации контроллер Twido переводит модуль AS-Interfacев защищенный режим.

Автономный режим

Когда модуль переводится в автономный режим, он сначала сбрасывает все присутствующие slave в ноль и останавливает обмен на шине. В автономном режимы выходы устанавливаются в ноль.

Кроме использования кнопки PB2 на модуле TWDNOI10M3 AS-Interface, автономного режима можно достичь через ПО, при помощи инструкции ASI_CMD (См. *Примеры программирования инструкции ASI_CMD, стр. 188*), которая также позволяет выйти из режима и возвратиться в защищенный режим.

Режим без обмена данными

При переходе в режим без обмена данными обмены на шине продолжают выполняться, но данные не обновляются.

В этот режим можно перейти только по инструкции ASI_CMD (См. *использование инструкции ASI_CMD, стр.186*).

Работа дисплея оператора

10

Обзор

Предмет

В этой главе представлена подробная информация о использовании опционального дисплея оператора Twido.

Содержание главы

Эта глава содержит следующие темы:

Дисплей оператора

Введение

Дисплей оператора является опцией Twido для отображения и контроля прикладных данных и некоторых функций контроллера, таких как рабочее состояние и часы реального времени (RTC). Эта опция доступна в виде картриджа (TWDXCPODC) для компактных контроллеров или модуля расширения (TWDXCPODM) для модульных контроллеров.
Дисплей оператора имеет два рабочих режима:

- Режим отображения: только отображает данные.
- Режим редактирования: позволяет изменять данные.

Примечание: Дисплей оператора обновляется в определенный интервал цикла сканирования контроллера. Это может вызвать путаницу при интерпретации изображения выводов, назначенных для %PLS или импульсов %PWM. В момент считывания значений выходов, они всегда будут в нуле и это значение будет отображаться.

Экраны и функции

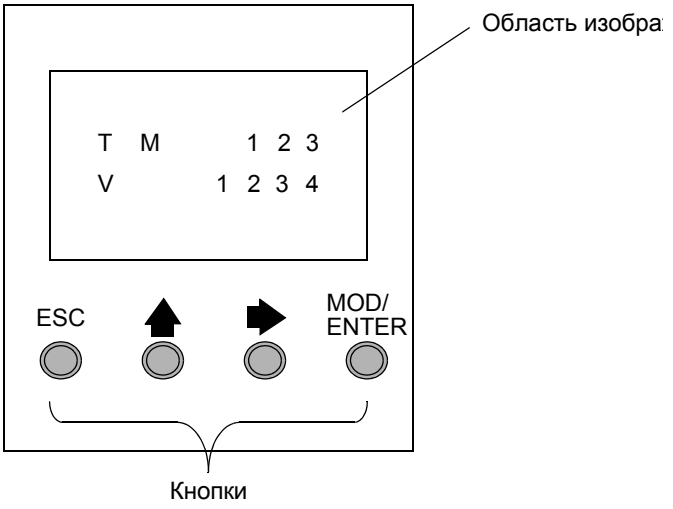
Дисплей оператора представляет следующие отдельные экраны с ассоциированными функциями, которые Вы можете выполнить для каждого экрана.

- Идентификация контроллера и информация о состоянии: Operations Display
  Отображает обзор программно-аппаратных средств и состояние контроллера. Изменение состояния контроллера при помощи команд Run, Initial и Stop.
- Системные объекты и переменные: Data Display
  Выбор прикладных данных по адресам: %I, %Q, и всех остальных программных объектов базового контроллера. Контроль и изменение значения выбранного объекта данных.
- Настройки последовательного порта: Communication Display
  Отображение и изменение настроек последовательного порта.
- Время: Time/Date Display
  Отображение и конфигурирование текущих даты и времени (если установлены часы реального времени).
- Коррекция реального времени: RTC Factor
  Отображение и изменение корректирующего значения RTC для опции RTC.

Примечание: Часы и коррекция реального времени доступны только, если установлен опциональный картридж часов реального времени (TWDXCPRTC).

Иллюстрация

На следующей иллюстрации показан вид дисплея оператора , состоящего из области изображения и четырех кнопок.



Область изображения

Дисплей оператора имеет LCD дисплей, способный отобразить две строки символов:

- Первая строка дисплея имеет три 13-сегментных символа и четыре 7-сегментных символа.
- Вторая линия имеет один 13-сегментный символ, один 3-сегментный символ (для знака плюс/минус) и пять 7-сегментных символа.

Кнопки

Функции кнопок зависят от режима дисплея оператора.

Кнопка	В режиме отображения	В режиме редактирования
ESC		Отказ от изменений и возврат в предыдущий экран.
▲		Переход к следующему значению редактируемого объекта.
➡	Переход в следующий экран.	Переход к следующему типу объекта для редактирования.
MOD/ENTER	Переход в режим редактирования	Принятие изменений и возврат в предыдущий экран.

Выбор и навигация по экранам

Начальный экран дисплея оператора показывает информацию об идентификации и состоянии контроллера. Нажмите на кнопку ➡ для последовательного перехода между экранами. Экраны для часов и корректирующего фактора реального времени не отображаются, если опциональный картридж RTC (TWDXCPRTC) не определен на контроллере. Нажмите на ESC для возврата в начальный экран. Для большинства экранов нажатие на кнопку ESC вернет в экран Информации об идентификации и состоянии контроллера. Только во время редактирования системных объектов и переменных, не являющихся начальными данными (%I0.0.0), нажатие на ESC вернет в первое или начальное значение системного объекта.

Для изменения значения объекта, вместо нажатия на кнопку ➡ для перехода к первой цифре значения, еще раз нажмите кнопку MOD/ENTER.

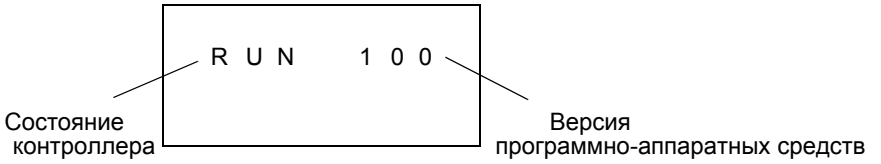
Информация об идентификации и состоянии контроллера

Введение

Начальный экран опционального дисплея оператора Twido отображает информацию об идентификации и состоянии контроллера.

Пример

Версия программно-аппаратных средств отображается в правом верхнем углу области изображения, а состояние контроллера отображается в верхнем левом углу, как показано на рисунке:



Состояния контроллера

Возможные состояния контроллера:

- **NCF: Не сконфигурирован**
  Контроллер находится в состоянии NCF до загрузки приложения. Никакое другое состояние не возможно до загрузки прикладной программы. Вы можете проверить Вх/Вых, изменяя системный бит S8 (см. *Системные биты (%S), стр. 434*).
- **STP: Остановка**
  Когда приложение присутствует в контроллере, состояние изменяется на STP или Остановка . В этом состоянии приложение не выполняется. Входы обновляются и значения данных сохраняются в последнем значении. Выходы в этом состоянии не обновляются.
- **INI: Начальное**
  Вы можете перевести контроллер в начальное состояние INI только из состояния STP. Приложение не выполняется. Входы обновляются и значения данных устанавливаются в начальное значение. Выходы в этом состоянии не обновляются.
- **RUN: Выполнение**
  В состоянии RUN приложение выполняется. Входы контроллера обновляются и значения данных устанавливаются в соответствии с приложением. Это единственное состояние, в котором выходы обновляются.
- **HLT: Останов (ошибка приложения пользователя)**
  Если контроллер вошел в состояние ERR или ошибка, приложение останавливается. Входы обновляются и значения данных сохраняются в последнем значении. Из этого состояния выходы не обновляются. В этом режиме код ошибки отображается в нижней правой части дисплея оператора как беззнаковое десятичное число.
- **NEX: Не выполнимо**
  В пользовательскую логику были внесены изменения online. Последствия: приложение больше не выполнимо. Оно не выйдет из этого состояния, пока все причины не будут устранены.

Отображение и изменение состояний контроллера

Используя дисплей оператора, Вы можете перейти в состояние INI из STP, или из STP в RUN, или из RUN в STP. Выполните следующее для изменения состояния контроллера:

Шаг	Действие
1	Нажимайте кнопку ➡, пока не отобразится Экран Операций (или нажмите ESC). Текущее состояние контроллера отображается в левом верхнем углу области изображения.
2	Нажмите кнопку MOD/ENTER для входа в режим редактирования.
3	Нажмите кнопку ⬆ для выбора состояния контроллера.

Шаг	Действие
4	Нажмите кнопку MOD/ENTER для приема измененного значения, или нажмите ESC для отмены любых изменений, сделанных в режиме редактирования.

Системные объекты и переменные

Введение

Опциональный дисплей оператора представляет следующие возможности для мониторинга и настройки прикладных данных:

- Выбор прикладных данных по адресу (такому как %I или %Q).
- Контроль значения выбранного программного объекта/переменной.
- Изменение значения отображаемого в данный момент объекта данных (включая форсирование входов и выходов).

Системные объекты и переменные

В следующей таблице перечислены системные объекты и переменные, в порядке доступа, которые могут быть отображены и изменены в дисплее оператора.

Объект	Переменная/ Атрибут	Описание	Доступ
Вход	%Ix.y.z	Значение	Чтение/ Форсирование
Выход	%Qx.y.z	Значение	Чтение/Запись/ Форсирование
Таймер	%TMX.V %TMX.P %TMX.Q	Текущее значение Предуст. значение Готово	Чтение/Запись Чтение/Запись Чтение
Счетчик	%Cx.V %Cx.P %Cx.D %Cx.E %Cx.F	Текущее значение Предуст. значение Готово Пуст Полон	Чтение/Запись Чтение/Запись Чтение Чтение Чтение
Бит памяти	%Mx	Значение	Чтение/Запись
Слово памяти	%MWx	Значение	Чтение/Запись
Константное слово	%KWx	Значение	Чтение
Системный бит	%Sx	Значение	Чтение/Запись
Системное слово	%SWx	Значение	Чтение/Запись
Аналоговый вход	%IWx.y.z	Значение	Чтение
Аналоговый выход	%QWx.y.z	Значение	Чтение/Запись
Быстрый счетчик	%FCx.V %FCx.P %FCx.D	Текущее значение Предуст. значение Готово	Чтение Чтение/Запись Чтение
Очень быстрый счетчик	%VFCx.V %VFCx.P %VFCx.U %VFCx.C %VFCx.S0 %VFCx.S1 %VFCx.F %VFCx.M %VFC.T %VFC.R %VFC.S	Текущее значение Предуст. значение Направление счета Запирающ. значение Порог 0 Порог 1 Переполнен Частота Временная ось Разреш.отображ.вых Разреш.отображ.вх.	Чтение Чтение/Запись Чтение Чтение Чтение/Запись Чтение/Запись Чтение Чтение/Запись Чтение/Запись Чтение/Запись Чтение/Запись
Вх. сетевое слово	%INWx.z	Значение	Чтение
Исх. сетевое слово	%QNWx.z	Значение	Чтение/Запись

Объект	Переменная/ Атрибут	Описание	Доступ
Grafcet	%Xx	Бит шага	Чтение
Генератор импульсов	%PLS.N %PLS.P %PLS.D %PLS.Q	Число импульсов Предуст. значение Готово Текущий выход	Чтение/Запись Чтение/Запись Чтение Чтение
Генератор широтной модуляции	%PWM.R %PWM.P	Коэффициент Предуст. значение	Чтение/Запись Чтение/Запись
Барабанный контроллер	%DRx.S %DRx.F	Текущий номер шага Полон	Чтение Чтение
Счетчик шага	%SCx.n	Бит счетчика шага	Чтение/Запись
Регистр	%Rx.I %Rx.O %Rx.E %Rx.F	Вход Выход Пуст Полон	Чтение/Запись Чтение/Запись Чтение Чтение
Сдвиговый регистр битов	%SBR.x.yy	Бит регистра	Чтение/Запись
Сообщение	%MSGx.D %MSGx.E	Готово Ошибка	Чтение Чтение
Вход AS-Interface slave	%IA.x.y.z	Значение	Чтение/ Форсирование
Аналоговый вход AS-Interface slave	%IWA.x.y.z	Значение	Чтение
Выход AS-Interface slave	%QA.x.y.z	Значение	Чтение/Запись/ Форсирование
Аналоговый выход AS-Interface slave	%QWA.x.y.z	Значение	Чтение/Запись

Примечания:

1. Переменные не будут отображаться, если они не используются в приложении, потому что Twido использует динамическое выделение памяти.
2. Если значение %MW больше +32767 или меньше -32768, дисплей оператора продолжит мигать.
3. Если значение %SW больше 65535, дисплей оператора продолжит мигать, исключая %SW0 и %SW11.Если введенное значение больше предельного, значение вернется в конфигурируемое.
4. Если введенное значение для %PLS.P больше предельного, записанное значение будет значением насыщения.

Отображение и изменение объектов и переменных

Каждый тип системного объекта достигается, начиная с объекта входа (%I), последовательно через объект сообщения (%MSG)и в конце возвращаясь в объект входа (%I).
Для отображения системного объекта:

Шаг	Действие
1	Нажимайте кнопку ➡, пока не появится экран данных Data Display. Объект входа ("I") будет отображен в верхнем левом углу области изображения. Буква "I " (или название объекта, который до этого смотрели как данные) не мигает.
2	Нажмите кнопку MOD/ENTER для входа в режим редактирования. Символ объекта входа "I" (или название объекта, который до этого смотрели как данные) начинает мигать.
3	Нажимайте кнопку ⬆ для последовательного перемещения по списку объектов.
4	Нажимайте кнопку ➡ для последовательного перемещения по полям типа объекта и нажимайте кнопку ⬆ для увеличения значения этого поля. Вы можете использовать кнопки ➡ и ⬆ для навигации и изменения всех полей изображенного объекта.
5	Повторяйте шаги 3 и 4 пока редактирование не будет выполнено.
6	Нажмите на кнопку MOD/ENTER для принятия измененных значений. Примечание: название и адрес объекта должны быть объявлены действительными до принятия любых изменений. Они должны существовать в конфигурации контроллера к моменту использования дисплея оператора. Нажмите ESC для отмены изменений, сделанных в режиме редактирования.

Значения данных и форматы отображения

В общем, значение объекта или переменной показывается как целое число со знаком или без в нижней правой части области изображения. В добавлении к этому, во всех полях устраняются незначащие первые нули у изображаемых значений. Адрес каждого объекта отображается на дисплее оператора в одном из следующих семи форматов:

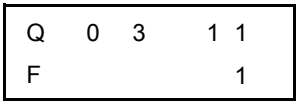
- Формат вх/вых
- Формат вх/вых slave AS-Interface
- Формат функционального блока
- Простой формат
- Формат сетевых вх/вых
- Формат счетчика шагов
- Формат сдвигающего регистра битов

Формат вх/вых

объекты входов/выходов (%I, %Q, %IW и %QW) имеют адреса, состоящие из трех частей (например: %IX.Y.Z), и изображаются следующим образом:

- Тип объекта и адрес контроллера вверху слева
- Адрес расширения вверху в центре
- Канал вх/вых вверху справа

В случае простого входа (%I) и выхода (%Q), нижняя левая часть дисплея будет содержать символ или "U" для нефорсируемого, или "F" для форсируемого бита. Форсируемое значение отображается снизу справа. Объект выхода %Q0.3.11 выглядит в области изображения следующим образом:

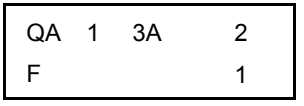


Формат вх/вых slave AS-Interface

Объекты вх/вых slave AS-Interface (%IA, %QA, %IWA и %QWA) имеют адрес, состоящий из четырех частей (например: %IAx.y.z), и изображаются следующим образом:

- Тип объекта вверху слева
- Адрес master AS-Interface на шине расширения в центре слева вверху
- Адрес slave на шине AS-Interface в центре справа вверху
- канал вх/вых slave вверху справа.

В случае простого входа (%IA) и выхода (%QA), нижняя левая часть дисплея будет содержать символ или "U" для нефорсируемого, или "F" для форсируемого бита. Форсируемое значение отображается снизу справа. Объект выхода %QA1.3A.2 выглядит в области изображения следующим образом:



Формат функционального блока

Функциональные блоки (%TM, %C, %FC, %VFC, %PLS, %PWM, %DR, %R и %MSGj) имеют адреса, состоящие из двух частей, содержащие номер объекта и название переменной или аттрибута. Они изображаются следующим образом:

- Название функционального блока вверху слева
- Номер (или экземпляр) функционального блока вверху справа
- Переменная или атрибут внизу слева
- Значение атрибута внизу справа

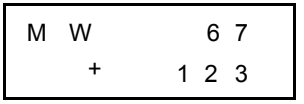
В следующем примере текущее значение таймера номер 123 установлено в 1,234.

Простой формат

Простой формат используется для объектов %M, %MW, %KW, %MD, %KD, %MF, %KF, %S, %SW и %X следующим образом:

- Номер объекта вверху справа
- Значение со знаком для объектов внизу

В следующем примере слово памяти номер 67 содержит значение +123.

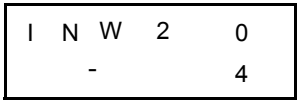


Формат сетевых вх/вых

Объекты сетевых вх/вых (%INW и %QNW) изображаются следующим образом:

- Тип объекта вверху слева
- Адрес контроллера вверху в центре
- Номер объекта вверху справа
- Значение со знаком для объекта внизу

В следующем примере для первого входного сетевого слова удаленного контроллера, сконфигурированного на удаленный адрес #2, установлено значение -4.

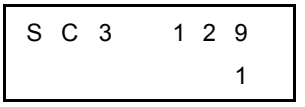


Формат счетчика шагов

Формат счетчика шагов (%SC) отображает номер объекта и бит счетчика шагов следующим образом:

- Название и номер объекта вверху слева
- Бит счетчика шагов вверху справа
- Значение бита счетчика шагов внизу

В следующем примере бит номер 129 счетчика шагов номер 3 установлен в 1.

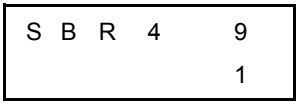


Формат сдвигающего регистра битов

Сдвигающий регистр битов (%SBR) изображаются следующим образом:

- Название и номер объекта вверху слева
- Номер бита регистра вверху справа
- Значение бита регистра внизу

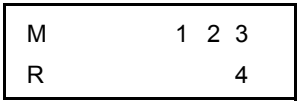
В следующем примере показан сдвигающий регистр номер 4.



Настройки последовательного порта

Введение

Дисплей оператора позволяет отображать настройки протокола и изменять адреса всех последовательных портов, сконфигурированных используя TwidoSoft. Максимальное количество последовательных портов - два. В следующем примере первый порт сконфигурирован как протокол Modbus с адресом 123. Второй последовательный порт сконфигурирован как дистанционная связь с адресом 4.



Отображение и изменение настроек последовательного порта

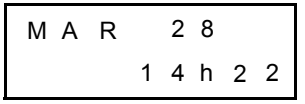
Контроллеры Twido поддерживают до двух последовательных портов. Для отображения настроек последовательного порта при помощи дисплея оператора:

Шаг	Действие
1	Нажимайте кнопку ➡ пока не появится Экран Коммуникаций (Communication Display). одиночная буква настройки протокола первого последовательного порта ("M", "R", or "A") будет отображена в верхнем левом углу дисплея оператора.
2	Нажмите кнопку MOD/ENTER для входа в режим редактирования.
3	Нажимайте кнопку ➡, пока не окажетесь в поле, которое хотите изменить.
4	Нажмите кнопку ⬆ для увеличения значения поля.
5	Продолжайте шаги 3 и 4, пока не закончите настройку адресов.
6	Нажмите кнопку MOD/ENTER для принятия измененных значений или ESC для отмены любых изменений, произведенных в режиме редактирования.
7	

Часы

Введение

Вы можете изменить дату и время, используя дисплей оператора, если опциональный картридж RTC (TWDXCPRTC) установлен на Вашем контроллере Twido. Месяц отображается в верхней левой части дисплея HMI. Пока не будет введено действительное время, поле месяца будет содержать значение "RTC". День месяца отображается в верхнем правом углу дисплея. Время дня показывается в военном формате. Часы и минуты отображаются в нижнем правом углу дисплея и разделены буквой "h". Следующий пример показывает, что часы RTC установлены на Март (March) 28, на 2:22 PM.



Отображение и изменение часов

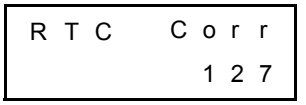
Для отображения и изменения часов:

Шаг	Действие
1	Нажимайте кнопку ➡, пока не появится Экран Время/Дата (Time/Date). Значение месяца ("JAN", "FEB") будет отображено в верхнем левом углу области изображения. Значение "RTC" будет отображено в верхнем левом углу, если месяц не был инициализирован.
2	Нажмите кнопку MOD/ENTER для входа в режим редактирования.
3	Нажимайте кнопку ➡, пока не окажетесь в поле, которое хотите изменить.
4	Нажимайте на кнопку ⬆, чтобы увеличивать значение этого поля.
5	Продолжайте шаги 3 и 4 пока не установите значение Time of Day.
6	Нажмите кнопку MOD/ENTER для принятия измененных значений или ESC для отмены любых изменений, произведенных в режиме редактирования.

Фактор коррекции реального времени

Введение

Вы можете отобразить и изменить фактор коррекции реального времени используя дисплей оператора. Каждый опциональный модуль часов реального времени (RTC) имеет значение фактора коррекции RTC, которое используется для исправления неточностей кристалла модуля RTC. Корректирующий фактор является беззнаковым 3-цифровым целым от 0 до 127 и отображается в нижнем правом углу дисплея.
Следующий пример показывает корректирующий фактор 127.



Отображение и изменение коррекции RTC

Для отображения и изменения фактора коррекции реального времени:

Шаг	Действие
1	Нажимайте кнопку ➡, пока не появится Экран Фактора RTC (RTC Factor Display). "RTC Corr" будет отображаться в верхней линии дисплея оператора.
2	Нажмите на кнопку MOD/ENTERдля входа в режим редактирования.
3	Нажимайте на кнопку ➡, пока не перейдете к полю, которое необходимо изменить.
4	Нажимайте на кнопку ⬆, чтобы увеличивать значение этого поля.
5	Продолжайте шаги 3 и 4, пока не завершите установку фактора коррекции RTC.
6	Нажмите кнопку MOD/ENTER для принятия измененных значений или ESC для отмены любых изменений, произведенных в режиме редактирования.

Описание языков Twido

III

Обзор

Тема этой части

Эта часть представляет инструкции для использования языков программирования Лестничной логики, Списка инструкций и Grafcet для создания управляющих программ для программируемых контроллеров Twido.

Содержание этой части

Эта часть содержит следующие главы:

Язык лестничной логики

11

Обзор

Предмет

Эта глава описывает программирование на языке лестничной логики.

Содержание главы

Эта глава содержит следующие темы:

Введение в лестничные диаграммы

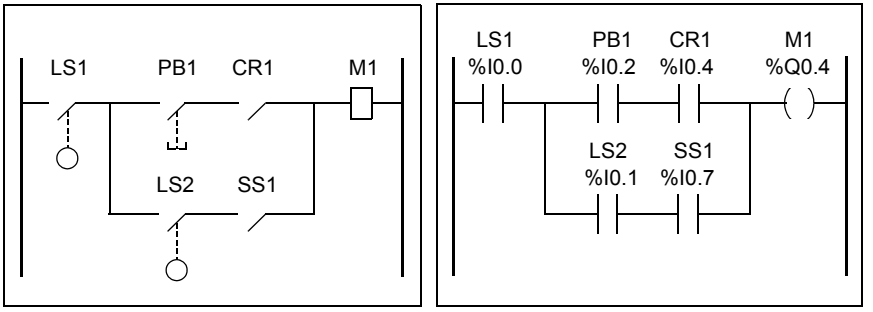
Введение

Лестничные диаграммы подобны релейным логическим диаграммам, которые представлены релейными схемами управления. Основные различия между ними состоят в следующих особенностях лестничного программирования, которые отсутствуют в релейных логических диаграммах:·

- Все входы представлены символами контактов (⊣⊢).
- Все выходы представлены символами катушек (⊣( )⊢).
- Числовые операции включены в набор графических лестничных инструкций.

Эквивалентность лестничных диаграмм и линейных схем

На следующем рисунке показана упрощенная схема соединений релейной логической схемы и эквивалентной лестничной диаграммы.



Релейная логическая схема

Лестничная диаграмма

Обратите внимание, что на рисунке все входы, связанные с переключательными устройствами на релейной логической диаграмме, показываются, как контакты на лестничной диаграмме. Выход M1, на релейной логической схеме, представлен символом катушки на лестничной диаграмме. Номера адресов, указанные над каждым символом контакта/катушки на лестничной диаграмме являются указателями на местоположение внешних входов/выходов контроллера.

Ступени в языке лестничной логики

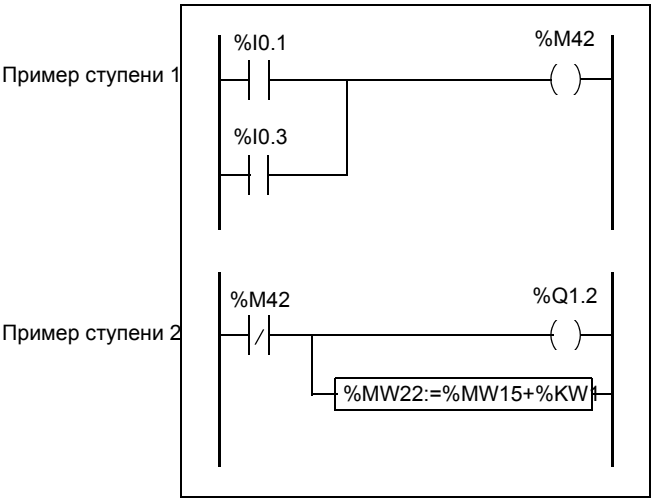
Программа, написанная на языке лестничной логики, состоит из ступеней, каждая из которых является набором графических инструкций, расположенных между двумя вертикальными шинами питания.
Ступени выполняются контроллером последовательно.
Графические инструкции включают в себя следующие функции:·

- Входы/выходы контроллера (кнопки, датчики, реле, контрольные индикаторы и т.д.)
- Функции контроллера (таймеры, счетчики и т.д.)
- Математические и логические операции (сложение, деление, логическое "и", исключающее "или" и т.д.)
- Операции сравнения и другие числовые операции(A<B, A=B, операция сдвига, операция поворота и т.д.)
- Внутренние переменные контроллера (биты, слова и т.д.)

Эти графические инструкции организуются вертикальными и горизонтальными связями, которые, в конечном счете, ведут к одному или нескольким выходам и/или действиям. Ступень не может поддерживать более чем одну группу связанных элементов.

Пример ступеней языка лестничной логики

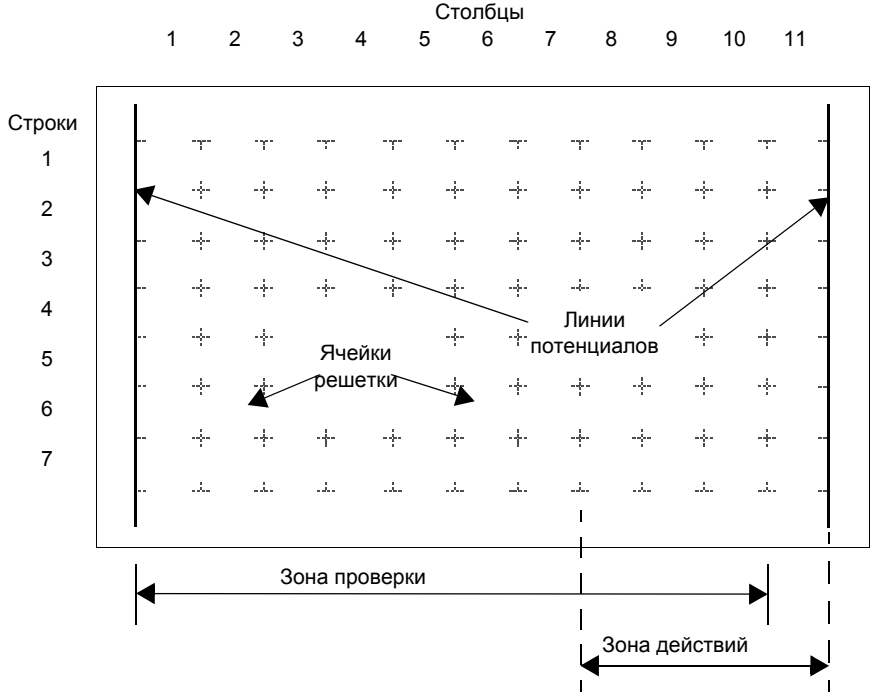
На следующем рисунке приведен пример программы из двух ступеней на языке лестничной логики.



Принципы программирования лестничных диаграмм

Программируемая решетка

Каждая ступень состоит из решетки, в которой семь строк и одинадцать столбцов, которые организованы в две зоны, как показано на следующем рисунке.



Зоны решетки

Программируемая решетка разделена на две зоны:

- Зона проверки
  Содержит условия, которые проверяются для выполнения действий. Она состоит из столбцов 1-10, и содержит контакты, функциональные блоки и блоки сравнения.
- Зона действий
  Содержит выход или оператор, который будет выполнен согласно результатам проверки условий в зоне проверки.
  Она состоит из столбцов 8-11, и содержит операционные блоки и обмотки.

Ввод инструкций в программируемую решетку

Ступень в языке лестничной логики снабжена программируемой решеткой, размером семь на одиннадцать, которая начинается с первой ячейки в левом верхнем углу решетки. Программирование состоит из ввода инструкции в ячейку решетки. Инструкции проверки, сравнения и функции вводятся в ячейки проверочной зоны и выравниваются по левому краю. Проверочная логика непрерывно снабжает зону действий, в которую введены и выровнены по правому краю катушек, числовые операции и инструкции управления потоками данных программы. Ступень выполняется или решается (проверяются условия, устанавливаются значения на выходах) в пределах решетки, сверху вниз и слева направо.

Заголовок ступени

В добавление к ступени, заголовок показывается непосредственно над самой ступенью. Используйте заголовки для документирования логической цели ступени. Заголовок ступени может содержать следующую информацию:

- Номер ступени
- Метки (%Li)
- Объявления подпрограмм (Sri:)
- Название ступени
- Комментарии

Для получения подробной информации об использовании заголовков для документирования ваших программ, см. *Программная документация, стр. 228*.

Блоки лестничных диаграмм

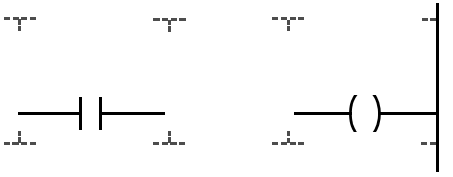
Введение

Лестничные диаграммы состоят из блоков, включающих в себя процесс выполнения программы и следующие функции:

- Контакты
- Катушки
- Команды процесса выполнения программы
- Функциональные блоки
- Блоки сравнения
- Операционные блоки

Контакты, катушки и процесс выполнения программы

Контакты, катушки и команды процесса выполнения программы (Jump и Call) занимают одну ячейку программируемой решетки. Функциональные блоки, блоки сравнения и операционные блоки занимают несколько ячеек.
Пример контакта и катушки:



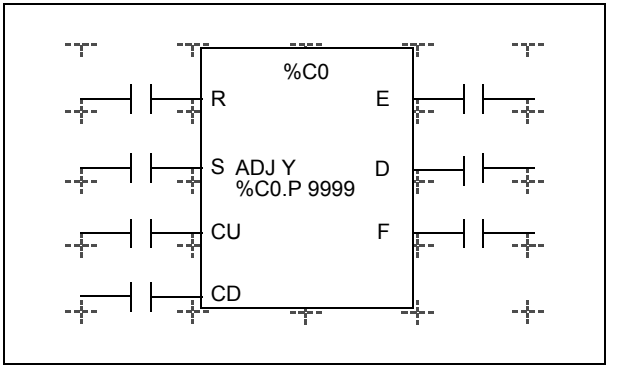
Контакт Катушка

Функциональные блоки

Функциональные блоки располагаются в зоне проверки программируемой решетки. Блок должен располагаться в первой строке; никакие лестничные инструкции и непрерывные линии не могут изображаться выше или ниже функционального блока. На входы блока поступают лестничные инструкции проверки, а на выходах образуются проверочные команды и/или команды действия.

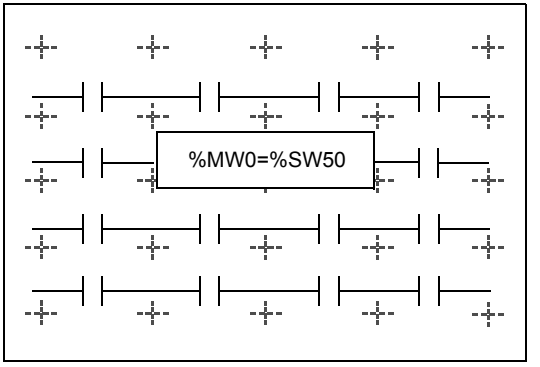
Функциональные блоки вертикально ориентированы и занимают два столбца и четыре строки программируемой решетки.

На следующем рисунке показан пример функционального блока счетчика.



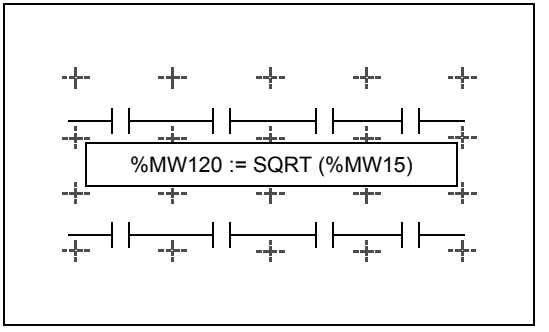
Блоки сравнения

Блоки сравнения размещаются в зоне проверки программируемой решетки. Блок может располагаться в любой строке или столбце зоны проверки, пока вся команда находится в пределах зоны. блоки сравнения горизонтально ориентированы и занимают два столбца и одну строку программируемой решетки.
На следующем рисунке показан пример блока сравнения.



Операционные блоки

Операционные блоки размещаются в зоне действий программируемой решетки. Блок может располагаться в любой строке зоны действий. Инструкция выравнивается по правому краю; она появляется справа и заканчивается в последнем столбце.
Операционные блоки горизонтально ориентированы и занимают четыре столбца в одной строке программируемой решетки.
На следующем рисунке показан пример операционного блока.



Графические элементы языка лестничной логики

Введение

Инструкции в лестничных диаграммах состоят из графических элементов.
.

Контакты

Графические элементы контакта программируются в зоне проверки и занимают одну ячейку (одну строку по высоте и один столбец по ширине).

Название	Графический элемент	Инструкция	Функция
Нормально разомкнутый контакт		LD	Контакт замкнут, когда битовый объект, который им управляет равен 1.
Нормально замкнутый контакт		LDN	Контакт замкнут, когда битовый объект, который им управляет равен 0.
Контакт обнаружения переднего фронта		LDR	Передний фронт: обнаруживает изменение управляющего битового объекта с 0 на 1.
Контакт обнаружения заднего фронта		LDF	Задний фронт: обнаруживает изменение управляющего битового объекта с 1 на 0.

Элементы связи

Графические элементы связи используются для соединения графических элементов проверки и действия.

Название	Графический элемент	Функция
Горизонтальная связь		Используются для соединения элементов проверки и элементов действия, расположенных последовательно между двумя шинами питания.
Вертикальная связь		Используются для параллельного соединения элементов действия и элементов проверки.

Катушки

Графические элементы катушек программируются в зоне действий и занимает одну ячейку (одну строку по высоте и один столбец по высоте).

Название	**Графический элемент**	**Инструкция**	**Функция**
Прямая обмотка		ST	Устанавливает соответствующий битовый объект в значение, равное результату, полученному в зоне проверки.
Обратная обмотка		STN	Устанавливает соответствующий битовый объект в значение, равное инверсии от результата, полученного в зоне проверки.
Устанавливающая обмотка		S	Устанавливает соответствующий битовый объект в 1, когда результат, полученный в зоне проверки, = 1.
Сбрасывающая обмотка		R	Устанавливает соответствующий битовый объект в 0, когда результат, полученный в зоне проверки, = 1.
Переход или вызов подпрограммы	->>%Li ->>%SRi	JMP SR	Соединяет с помеченной инструкцией, расположенной до или после текущей ступени.
Обмотка, реагирующая на перепад			Применяется в языке Grafcet, используется для программирования условий, связанных с переходом на следующую ступень.
Главное реле управления		MCS MCR	Изменяет процесс выполнения программы.
Возврат из подпрограммы	<RET>	RET	Размещается в конце подпрограммы для возврата в основную программу.
Остановка программы	<END>	END	Определяет конец программы.

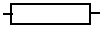
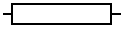
Функциональные блоки

Графические элементы функциональных блоков программируются в зоне проверки, и требуют четыре строки и два столбца (за исключением очень быстрых счетчиков, которым необходимо пять строк и два столбца).

Название	Графический элемент	Функция
Таймеры, счетчики, регистры и т.д.		Каждый из функциональных блоков использует входы/выходы для соединения с другими графическими элементами. Примечание: выходы функционального блока не могут быть связаны друг с другом (вертикальное короткое замыкание).

Операционные блоки и блоки сравнения

Блоки сравнения программируются в зоне проверки, а операционые блоки программируются в зоне действий.

Название	Графический элемент	Функция
Блок сравнения		Позволяет сравнивать два операнда, выход принимает значение, равное 1, если при сравнении получен истинный результат. Размер: 1строка, 2столбца.
Операционный блок		Выполняет арифметические и логические операции. Размер: 1строка, 4 столбца.

Специальные инструкции языка лестничной логики OPEN и SHORT

Введение

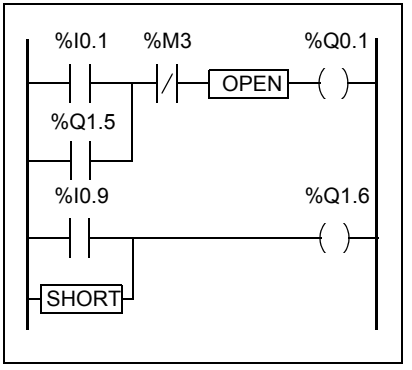
Инструкции OPEN и SHORT предоставляют удобный метод поиска и устранения неисправностей в программе, написанной на языке лестничной логики. Эти специальные инструкции изменяют логику ступени путем либо закорачивания, либо продления непрерывности ступени, как объяснено в следующей таблице.

Инструкция	Описание	Инструкция языка списков инструкций
OPEN	Создает нарушение непрерывности ступени, независимо от результатов последней логической операции.	AND 0
SHORT	Обеспечивает непрерывность выполнения ступени независимо от результатов последней логической операции.	OR 1

В программировании на языке списка инструкций, инструкции AND и OR используются для создания инструкций OPEN и SHORT, использующих прямые значения 0 и 1 соответственно.

Примеры

На следующем рисунке приведены примеры использования инструкций OPEN и SHORT.



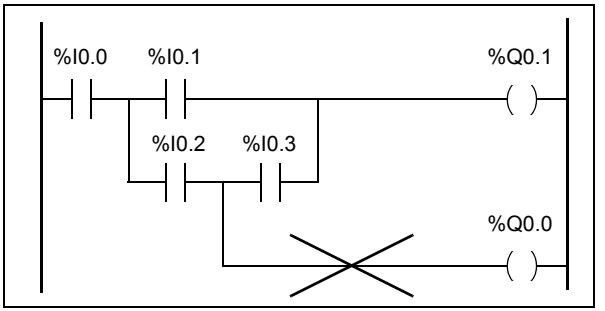

```
LD      %I0.1
OR      %Q1.5
ANDN    %M3
AND     0
ST      %Q0.1
LD      %I0.9
OR      1
ST      %Q1.6
```

Советы по программированию

Обработка переходов в программе	Используйте переходы в программе очень аккуратно, чтобы избежать длинных циклов, которые увеличивают время сканирования. Избегайте переходов к инструкциям, которые расположены до инструкции перехода.
Программирование выходов	Биты выходов, подобно внутренним битам, должны изменятся в программе только однажды . В случае битов выходов только последнее просмотренное значение учитывается при обновлении выходов.
Использование встроенных датчиков чрезвычайной остановки	Датчики, используемые непосредственно для чрезвычайной остановки, не должны обрабатываться контроллером. Они должны быть подсоединены непосредственно к соответствующим выходам.
Обработка включения питания	Сделайте условие включения питания при ручном управлении. Автоматический перезапуск установки может привести к выполнению оборудованием непредсказуемой операции (используйте системные биты %S0, %S1 и %S9) .
Управление временем и блоком планировщика	Должно быть проверено состояние системного бита %S51, который показывает сбои RTC.
Проверка синтаксиса и ошибок	Когда программа введена, TwidoSoft проверяет синтаксис инструкций, операндов и связи между ними.

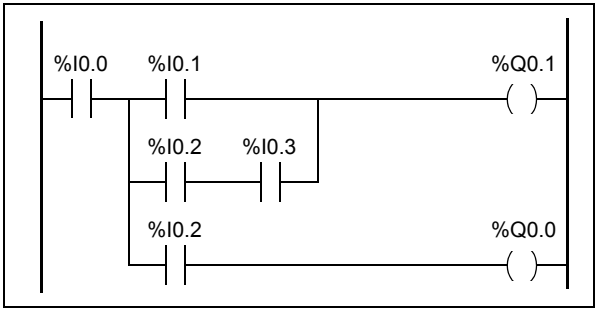
Дополнительные примечания по использованию круглых скобок

Операции присваивания не должны записываться без круглых скобок:



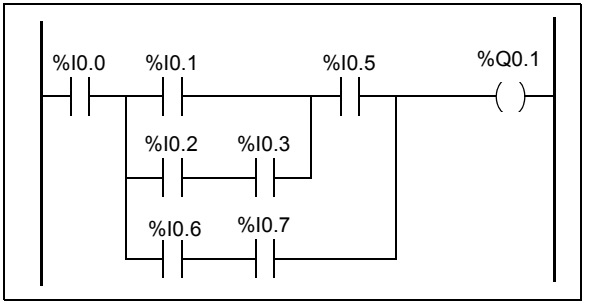

```
LD      %I0.0
AND     %I0.1
OR(     %I0.2
ST      %Q0.0
AND     %I0.3
)
ST      %Q0.1
```

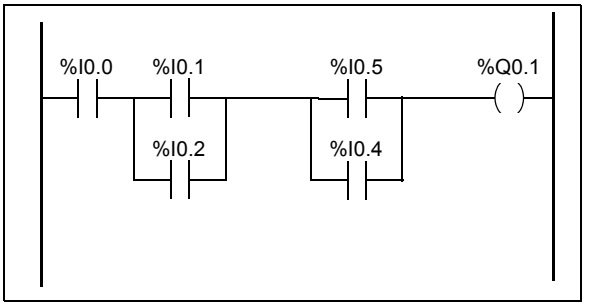
Для того, чтобы выполнить ту же самую функцию, должны быть запрограммированы следующие уравнения:



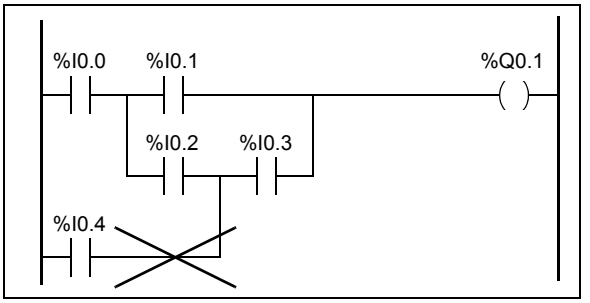

```
LD      %I0.0
MPS
AND(    %I0.1
OR(     %I0.2
AND     %I0.3
)
)
ST      %Q0.1
MPP
AND     %I0.2
ST      %Q0.0
```

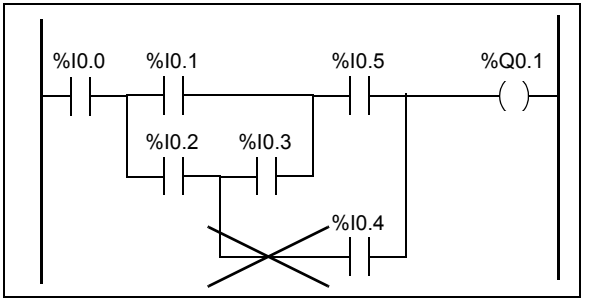
Если несколько контактов соединены параллельно, то они должны быть вложены друг в друга или полностью разделены:



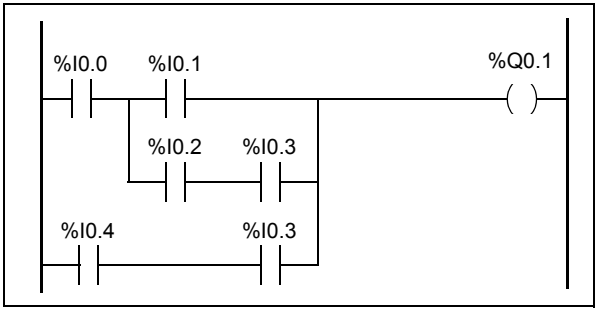


Следующие схемы не могут быть запрограммированы:

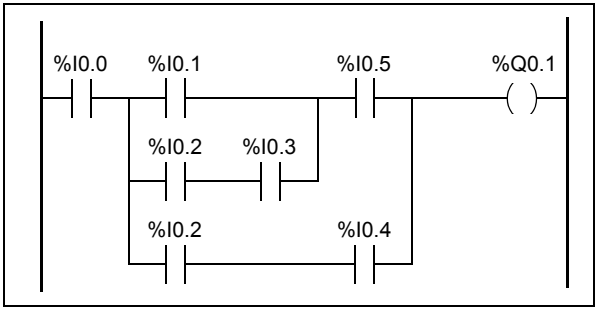




Чтобы выполнить эквивалентные им схемы, они должны быть модифицированы следующим образом:




```
LD      %I0.0
AND(    %I0.1
OR(     %I0.2
AND     %I0.3
)
)
OR(     %I0.4
AND     %I0.3
)
ST      %Q0.1
```




```
LD      %I0.0
AND(    %I0.1
OR(     %I0.2
AND     %I0.3
)
AND     %I0.5
OR(     %I0.2
AND     %I0.4
)
)
ST      %Q0.1
```

Обратимость языков лестничной логики и списка инструкций

Введение

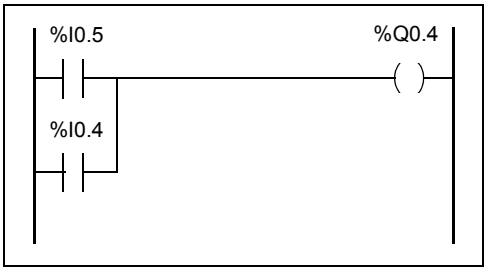
Обратимость программ это особенность программного обеспечения TwidoSoft, которая позволяет преобразовывать приложения языка лестничной логики в приложение языка списка инструкций и наоборот.
Используйте TwidoSoft, чтобы установить отображение программ по умолчанию: либо лестничный формат, либо формат списка инструкций (установкой предпочтений пользователя). TwidoSoft также может использоваться для переключения вида от лестничного к списку инструкций и наоборот.

Понятие обратимости

Ключом к пониманию особенности программной обратимости является исследование отношений лестничных ступеней и связанных инструкций последовательности языка списка инструкций:·

- **Лестничная ступень**: совокупность лестничных команд, составляющих логическое выражение.
- **Последовательность языка списка инструкций**: совокупность команд программирования языка списков инструкций, которые соответствуют лестничным командам и представляют такое же логическое выражение

На следующем рисунке показана обычная лестничная ступень и эквивалентная ей логика программы, выраженная в последовательности команд языка списка инструкций.




```
LD    %I0.5
OR    %I0.4
ST    %Q0.4
```

Приложение хранится внутри, как команды языка списка инструкций, независимо от того, на каком из языков, лестничной логики или списка инструкций, она написана. TwidoSoft использует преимущества сходства структуры программы на языке лестничной логики и языке списка инструкций и использует внутренней образ программы на языке списка инструкция для отображения ее в средствах просмотра лестничного языка и языка списка инструкций и редакторах, либо в виде программы языка списка инструкций (ее основная форма), либо графически, в виде Лестничной диаграммы, в зависимости от предпочтения пользователя.

Обеспечение обратимости

От программы, написанной на языке лестничной логики, всегда можно перейти к программе списка инструкций. Однако, некоторая логика языка списка инструкций может не перенестись в язык лестничной логики. Чтобы гарантировать обратимость от языка списка инструкций к языку лестничной логики важно изучить набор рекомендаций по программированию на языке списка инструкций, рассмотренных в разделе *Рекомендации по обратимости языков*, *стр.226*.

Рекомендации по обратимости языков

Инструкции, необходимые для обратимости

Структура обратимого функционального блока языка списка инструкций требует использования следующих инструкций:

- **BLK** отмечает начало блока и определяет начало ступени и начало входной части блока.
- **OUT_BLK** отмечает начало выходной части блока.
- **END_BLK** отмечает конец блока и ступени.

Использование обратимых инструкций функционального блока не обязательно для должным образом функционирующей программы языка списка инструкций. В языке списка инструкций возможно программирование некоторых команд, которые не являются обратимыми. Описание программирования необратимых стандартных функциональных блоков, см. *Принципы программирования стандартных функциональных блоков*, *стр.277.*

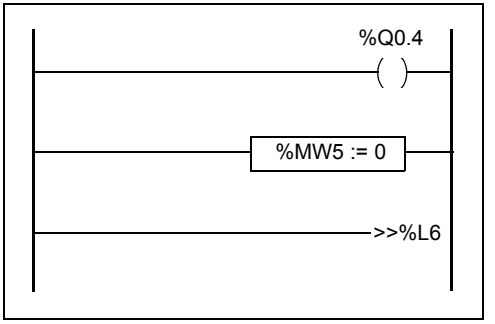
Неэквивалентные инструкции, которых следует избегать.

Избегайте использования некоторых инструкций языка списка инструкций или комбинаций команд и операндов, которые не имеют эквивалента в лестничных диаграммах. Например, инструкция N (инвертирует значение в Булевском аккумуляторе) не имеет эквивалентной лестничной команды.
Следующая таблица определяет все инструкции языка списка инструкций, которые не обратимы в язык лестничной логики.

Инструкция языка списка инструкций	Операнд	Описание
JMPCN	%Li	Переход по условию НЕ
N	нет	Отрицание (НЕ)
ENDCN	нет	Завершение по условию НЕ

Безусловные ступени

Программирование безусловных ступеней также требует изучения следующих рекомендаций по программированию на языке списка инструкций для обеспечения обратимости языка списка инструкций в язык лестничной логики. Безусловные ступени не содержат в себе каких либо условий или проверок. Выходы или команды действия всегда находятся под напряжением или выполняются.
На следующем рисунке представлен пример безусловной ступени и эквивалентной ей последовательности языка списка инструкций.




```
LD    1
ST    %Q0.4
LD    1
[%MW5 := 0]
JMP   %L6
```

Обратите внимание, что каждая из вышеупомянутых безусловных последовательностей языка списка инструкций начинается с команды загрузки 1, за исключением команды JMP. Эта комбинация устанавливает значение Булевского аккумулятора в 1, поэтому значение катушки устанавливается в 1(команда запоминания), а значение %MW5 устанавливается в 0 при каждом сканировании программы. Исключением является команда безусловного перехода языка списка инструкций (JMP %L6), которая выполняется независимо от значения аккумулятора и не нуждается в установке этого значения в 1.

Лестнично-списочные ступени

Если программа языка списка инструкций не полностью обратима в язык лестничной логики, обратимые части отображаются в лестничном представлении, а необратимая отображается в виде лестнично-списочной ступени.
Команды лестнично-списочной ступени представляют собой маленький редактор языка списка инструкций, позволяющий пользователю просматривать и редактировать необратимые части программы.

Программная документация

Документирование Вашей программы

Вы можете документировать Вашу программу добавлением комментариев, используя редакторы языков лестничной логики и списка инструкций:

- Используйте редактор языка списка инструкций для документирования Вашей программы с помощью строк комментариев языка списка инструкций. Эти комментарии могут располагаться на той же строке, что и команда программы, или они могут располагаться на своих собственных строках.
- Используйте редактор языка лестничной логики для документирования Вашей программы с помощью заголовков ступеней. Они располагаются непосредственно над ступенью

TwidoSoft использует эти комментарии для обратимости. Когда происходит обращение программы языка списка инструкций в программу лестничной логики, TwidoSoft использует некоторые из комментариев языка списка инструкций для преобразования в заголовки ступеней в языке лестничной логики. Для этого комментарии, вставленные между последовательностями языка списка инструкций используются для заголовков ступеней.

Пример строки комментариев в языке списка инструкций

Следующий листинг является примером программы, написанной на языке списка инструкций со строками комментариев:

```
---- ( * THIS IS THE TITLE OF THE HEADER FOR RUNG 0 * )
---- ( * THIS IS THE FIRST HEADER COMMENT FOR RUNG 0 * )
---- ( * THIS IS THE SECOND HEADER COMMENT FOR RUNG 0 * )
     0   LD  % I0. 0  ( * THIS IS A LINE COMMENT *)
     1   OR  %I0. 1  ( * A LINE COMMENT IS IGNORED WHEN REVERSING TO LADDER * )
     2   ANDM  %M10
     3   ST         M101
---- ( * THIS IS THE HEADER FOR RUNG 1 * )
---- ( * THIS RUNG CONTAINS A LABEL * )
---- ( * THIS IS THE SECOND HEADER COMMENT FOR RUNG 1 * )
---- ( * THIS IS THE THIRD HEADER COMMENT FOR RUNG 1 * )
---- ( * THIS IS THE FOURTH HEADER COMMENT FOR RUNG 1 * )
     4   % L5:
     5   LD  %M101
     6   [ %MW20 := %KW2 * 16 ]
---- ( * THIS RUNG ONLY CONTAINS A HEADER TITLE * )
     7   LD  %Q0. 5
     8   OR  %I0.  3
     9   ORR  I0.  13
    10   ST  %Q0.5
```

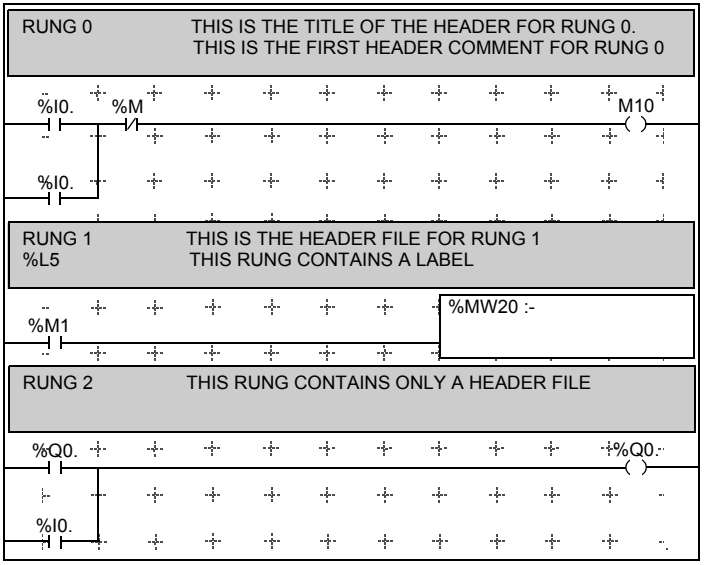
Обратимость комментариев языка списка инструкций в комментарии языка лестничной логики

Когда команды языка списка инструкций обращены в лестничную диаграмму, строки комментариев языка списка инструкций изображаются в редакторе языка лестничной логики согласно следующим правилам:

- Первый комментарий, находящийся на строке отдельно, назначается заголовком ступени.
- Все комментарии, найденные после первого, становятся телом ступени.
- После того, как все строки тела заголовка будут заняты, остальные линии комментариев, расположенных между последовательностями языка списка инструкций, игнорируются, как и любые комментарии, найденные на строках языка списка инструкций, которые также содержат инструкции.

Пример комментариев в заголовке ступени

Программа на языке лестничной логики с комментариями в заголовке.



Обратимость комментариев языка лестничной логики в комментарии языка списка инструкций

Когда лестничная диаграмма обращена в инструкции языка списка инструкций, комментарий заголовка ступени отображается в редакторе языка списка инструкций согласно следующим правилам:

- Любые комментарии заголовка ступени вставляются в связанную последовательность языка списка инструкций.
- Любые метки(%Li) или объявления подпрограмм(Sri:) размещаются на следующей линии после заголовка и непосредственно перед последовательностью языка списка инструкций.
- Если программа языка списка инструкций была обращена в программу лестничной логики, то все комментарии, которые были проигнорированы, вновь появятся в редакторе языка списка инструкций

Язык списка инструкций

12

Обзор

Предмет

В этой главе описано программирование с использованием языка списка инструкций.

Содержание главы

Эта глава содержит следующие темы:

Обзор программ на языке списка инструкций

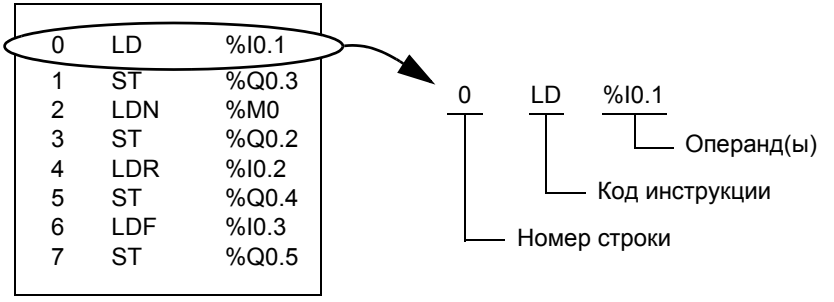
Введение

Программа, написанная на языке списка инструкций, состоит из набора инструкций, выполняемых контроллером последовательно. Каждая инструкция представляет собой одну программную строку и состоит из трех компонент:

- Номер строки
- Код инструкции
- Операнд(ы)

Пример программы на языке списка инструкций

Ниже приведен пример программы на языке списка инструкций.



Номер строки

Номер строки генерируется автоматически при вводе инструкции. Пустые строки и строки с комментариями не имеют номера.

Код инструкции

Код инструкции это символ, идентифицирующий для оператора операцию, которая будет выполнена над операндами.
Например, в простой программе, приведенной выше, LD это аббревиатура кода инструкции LOAD. Инструкция LOAD загружает значение операнда %I0.1 во внутренний регистр, называемый аккумулятором.
Есть два основных типа инструкций:

- Инструкции проверки
  Устанавливают или проверяют необходимые условия для выполнения действия. Например: LOAD(LD) и AND.
- Инструкции действий
  Выполняют действие, как результат установленных условий. Например инструкции присваивания такие как STORE (ST) и RESET (R).

Операнд

Операнд это число, адрес или символ, представляющий значение, которое программа может использовать в инструкциях. Например, в программе приведенной выше, операнд %I0.1 это адрес, указывающий на значение входа контроллера. Инструкция может иметь от нуля до трех операндов, в зависимости от типа кода инструкции.

Операнды могут представлять следующее:

- Входы и выходы контроллера, такие как сенсоры, кнопки и реле.
- Предопределенные системные функции, такие как таймеры и счетчики.
- Арифметические, логические, численные операции и операции сравнения.
- Внутренние переменные контроллера, такие как биты и слова.

Выполнение инструкций

Введение

Инструкции языка списка инструкций имеют только один явно указанный операнд, второй подразумевается. Подразумеваемый операнд это значение булевского аккумулятора. Например, в инструкции LD %I0.1, %I0.1 это явно указанный операнд. Подразумеваемый операнд содержится в аккумуляторе и будет перезаписан значением %I0.1.

Выполнение

Инструкции выполняют указанную операцию над содержимым аккумулятора и явного операнда, и записывают в аккумулятор результат. Например, AND %I1.2 выполняет операцию "логическое И" между содержимым аккумулятора и входом 1.2, после записывает результат в аккумулятор.
Все булевские инструкции, кроме Load, Store и Not, оперируют двумя операндами. Значение операндов может быть либо "Истина", либо "Ложь", выполнение инструкций также приводит к одному значению: либо "Истина", либо "Ложь". Инструкция Load загружает значение операнда в аккумулятор, инструкция Store перемещает значение аккумулятора в операнд. Инструкция Not не имеет явного операнда и просто инвертирует состояние аккумулятора.

Поддерживаемые инструкции языка списка

В следующей таблице показан набор инструкций языка Списка инструкций:

Тип инструкции	Пример	Функция
Битовая инструкция	LD %M10	Чтение внутреннего бита %M10
Инструкция блока	IN %TM0	Запуск таймера %TM0
Инструкция слова	[%MW10 := %MW50+100]	Операция сложения
Программная инстр.	SR5	Вызов подпрограммы #5
Интрукция Grafcet	-*-8	Шаг #8

Инструкции языка

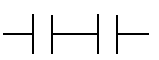
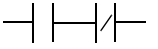
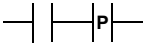
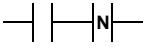
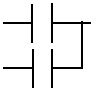
Введение

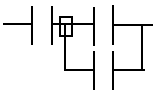
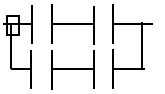
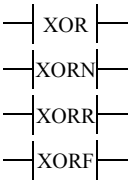
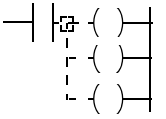
Язык состоит из следующих типов инструкций:
- Инструкции проверки
- Инструкции действий
- Инструкции функциональных блоков

В этом разделе определены и описаны инструкции Twido для программирования на языке списка инструкций.

Инструкции проверки

В следующей таблице описаны инструкции проверки.

Название	Эквивалентный графический элемент	Функция
LD		Булевский результат равен значению операнда.
LDN		Булевский результат равен инвертированному значению операнда.
LDR		Булевский результат изменяется в 1 при выявлении перехода операнда из 0 в 1 (по фронту).
LDF		Булевский результат изменяется в 1 при выявлении перехода операнда из 1 в 0 (по спаду).
AND		Булевский результат равен логическому "И" между булевским результатом предыдущей инструкции и значением операнда.
ANDN		Булевский результат равен логическому "И" между булевским результатом предыдущей инструкции и инверсным значением операнда.
ANDR		Булевский результат равен логическому "И" между булевским результатом предыдущей инструкции и выявлении перехода операнда из 0 в 1 (1 = фронт).
ANDF		Булевский результат равен логическому "И" между булевским результатом предыдущей инструкции и выявлении перехода операнда из 1 в 0 (1 = спад).
OR		Булевский результат равен логическому "ИЛИ" между булевским результатом предыдущей инструкции и значением операнда.

Название	Эквивалентный графический элемент	Функция
AND(		Логическое И (8 уровней вложения).
OR(		Логическое ИЛИ (8 уровней вложения).
XOR, XORN, XORR, XORF		Исключающее ИЛИ.
MPS MRD MPP		Переключение на катушки.
N	-	Отрицание ("НЕ")

Инструкции действий

В следующей таблице описаны инструкции действий.

Название	Эквивалентный графический элемент	Функция
ST		Связанный операнд принимает значение, равное результату, полученному в зоне проверки.
STN		Связанный операнд принимает значение, обратное результату, полученному в зоне проверки.
S		Связанный операнд устанавливается в 1, когда результат, полученный в зоне проверки, равен 1.
R		Связанный операнд устанавливается в 0, когда результат, полученный в зоне проверки, равен 1.

Название	Эквивалентный графический элемент	Функция
JMP	->>%Li	Безусловно подключает в начало или конец помеченной последовательности.
SRn	->>%SRi	Подключение к началу подпрограммы.
RET	<RET>	Возврат из подпрограммы.
END	<END>	Конец программы.
ENDC	<ENDC>	Конец условной программы при булевом булевском результате 1.
ENDCN	<ENDCN>	Конец условной программы при булевом булевском результате 0.

Инструкции функциональных блоков

В следующей таблице описаны инструкции функциональных блоков.

Название	Эквивалентный графический элемент	Функция
Таймеры, счетчики, регистры и т.д.		Существуют инструкции для управления каждым из функциональных блоков. Для прямого соединения входов и выходов функциональных блоков используется специальная структурная форма. **Примечание:** Выходы функциональных блоков не могут быть соединены между собой (вертикальное КЗ).

Использование круглых скобок

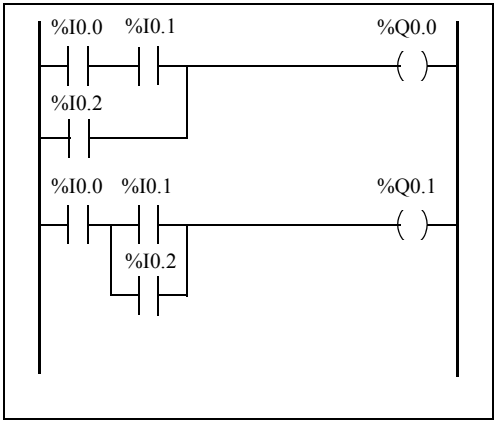
Введение

В логических инструкциях AND и OR, круглые скобки используются также, как разветвление в Лестничных Редакторах. Связывание круглых скобок с инструкциями происходит следующим способом:

- Открывающая круглая скобка связывается с соответствующей инструкцией AND или OR.
- Закрывающая круглая скобка является инструкцией, обязательной для каждой открывающей скобки.

Пример использование инструкции AND (“И”)

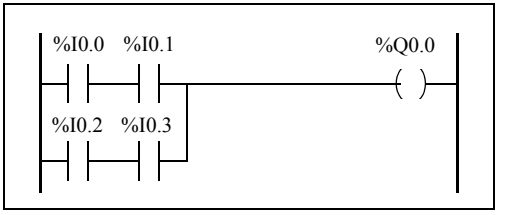
Пример использования круглых скобок с инструкцией AND показан на следующих рисунках: AND(...).




```
LD      %I0.0
AND     %I0.1
OR      %I0.2
ST      %Q0.0

LD      %I0.0
AND(    %I0.1
OR      %I0.2
)
ST      %Q0.1
```

Пример использования инструкции OR (“ИЛИ”)

Пример использования круглых скобок с инструкцией OR показан на следующих рисунках: OR(...).




```
LD      %I0.0
AND     %I0.1
OR(     %I0.2
AND     %I0.3
)
ST      %Q0.0
```

Модификаторы

В следующей таблице приведен список модификаторов, которые могут использоваться с круглыми скобками.

Модификатор	Функция	Пример
N	Отрицание	AND(N или OR(N
F	Спад	AND(F или OR(F
R	Фронт	AND(R или OR(R
[	Сравнение	См. *Инструкции сравнения, стр. 305*

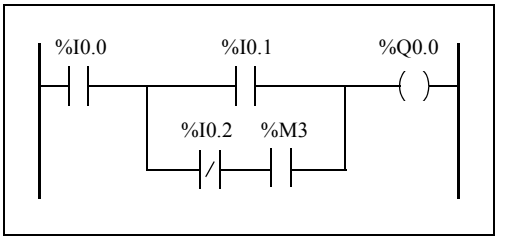
Вложенные круглые скобки

Возможно вкладывать до восьми уровней круглых скобок. Соблюдайте следующие правила при вложении круглых скобок:

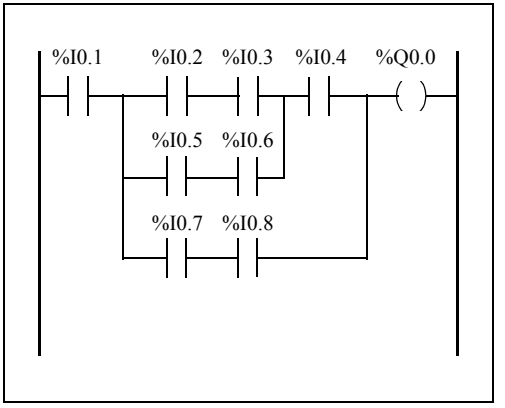
- Каждая открывающая скобка должна иметь соответствующую закрывающую скобку.
- Метки (%Li), подпрограммы (SRi), инструкции перехода (JMP) и инструкции функциональных блоков не могут размещаться в выражениях между круглыми скобками.
- Инструкции хранения - ST, STN, S и R не могут быть запрограммированы между круглыми скобками.
- Стековые инструкции MPS, MRD и MPP не могут использоваться между круглыми скобками.

Пример вложения круглых скобок

На следующем рисунке показан пример вложения круглых скобок.




```
LD      %I0.0
AND(    %I0.1
OR(N    %I0.2
AND     %M3
)
)
ST      %Q0.0
```




```
LD      %I0.1
AND(    %I0.2
AND     %I0.3
OR(     %I0.5
AND     %I0.6
)
AND     %I0.4
OR(     %I0.7
AND     %I0.8
)
)
ST      %Q0.0
```

Инструкции стека (MPS, MRD, MPP)

Введение

Стековые инструкции используются для обработки катушек в особом порядке. Инструкции MPS, MRD и MPP используют временную область хранения, называемую стек, которая может хранить до восьми битов булевых выражений.

Примечание: Эти инструкции не могут использоваться внутри выражений, расположенных между круглыми скобками.

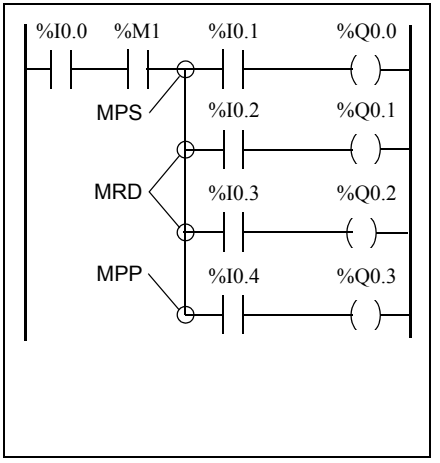
Действие инструкций стека

В следующей таблице описано действие трех инструкций стека.

Инструкция	Описание	Функция
MPS	Запись данных в стек	Запоминает результат последней логической инструкции (содержимое аккумулятора) в вершине стека и сдвигает остальные элементы стека вниз.
MRD	Чтение данных из стека	Считывает вершину стека в аккумулятор.
MPP	Выталкивание данных из стека	Копирует значение вершины стека в аккумулятор и сдвигает остальные элементы стека вверх.

Примеры инструкций стека

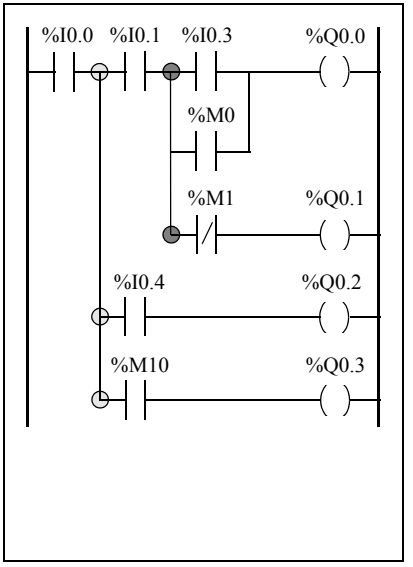
На следующих рисунках приведены примеры использование инструкций стека.




```
LD      %I0.0
AND     %M1
MPS
AND     %I0.1
ST      %Q0.0
MRD
AND     %I0.2
ST      %Q0.1
MRD
AND     %I0.3
ST      %Q0.2
MPP
AND     %I0.4
ST      %Q0.3
```

Примеры работы инструкций стека

На следующих рисунках показано, как работают инструкции стека.




```
LD      %I0.0
MPS
AND     %I0.1
MPS
AND(    %I0.3
OR      %M0
)
ST      %Q0.0
MPP
ANDN    %M1
ST      %Q0.1
MRD
AND     %I0.4
ST      %Q0.2
MPP
AND     %M10
ST      %Q0.3
```

Язык Grafcet

13

Обзор

Предмет

В этой главе описано программирование на языка Grafcet.

Содержание главы

Эта глава содержит следующие темы:

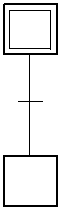
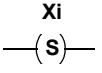
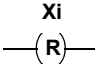
Описание инструкций Grafcet

Введение

Инструкции языка Grafcet в TwidoSoft предлагают простой метод трансляции контрольной последовательности (Grafcet-диаграмма).
Максимальное число Grafcet-шагов определяется типом контроллера Twido. В любой момент времени максимальное число активных шагов ограничивается только общим числом шагов. Для TWDLCAA10DRF и TWDLCAA16DRF доступны шаги с 1 по 62. Шаги 0 и 63 зарезервированы для пред- и пост-обработки. Для всех остальных контроллеров доступны шаги с 1 по 95.

Инструкции Grafcet

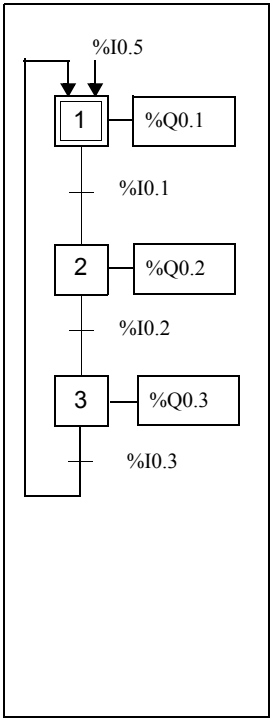
В следующей таблице перечислены все инструкции и объекты, необходимые для программирования Grafcet-диаграмм

Графическое представление (1)	**Транскрипция в языке TwidoSoft**	**Функция**
Рисунок:  **Начальный шаг** **Переход** **Шаг**	=*= i	Начинает первый шаг (2)
	# i	Активизирует шаг i после дезактивирования текущего шага
	-*- i	Начинает шаг i и утверждает связанный переход (2)
	#	Дезактивирует текущий шаг без активации другого шага
	#Di	Дезактивирует шаг i и текущий шаг
	=*= POST	Начинает пост-обработку и заканчивает последовательную обработку
	%Xi	Бит связанный с шагом i, может быть проверен и записан (максимальное число шагов определяется контроллером)
Xi 	LD %Xi, LDN %Xi AND %Xi, ANDN %Xi, OR %Xi, ORN %Xi XOR %Xi, XORN %Xi	Проверяет активность шага i
Xi  **S**	S %Xi	Активизирует шаг i
Xi  **R**	R %Xi	Дезактивирует шаг i

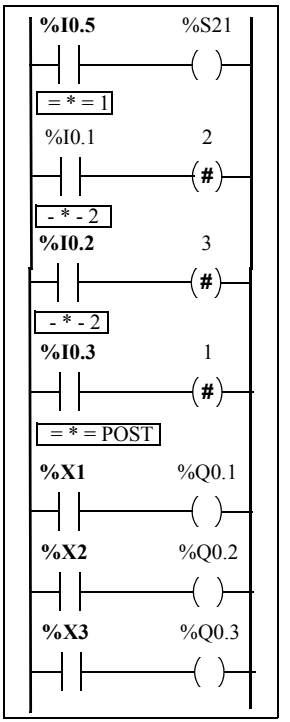
(1) Графическое представление не учитывается.
(2) Первый шаг =*= или -*- показывает начало последовательного выполнения и конец предварительной обработки.

Примеры Grafcet

Линейные последовательности:



Не поддерживается

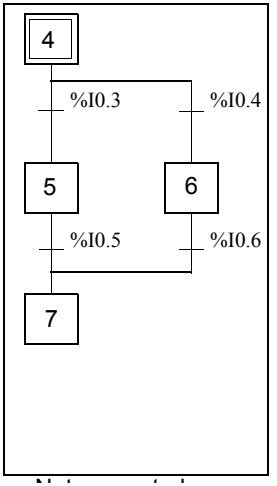


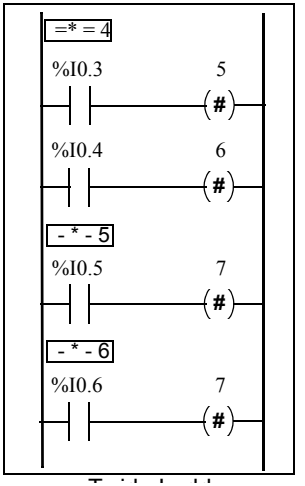
ПрограммаTwido на языке Лестн. логики.

```
LD      %I0.5
ST      %S21
=*=     1
LD      %I0.1
 #      2
-*-     2
LD      %I0.2
 #      3
-*-     3
LD      %I0.3
 #      1
=*=     POST
LD      %X1
ST      %Q0.1
LD      %X2
ST      %Q0.2
LD      %X3
ST      %Q0.3
```

ПрограммаTwido на языке Списка инстр.

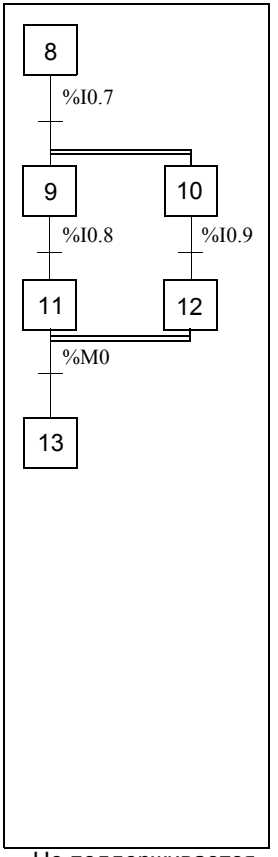
Альтернативные последовательности:



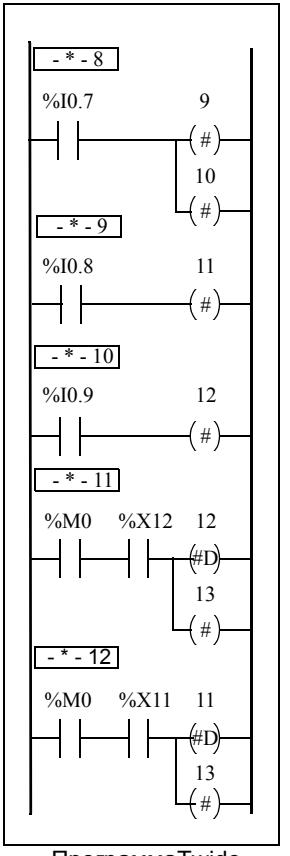



```
=*=     4
 LD     %I0.3
 #      5
 LD     %I0.4
 #      6

 -*-    5
 LD     %I0.5
 #      7

-*-     6
 LD     %I0.6
 #      7
```

Синхронные последовательности:



Не поддерживается



ПрограммаTwido
на языке Лестн. логики.

```
-*-      8
LD       %I0.7
#        9
#        10

-*-      9
LD       %I0.8
#        11

-*-      10
LD       %I0.9
#        12

-*-      11
LD       %M0
AND      %X12
#D       12
#        13

-*-      12
LD       %M0
AND      %X11
#D       11
#        13
```

ПрограммаTwido
на языке Списка инстр.

Примечание: Чтобы Grafcet-диаграммы были рабочими, необходимо, чтобы хотя один активный шаг был объявлен, используя инструкцию =*= (начальный шаг), или диаграмма должна быть предварительно установлена во время пред-обработки, используя системный бит %S23 и инструкцию S %Xi.

Описание структуры программы Grafcet

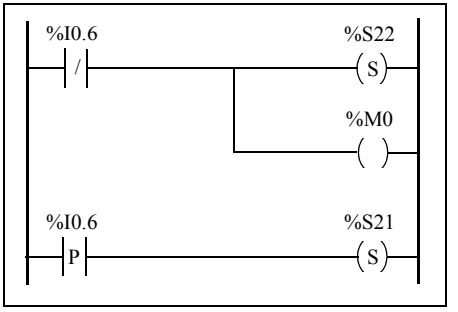
Введение

Программа TwidoSoft на языке Grafcet имеет три части:
- Предварительная обработка
- Последовательная обработка
- Последующая обработка

Предварительная обработка

Предварительная обработка состоит из следующего:
- Включение питания
- Ошибки
- Смена режима работы
- Предустановочные шаги Grafcet
- Входная логика

Фронт на входе %I0.6 устанавливает бит %S21 в 1. Это блокирует активные шаги и запускает неактивные шаги.




```
000   LDN   %I0.6
001   S     %S22
002   ST    %M0
003   LDR   %I0.6
004   S     %S21
```

Предварительная обработка начинается с первой строки программы и заканчивается при первой встрече инструкции *=*=* или инструкции *-*-*. Три системных бита предопределены для управления в Grafcet: %S21,%S22 и %S23. Каждый из этих битов устанавливается в 1(при необходимости) приложением, обычно, во время предварительной обработки. В конце предварительной обработки соответствующая функция выполняется системой, и системный бит устанавливается в 0

Системный бит	Название	Описание
%S21	Инициализация Grafcet	Все активные шаги дезактивируются, и активируются начальные шаги.
%S22	Повторная инициализация Grafcet	Все шаги дезактивируются.

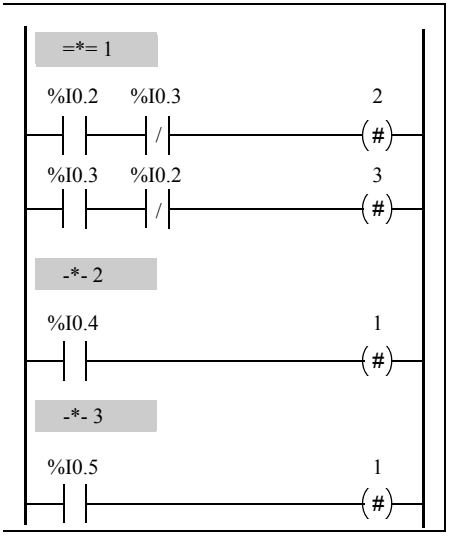
Системный бит	Название	Описание
%S23	Препозиция Grafcet	Этот бит должен быть установлен в 1, если %Xi объекты явно записаны приложением в предварительной обработке. Если этот бит поддерживается в 1 во время предварительной обработки без явного изменения %Xi объектов, то Grafcet замораживается (обновления не учитываются).

Последовательная обработка

Последовательностная обработка диаграммы (инструкций, представляющих диаграмму):

- Шаги
- Действия, связанные с шагами
- Переходы
- Переходные состояния

Пример:




```
005    =*=     1
006    LD      %I0.2
007    ANDN    %I0.3
008    #       2
009    LD      %I0.3
010    ANDN    %I0.2
011    #       3
012    -*-     2
013    LD      %I0.4
014    #       1
015    -*-     3
016    LD      %I0.5
017    #       1
```

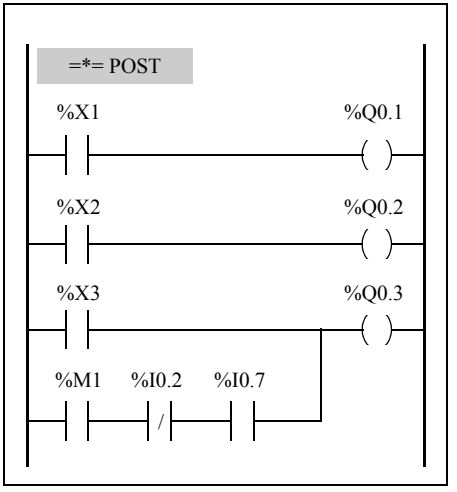
Последовательная обработка заканчивается инструкцией "= * = POST" или концом программы.

Последующая обработка

Последующая обработка состоит из:

- Команд от последовательной обработки для контролирования выходов
- Безопасной блокировки определенных выходов

Пример:




```
018   =*=     POST
019   LD      %X1
020   ST      %Q0.1
021   LD      %X2
022   ST      %Q0.2
023   LD      %X3
024   OR(     %M1
025   ANDN    %I0.2
026   AND     %I0.7
027   )
028   ST      %Q0.3
```

Действия, связанные с шагами Grafcet

Введение

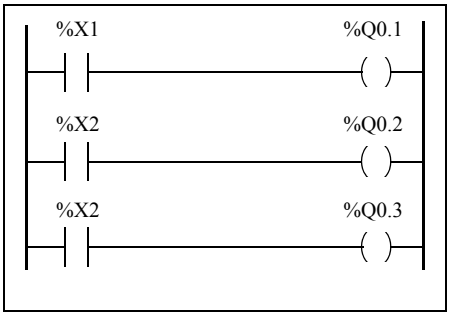
Программы TwidoSoft на Grafcet представляют два пути программирования действий, связанных с шагами:

- В части последующей обработке
- В пределах самих инструкций списка или ступеней лестницы

Связывание действий в последующей обработке

Для безопасности или в случае ограниченности работоспособности предпочтительно программировать действия в части последующей обработки приложения Grafcet. Вы можете использовать инструкции Set и Reset языка Списка Инструкций или возбуждать катушки в Лестничной программе для активации Grafcet-шагов (%Xi).

Пример:

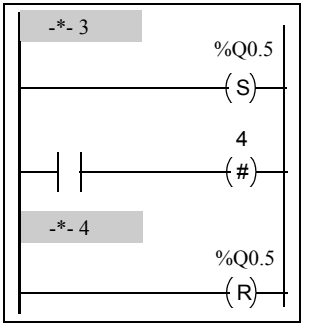



```
018   =*=   POST
019   LD    %X1
020   ST    %Q0.1
021   LD    %X2
022   ST    %Q0.2
023   LD    %X3
024   ST    %Q0.3
```

Связывание действий от программы

Вы можете программировать действия, связанные с шагами используя язык списка или язык лестничной логики. В этом случае, инструкция языка списка или ступень языка лестничной логики не сканируются до тех пор, пока шаг не активирован. Это наиболее эффективный, читабельный и изменяемый путь использования Grafcet.

Пример:




```
020   -*-   3
021   LD    1
022   S     %Q0.5
023   LD    %M10
024   #     4
025   -*-   4
026   LD    1
027   R     %Q0.5
028   ...
029   ...
```

Описание инструкций и функций

IV

Обзор

Тема этой части

В этой части представлено подробное описание основных и дополнительных инструкций и системных битов и слов для языков Twido.

Содержание этой части

Эта часть содержит следующие главы:

Основные инструкции

14

Обзор

Предмет

Эта глава содержит сведения об инструкциях и функциональных блоках, которые используются для создания основных управляющих программ для контроллеров Twido.

Содержание главы

Эта глава содержит следующие темы:

14.1 Логическая обработка

Обзор

Цель этого раздела

Этот раздел представляет введение в булевскую обработку, включая описания и программные принципы для булевских инструкций.

Содержание раздела

Этот раздел содержит следующие темы:

Логические инструкции

Введение

Булевские инструкции можно сравнить с элементами языка лестничной логики. Эти инструкции приведены в следующей таблице.

Элемент	Инструкция	Пример	Описание
Элементы проверки	Инструкция загрузки (LD) эквивалентна открытому контакту.	LD %I0.0	Контакт закрыт, когда бит %I0.0 равняется 1.
Элементы действия	Инструкция хранения (ST) эквивалентна обмотке.	ST %Q0.0	Связанный с элементом битовый объект принимает логическое значение бита аккумулятора (результат предыдущей логики).

Результат булевских операций над тестовыми элементами присваивается элементам действия, как показано следующими инструкциями.

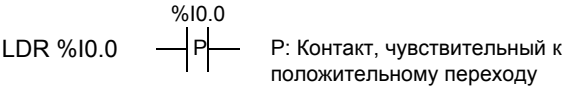
```
LD    %I0.0
AND   %I0.1
ST    %Q0.0
```

Тестирование входов контроллера

Булевские тестовые инструкции могут использоваться для выявления передних и задних фронтов на входах контроллера. Фронт выявляется, когда состояние входа изменяется от 'сканирования n-1' до текущего 'сканирования n'. Этот фронт остается выявленным в течении текущего цикла сканирования.

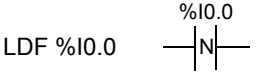
Выявление переднего фронта

Инструкция LDR (load rising edge - загрузка переднего фронта) эквивалентна контакту выявления переднего фронта. Передний фронт означает изменение входной величины с 0 на 1. Контакт, чувствительный к положительному переходу используется для выявления переднего фронта, как показано на диаграмме.



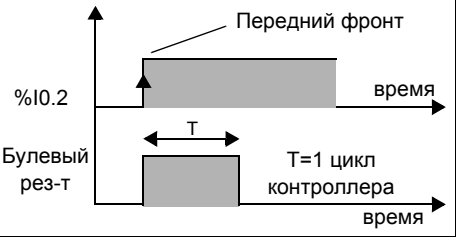
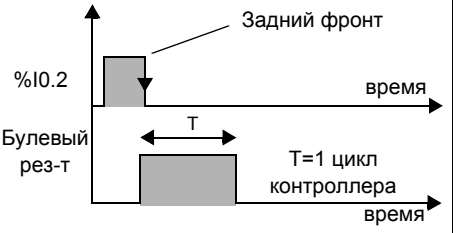
Выявление заднего фронта

Инструкция LDF(load falling edge - загрузка заднего фронта) эквивалентна контакту выявления заднего фронта. Задний фронт свидетельствует об изменении контролируемого входа с 1 на 0. Контакт, чувствительный к отрицательному переходу используется для выявления заднего фронта, как показано на диаграмме.



Выявление фронта

Инструкции и временные диаграммы для выявления фронтов сведены в следующую таблицу:

Фронт	**Инструкция проверки**	**Лестничная диаграмма**	**Временная диаграмма**
Передний фронт	LDR %I0.0	%I0.0 P	
Задний фронт	LDF %I0.0	%I0.0 N	

Примечание: Можно применять инструкции фронта к %Mi внутренним битам.

Формат для описания логических инструкций

Введение

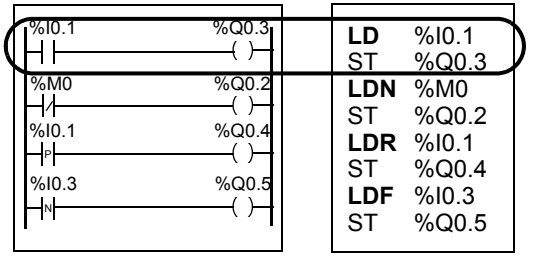
Каждая булевская инструкция в этом разделе описана с использованием следующей информации:

- Краткое описание
- Пример инструкции и соответствующей лестничной диаграммы
- Список разрешенных операндов
- Временная диаграмма

Далее следуют более подробные объяснения того, как булевские инструкции описываются в этом разделе.

Примеры

На следующем рисунке показано, каким образом каждой инструкции сопоставляется пример.



Эквивалентные лестничные диаграммы | Список инструкций

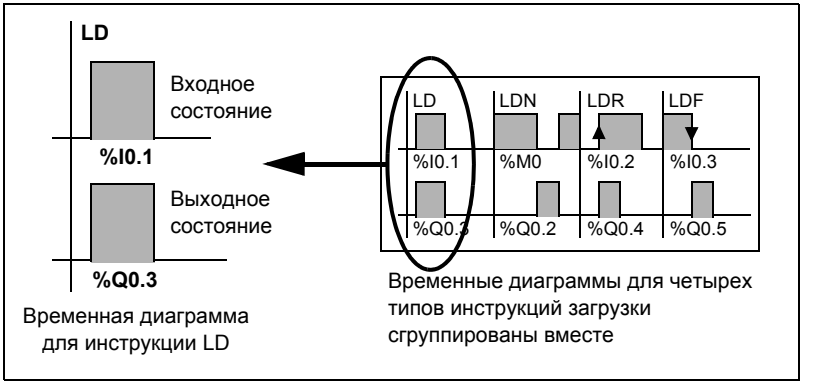
Разрешенные операнды

Следующая таблица определяет типы разрешенных операндов, используемых для булевых инструкций.

Операнд	Описание
0/1	Непосредственное значение 0 или 1
%I	Вход контроллера %Ii.j
%Q	Выход контроллера %Qi.j
%M	Внутренний бит %Mi
%S	Системный бит %Si
%X	Бит шага %Xi
%BLK.x	Бит функционального блока (например, %TMi.Q)
%•:Xk	Бит слова (например, %MWi:Xk)
[	Выражение сравнения (например, [%MWi<1000])

Временные диаграммы

На следующем рисунке показано, как отображаются временные диаграммы для каждой инструкции.



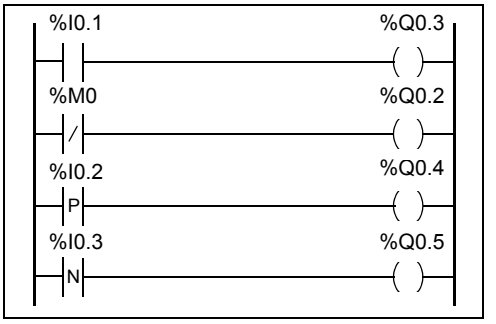
Инструкции загрузки (LD, LDN, LDR, LDF)

Введение

Инструкциям загрузки LD, LDN, LDR и LDF эквивалентны, соответственно, открытый, закрытый контакты и контакты переднего и заднего фронтов (LDR и LDF используются только с входами контроллера и внутренними словами, и для входов slave AS-Interface).

Примеры

На следующих рисунках приведены примеры инструкций загрузки.




```
LD    %I0.1
ST    %Q0.3
LDN   %M0
ST    %Q0.2
LDR   %I0.2
ST    %Q0.4
LDF   %I0.3
ST    %Q0.5
```

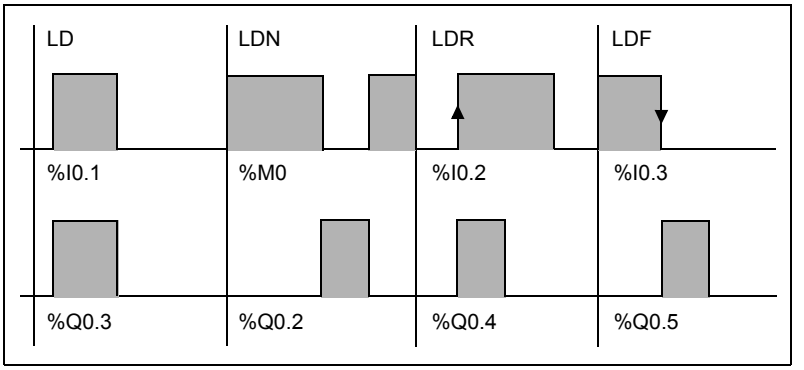
Разрешенные операнды

Следующая таблица содержит типы инструкций загрузки, эквиваленты в лестничной логике и разрешенные операнды.

Инструкция списка	Лестничный эквивалент	Разрешенные операнды
LD		0/1, %I, %IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk,[
LDN		0/1, %I, %IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk,[
LDR		%I, %IA, %M
LDF		%I, %IA, %M

Временная диаграмма

Ниже приведена временная диаграмма для инструкций загрузки.



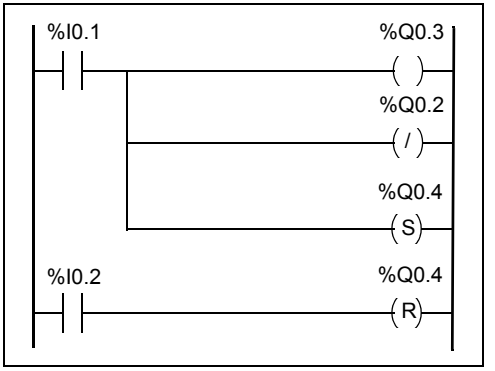
Инструкции присваивания (ST, STN, R, S)

Введение

Инструкциям присваивания ST, STN, S и R соответствуют прямая, обратная катушка, катушка установки и катушка переустановки.

Примеры

На следующих рисунках приведены примеры инструкций присваивания.




```
LD      %I0.1
ST      %Q0.3

STN     %Q0.2
S       %Q0.4

LD      %I0.2
R       %Q0.4
```

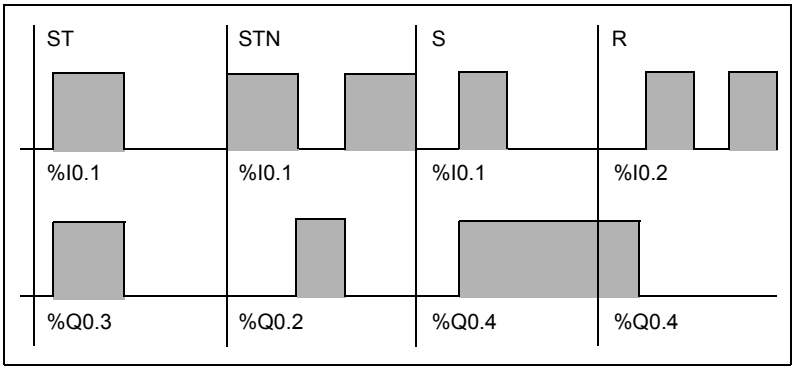
Разрешенные операнды

Следующая таблица содержит типы инструкций присваивания, эквиваленты в лестничной логике и разрешенные операнды

Инструкция списка	Лестничный эквивалент	Разрешенные операнды
ST	( )	%Q,%QA,%M,%S,%BLK.x,%•:Xk
STN	(/)	%Q,%QA%M,%S,%BLK.x,%•:Xk
S	(S)	%Q,%QA,%M,%S,%X,%BLK.x,%•:Xk
R	(R)	%Q,%QA,%M,%S,%X,%BLK.x,%•:Xk

Временная диаграмма

Ниже приведена временная диаграмма для инструкций присваивания..



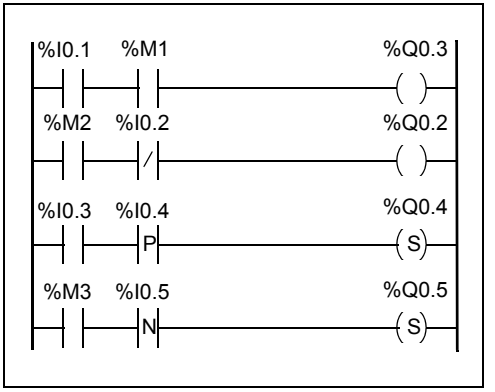
Инструкции логического И (AND, ANDN, ANDR, ANDF)

Введение

Инструкции AND выполняют операцию логического И между операндом (или его инверсией, или его передним или задним фронтом) и булевым результатом предыдущей инструкции.

Примеры

На следующих рисунках приведены примеры инструкции логического И (AND).




```
LD      %I0.1
AND     %M1
ST      %Q0.3
LD      %M2
ANDN    %I0.2
ST      %Q0.2
LD      %I0.3
ANDR    %I0.4
S       %Q0.4
LD      %M3
ANDF    %I0.5
S       %Q0.5
```

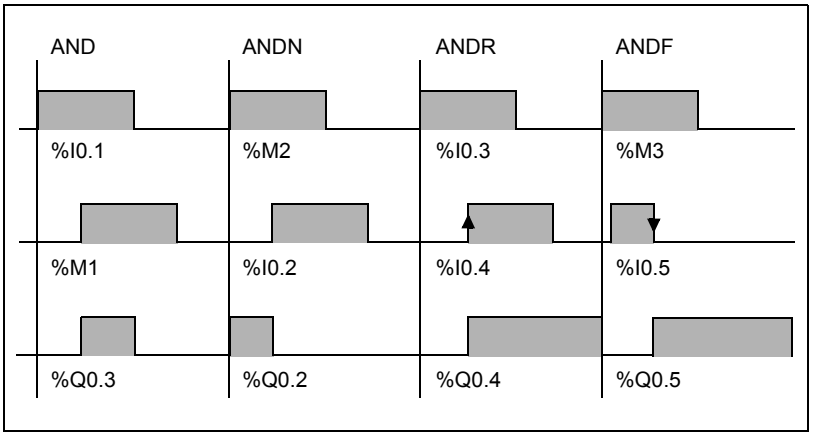
Разрешенные операнды

Следующая таблица содержит типы инструкций логического И, эквива ленты в лестничной логике и разрешенные операнды.

Инструкция списка	Лестничный эквивалент	Разрешенные операнды
AND		0/1, %I, %IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk, [
ANDN		0/1, %I, %IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk, [
ANDR		%I, %IA, %M
ANDF		%I, %IA, %M

Временная диаграмма

Ниже приведена временная диаграмма для инструкции AND.



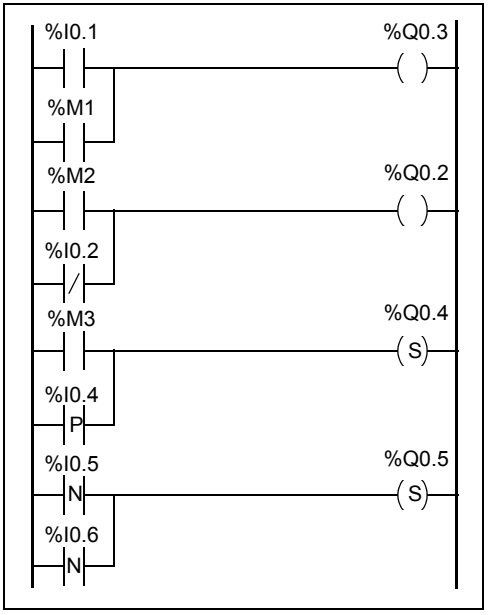
Инструкции логического ИЛИ (OR, ORN, ORR, ORF)

Введение

Инструкции OR выполняют операцию логического ИЛИ между операндом (или его инверсией, или его передним или задним фронтом) и булевым результатом предыдущей инструкции.

Примеры

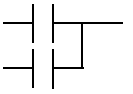
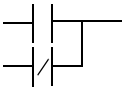
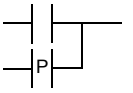
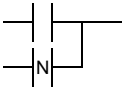
На следующих рисунках приведены примеры инструкции логического ИЛИ (OR).




```
LD      %I0.1
OR      %M1
ST      %Q0.3

LD      %M2
ORN     %I0.2
ST      %Q0.2

LD      %M3
ORR     %I0.4
S       %Q0.4

LDF     %I0.5
ORF     %I0.6
S       %Q0.5
```

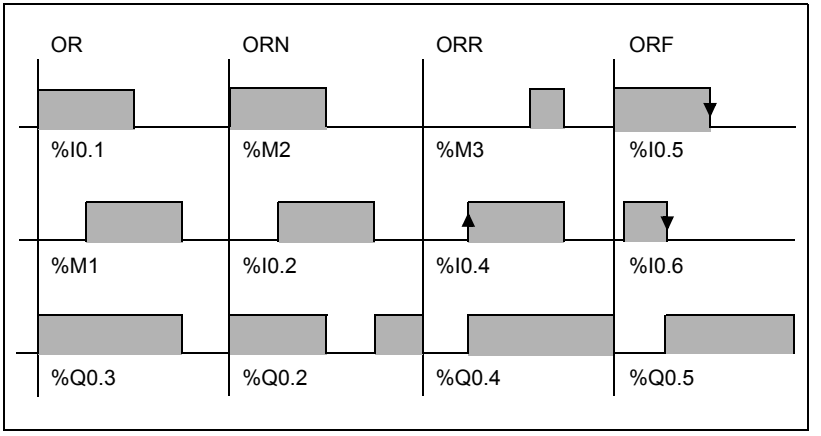
Разрешенные операнды

Следующая таблица содержит типы инструкций логического ИЛИ, эквиваленты в лестничной логике и разрешенные операнды.

Инструкция списка	Лестничный эквивалент	Разрешенные операнды
OR		0/1, %I,%IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk
ORN		0/1, %I,%IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk
ORR		%I, %IA, %M
ORF		%I, %IA, %M

Временная диаграмма

Ниже приведена временная диаграмма для инструкции OR.



Инструкции исключающего ИЛИ (XOR, XORN, XORR, XORF)

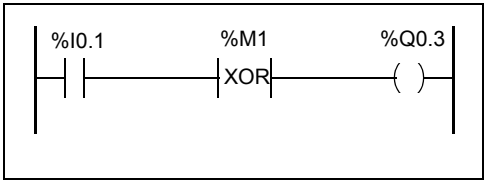
Введение

Инструкции XOR выполняют операцию исключающего ИЛИ между операндом (или его инверсией, или его передним или задним фронтом) и булевым результатом предыдущей инструкции.

Примеры

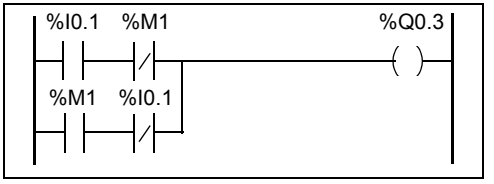
Следующий пример показывает использование инструкций XOR.

Схема использования инструкции XOR




```
LD    %I0.1
XOR   %M1
ST    %Q0.3
```

Схема NOT, использующая XOR




```
LD    %I0.1
ANDN  %M1
OR(   %M1
ANDN  %I0.1
)
ST    %Q0.3
```

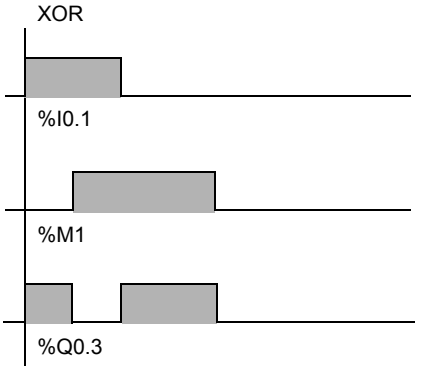
Разрешенные операнды

Следующая таблица содержит типы инструкций исключающего ИЛИ и разрешенные операнды.

Инструкция списка	Разрешенные операнды
XOR	%I, %IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk
XORN	%I, %IA, %Q, %QA, %M, %S, %X, %BLK.x, %•:Xk
XORR	%I, %IA, %M
XORF	%I, %IA, %M

Временная диаграмма

Ниже приведена временная диаграмма для инструкции XOR.

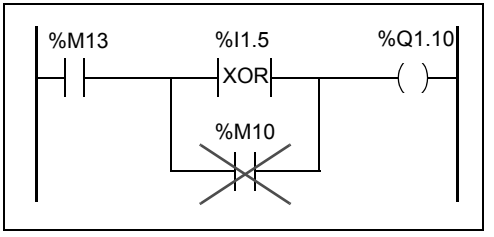


Особые случаи

Ниже перечислены особые меры предосторожности при использовании инструкций исключающего ИЛИ в программах лестничной логики:

- Не вставляйте контакты исключающего ИЛИ в первую позицию ступени.
- Не вставляйте контакты исключающего ИЛИ параллельно с другими элементами лестничной логики (см. пример ниже)

Как показано в последующем примере, подключение элемента параллельно с контактом исключающего ИЛИ создаст ошибку достоверности.



Инструкция НЕ (N)

Введение

Инструкция NOT (N) инвертирует булевский результат предыдущей инструкции.

Пример

Ниже приведен пример использования инструкции NOT.

```
LD      %I0.1
OR      %M2
ST      %Q0.2
N
AND     %M3
ST      %Q0.3
```

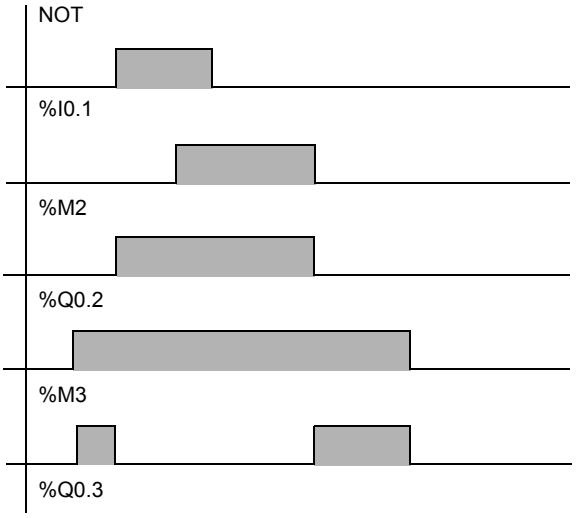
Примечание: Инструкция NOT не обратима.

Разрешенные операнды

Не применяются.

Временная диаграмма

Ниже приведена временная диаграмма для инструкции NOT.



14.2 Основные функциональные блоки

Обзор

Цель этого раздела

Этот раздел содержит описания и принципы программирования для использования основных функциональных блоков.

Содержание раздела

Этот раздел содержит следующие темы:

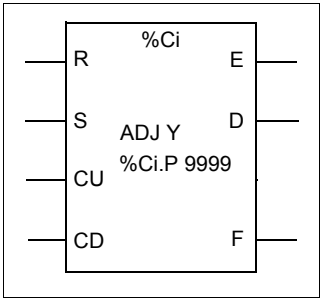
Основные функциональные блоки

Введение

Функциональные блоки - это источники для битовых объектов и специальных слов, которые используются программами. Базовые функциональные блоки реализуют простые функции, такие как таймеры или реверсивный счетчик up/down.

Пример функционального блока

На следующем рисунке приведен пример функционального блока реверсивного счетчика.



Up/down counter block

Битовые объекты

Битовые объекты соответствуют выходам блока. Эти биты могут быть доступны булевым тестовым инструкциям за счет использования одного из следующих методов:

- Непосредственно (например, LD E), если они подключены к блоку в обратимом программировании (см. *Принципы программирования стандартных функциональных блоков, стр. 276*)
- Указывая тип блока (например, LD %Ci.E).

Входы могут быть доступны в виде инструкций.

Объекты слова

Объекты-слова соответствуют определенным параметрам и значениям, таким как:

- **Параметры конфигурации блока:** некоторые параметры доступны программе (например, предварительного выбранные параметры), а некоторые параметры недоступны программе (например, ось времени).
- **Текущие значения:** Например, %Ci.V, текущее значение счета.

Доступные битовые объекты и объекты-слова

В следующей таблице описаны основные функциональные блоки, которые могут быть доступны программе.

Основной функциональный блок	Символ	Диапазон (i)	Типы объектов	Описание	Адрес	Доступ к записи
Таймер	%TMi	0 - 127	Слово	Текущее значение	%TMi.V	нет
				Предустановленное значение	%TMi.P	да
			Бит	Выход таймера	%TMi.Q	нет
Реверсивный счетчик	%Ci	0 - 127	Слово	Текущее значение	%Ci.V	нет
				Предустановленное значение	%Ci.P	да
			Бит	Выход потери значимости (empty)	%Ci.E	нет
				Выход достижения предустановки	%Ci.D	нет
				Выход переполнения (full)	%Ci.F	нет

Принципы программирования стандартных функциональных блоков

Введение

Используйте один из следующих методов для программирования стандартных функциональных блоков:

- Инструкции функциональных блоков (например, BLK %TM2): Этот обратимый метод программирования на языке лестничной логикипозволяет выполнять операции над блоками в одном месте программы.
- Особые инструкции (например, CU %Ci): Этот необратимый метод позволяет выполнять операции над входами блоков в нескольких местах программы (например, строка 100 CU %C1, строка 174 CD %C1, строка 209 LD %C1.D).

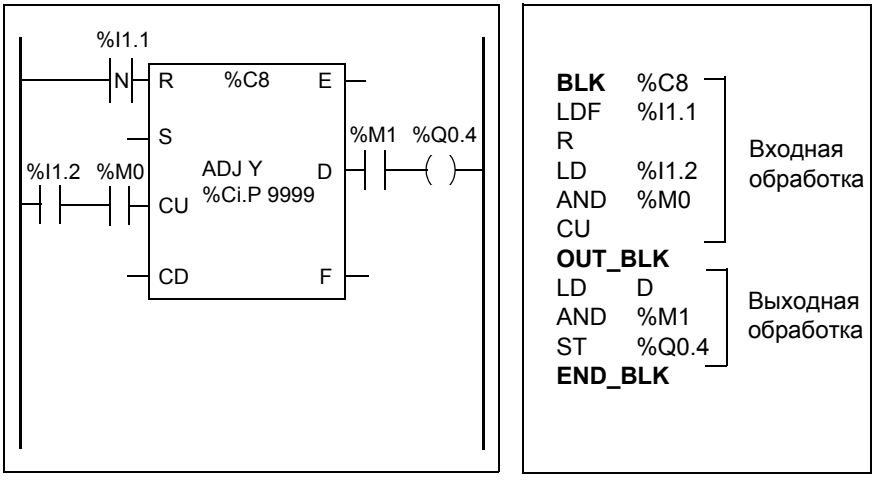
Обратимое программирование

Используйте инструкции BLK, OUT_BLK и END_BLK для обратимого программирования:

- **BLK:** Указывает начало блока.
- **OUT_BLK:** Используется для прямого подключения выходов блока
- **END_BLK:** Указывает конец блока.

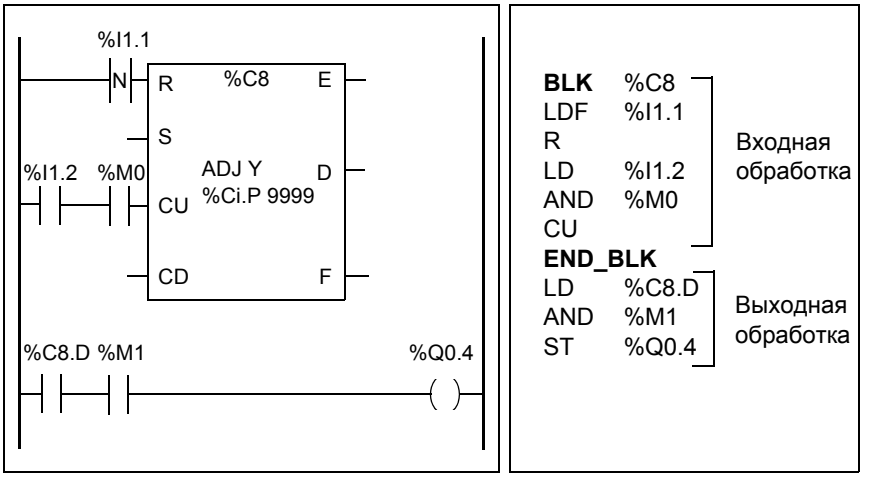
Пример с подключением выхода

Следующий пример показывает обратимое программирование функционального блока "счетчик" с подключенными выходами.



Пример без подключения выходов

Этот пример показывает обратимое программирование функционального блока счетчика без подключения выходов.



Примечание: Только тестовые и входные инструкции над соответствующим блоком могут быть расположены между инструкциями BLK и OUT_BLK (или между BLK и END_BLK, если OUT_BLK не запрограммирован).

Функциональный блок таймера (%TMi)

Введение

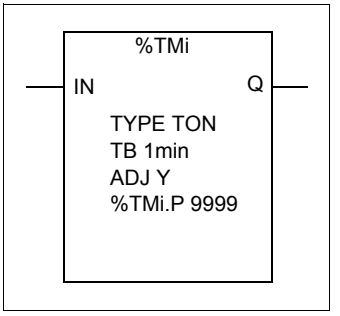
Существуют 3 типа функциональных блоков таймера:

- TON (Timer On-Delay - таймер задержки включения): этот тип таймера используется для управления действиями по задержке включения.
- TOF (Timer Off-Delay - таймер задержки выключения): этот тип таймера используется для управления действиями по задержке выключения.
- TP (Timer - Pulse - таймер импульса): этот тип таймера используется для генерации импульса заданной длительности

Задержки или периоды импульсов могут программироваться и могут быть модифицированы при помощи программного обеспечения Twido.

Иллюстрация

Ниже приведена иллюстрация функционального блока таймера.



Функциональный блок таймера

Параметры

Функциональный блок таймера имеет следующие параметры:

Параметр	Метка	Значение
Номер таймера	%TMi	От 0 до 63: TWDLCAA10DRF и TWDLCAA16DRF. От 0 до 127 для всех остальных контроллеров.
Тип	TON	• Таймер задержки включения (по умолчанию)
	TOF	• Таймер задержки выключения
	TP	• Импульс (одновибратор)
Ось времени	TB	1 мин. (по умолчанию), 1 с., 100 мс.,10мс., 1мс.
Текущее значение	%TMi.V	Слово, которое инкрементируется от 0 до %TMi.P, когда таймер работает. Может быть прочитано и проверено, но не записано программой. %TMi.V может быть изменено при помощи редактора анимационных таблиц.
Предустановленное значение	%TMi.P	0-9999. Слово, которое может быть прочитано, проверено и записано программой. Значение по умолчанию: 9999. Генерируемый период или задержка составляет %TMi.P x TB
Редактор анимационных таблиц	Y/N	Y: Yes (да), предустановленное значение %TMi.P может быть изменено при помощи редактора таблиц анимации. N: No (нет), предустановленное значение %TMi.P не может быть изменено.
Вход разрешения (или инструкции)	IN	Запускает таймер по переднему фронту (типы TON или TP) или заднему фронту (тип TOF).
Выход таймера	Q	Соответствующий бит %TMi.Q устанавливается в 1, в зависимости от того, какая функция выполняется: TON, TOF или TP.

Примечание: Чем больше предустановленное значение, тем выше точность таймера.

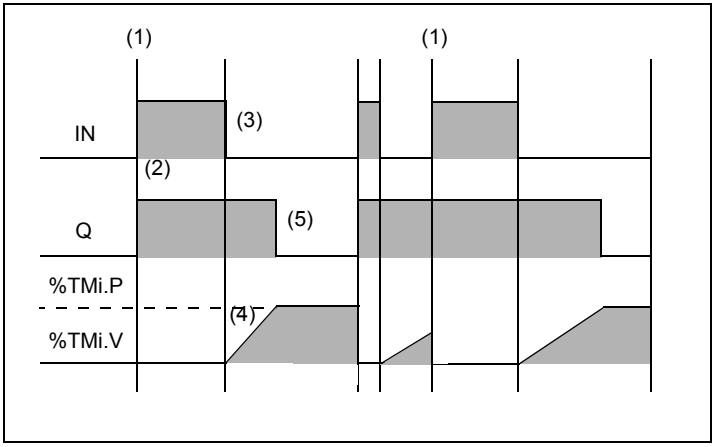
Тип таймера TOF

Введение

Тип таймера TOF (timer off-delay - таймер задержки выключения) используется для управления действиями по задержке выключения. Эта задержка программируется с помощью TwidoSoft.

Временная диаграмма

Следующая временная диаграмма демонстрирует функционирование таймера типа TOF.



Функционирование

Следующая таблица описывает функционирование таймера типа TOF.

Фаза	Описание
1	Текущее значение %TMi.V устанавливается в 0 по переднему фронту на входе IN, даже если таймер уже запущен.
2	Выходной бит %TMi.Q устанавливается в 1 по переднему фронту на входе IN.
3	Таймер запускается по заднему фронту на входе IN.
4	Текущее значение %TMi.V увеличивается от 0 до TMi.P на одну единицу при каждом импульсе временной базы TB.
5	Выходной бит %TMi.Q сбрасывается в ноль при достижении текущим значением %TMi.P.

Тип таймера TON

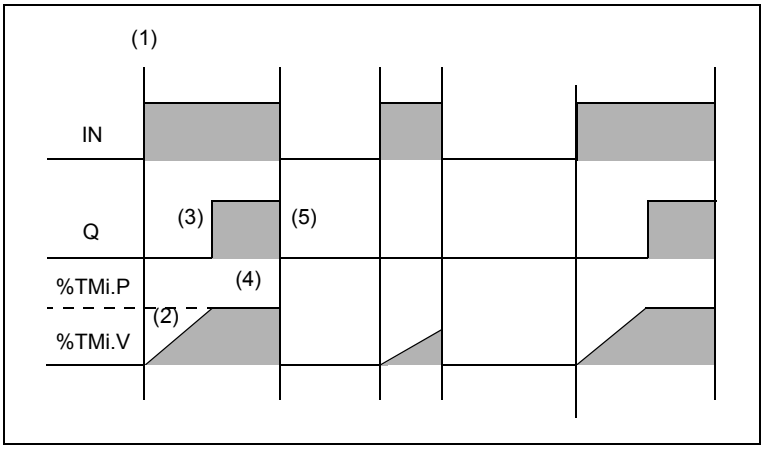
Введение

Тип таймера TON (timer on-delay - таймер задержки выключения) используется для управления действиями по задержке включения. Эта задержка программируется с помощью TwidoSoft.

.

Временная диаграмма

Следующая временная диаграмма демонстрирует функционирование таймера типа TON.



Функционирование

Следующая таблица описывает функционирование таймера типа TON.

Фаза	Описание
1	Запуск таймера по переднему фронту входа IN.
2	Текущее значение %TMi.V увеличивается от 0 до TMi.P на одну единицу при каждом импульсе временной базы TB.
3	Выходной бит %TMi.Q устанавливается в 1 при достижении текущим значением %TMi.P.
4	Выходной бит %TMi.Q остается в 1 пока на входе IN 1.
5	Когда на входе IN появляется задний фронт, таймер останавливается, даже если таймер не достиг %TMi.P, и %TMi.V установлено в 0.

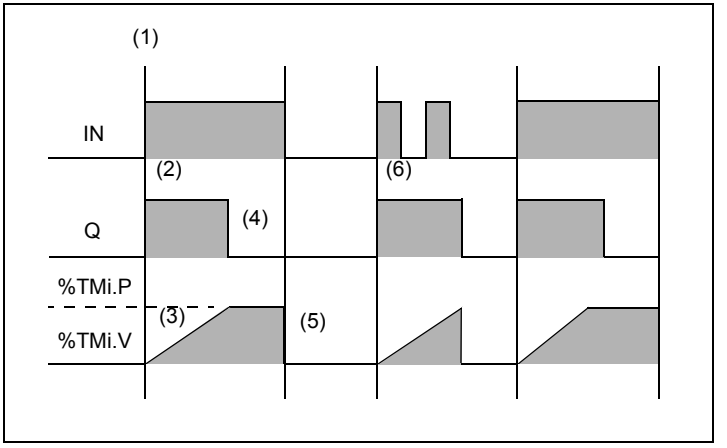
Тип таймера TP

Введение

Тип таймера TP (timer pulse - таймер импульса) используется для генерации импульсов заданной длительности. Эта длительность программируется с помощью TwidoSoft.

Временная диаграмма

Следующая временная диаграмма демонстрирует функционирование таймера типа TP.



Функционирование

Следующая таблица описывает функционирование таймера типа TP.

Фаза	Описание
1	Запуск таймера по переднему фронту входа IN. Текущее значение %TMi.V устанавливается в 0, если таймер еще не запущен.
2	Выходной бит %TMi.Q устанавливается в единицу при запуске таймера.
3	Текущее значение %TMi.V увеличивается от 0 до TMi.P на одну единицу при каждом импульсе временной базы TB.
4	Выходной бит %TMi.Q сбрасывается в 0 при достижении текущим значением %TMi.P.
5	Текущее значение %TMi.V устанавливается в 0, когда %TMi.V равно %TMi.P и вход IN возвращается в 0.
6	Этот таймер не может быть переустановлен. Если %TMi.V равно %TMi.P, и на входе IN ноль, то %TMi.V устанавливается в 0.

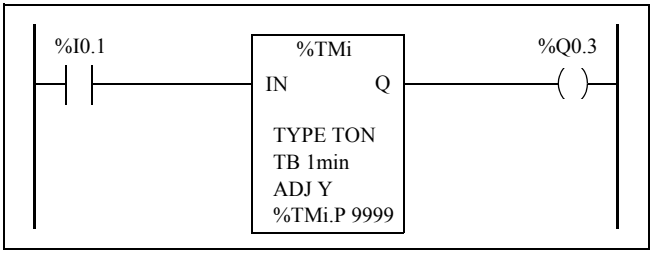
Программирование и конфигурирование таймеров

Введение

Функциональные блоки таймеров программируются одинаково, вне зависимости от того, как они будут использоваться. Функция таймера (TON, TOF или TP) выбирается во время конфигурации.

Примеры

Ниже изображен функциональный блок таймера с примерами обратимого и необратимого программирования.



Обратимое программирование

```
BLK     %TM1
LD      %I0.1
IN
OUT_BLK
LD      Q
ST      %Q0.3
END_BLK
```

Необратимое программирование

```
LD      %I0.1
IN      %TM1
LD      %TM1.Q
ST      %Q0.3
```

Конфигурация

Следующие параметры должны быть введены во время конфигурации:

- Тип таймера: TON, TOF или TP
- Масштаб: 1 мин., 1 сек., 100 мсек., 10 мсек., 1 мсек
- Предустановленное значение (%TMi.P): от 0 до 9999
- Настройка: Выполнена или Не выполнена

Особые случаи

Следующая таблица содержит список особых случаев для программирования функционального блока Сдвигающего регистра битов

Особый случай	**Описание**
Результат холодного перезапуска (%S0=1)	Сбрасывает текущее значение в 0. Устанавливает выход TMi.Q в 0. Предустановленному значению присваивается значение, определенное во время конфигурации.
Результат теплого перезапуска (%S1=1)	Не влияет на текущее и предустановленное значения таймера. Текущее значение не меняется во время отключения питания.
Результат остановки контроллера	Остановка контроллера не замораживает текущее значение.
Результат программного перехода	При переходе через блок таймера таймер не останавливается. Таймер будет продолжать инкрементироваться, пока не достигнет предустановленного значения (%TMi.P). В этот момент бит выполнения (%TMi.Q), приписанный выходу Q блока таймера, меняет свое состояние. Тем не менее, соответствующий выход, подключенный непосредственно к выходу блока, не активируется и не сканируется контроллером.
Проверка при помощи бита %TMi.Q (бит выполнения)	Рекомендуется проверять бит %TMi.Q только в одном месте программы.
Результат инструкций главного управляющего реле MCS/MCR	Блок таймера, который запрограммирован между двумя инструкциями MCS/MCR, переустанавливается при активизации инструкции MCS.
Результат модифицирования предустановленного TMi.P	Модификация текущего значения использованием инструкции или установкой значения происходит только в момент следующей активации таймера.

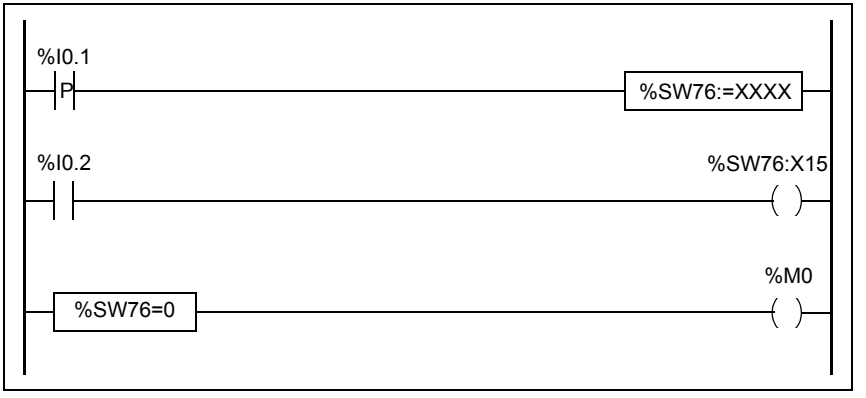
Таймеры с масштабом 1 мс

Масштаб 1 мс доступен только с первыми 5 таймерами. Четыре системных слова %SW76, %SW77, %SW78 и %SW79 могут быть использованы как "песочные часы". Эти 4 слова декрементируются системой индивидуально каждую миллисекунду, **если они имеют положительное значение**. Многократная синхронизация может быть достигнута успешной загрузкой этих слов или тестированием промежуточных значений. Если значение одного из этих четырех слов меньше 0, оно не будет модифицировано. Таймер может быть "заморожен" установкой соответствующего бита 15 в 1, и затем "разморожен" переустановкой его в 0.

Пример программирования

Ниже приведен пример программирования функционального блока таймера.

```
LDR     %I0.1          (Запуск таймера по переднему фронту %I0.1)
[%SW76:=XXXX]          (XXXX = требуемое значение)
LD      %I0.2          (необязательное управление “заморозкой”, входt
                       I0.2 замораживается)
ST      %SW76:X15
LD      [%SW76=0]      (таймер заканчивает тестирование)
ST      %M0
.............
```



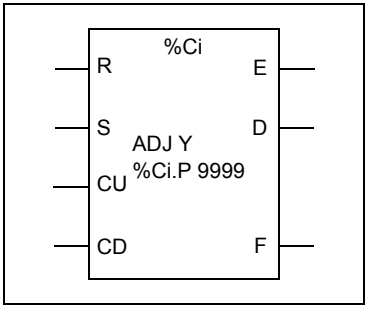
Функциональный блок счетчика Up/Down (%Ci)

Введение

Функциональный блок счетчика (%Ci) обеспечивает прямой и обратный счет событий. Эти две операции могут быть выполнены одновременно.

Иллюстрация

Ниже приводится изображение функционального блока реверсивного счетчика up/down.



Функциональный блок счетчика up/down

Параметры

Функциональный блок счетчика имеет следующие параметры:

Параметр	Метка	Значение
Номер счетчика	%Ci	От 0 до 127
Текущее значение	%Ci.V	Слово инкрементируется или декрементируется в соответствии с входами (или инструкциями) CU и CD. Может быть прочитано и протестировано, но не может быть записано программой. Для модификации %Ci.V используется редактор данных.
Предустановленное значение	%Ci.P	0<= %Ci.P <= 9999. Слово может быть прочитано, протестировано и записано (значение по умолчанию: 9999).
Редактирование с помощью редактора анимационных таблиц	ADJ	• Y: Да, предустановленное значение может быть изменено при помощи редактора анимационных таблиц. N: Нет, предустановленное значение не может быть изменено при помощи редактора анимационных таблиц.
Сброс входа (или инструкции)	R	В состоянии 1: %Ci.V=0.
Установка входа (или инструкции)	S	В состоянии 1: %Ci.V=%Ci.P.
Вход (или инструкция) прямого счета	CU	Инкрементирует %Ci.V по переднему фронту.
Вход (или инструкция) обратного счета	CD	Декрементирует %Ci.V по переднему фронту.
Выход переполнения обратного счета	E (Empty)	Соответствующий бит %Ci.E=1, когда обратный счетчик %Ci.V меняется от 0 до 9999 (устанавливает в 1, когда %Ci.V , и сбрасывает в 0, если счетчик продолжает обратный счет).
Достигнут предустановленный выход	D (Done)	Соответствующий бит %Ci.D=1, когда %Ci.V=%Ci.P.
Выход переполнения прямого счета	F (Full)	Соответствующий бит %Ci.F=1, когда %Ci.V меняется от 9999 до 0 (устанавливается в 1, когда %Ci.V достигает 0, и сбрасывается в 0, если счетчик продолжает считать)

Функционирование

Следующая таблица описывает основные этапы работы счетчика up/down.

Операция	Действие	Результат
Счет	На входе прямого счета CU появляется фронт (или активируется инструкция CU).	Текущее значение %Ci.V увеличивается на один.
	Текущее значение %Ci.V равно предустановочному значению %Ci.P.	Вых. бит "предустановка достигнута" %Ci.D переключается в 1.
	Текущее значение %Ci.V изменяется из 9999 в 0.	Вых. бит %Ci.F (переполнение прямого счета) переключается в 1.
	Если счетчик продолжает считать вверх.	Вых. бит%Ci.F (переполнение прямого счета) сбрасывается в 0.
Обратный счет	На входе обратного счета CD появляется фронт (или активируется инструкция CD).	Текущее значение %Ci.V уменьшается на один.
	Текущее значение %Ci.V изменяется из 0 в 9999.	Вых. бит %Ci.F (переполнение обратного счета) переключается в 1.
	Если счетчик продолжает считать вниз.	Вых. бит%Ci.F (переполнение обратного счета) сбрасывается в 0.
Счет Up/down	Чтобы одновременно использовать функции счета вверх и вниз (или активировать обе инструкции CD и CU), два соответствующих выхода CU и CD должны контролироваться одновременно. Эти два входа сканируются. Если они оба в 1, текущее значение не изменяется.	
Сброс	Вход R устанавливается в1 (или активируется инструкция R).	Текущее значение %Ci.V устанавливается 0. Выходы %Ci.E, %Ci.D и %Ci.F в 0. Результирующий вход имеет приоритет.
Предустановка	Если вход S устанавливается в 1 (или активируется инструкция S) и вход сброса в 0 (или инструкция R не активна).	Текущее значение %Ci.V принимает значение %Ci.P и выход %Ci.D устанавливается в 1.

Особые случаи

В следующей таблице представлен список особых случаев функционирования/конфигурации для счетчиков.

Особый случай	Описание
Влияние холодного перезапуска (%S0=1)	● Текущее значение %Ci.V устанавливается в 0. ● Выходные биты %Ci.E, %Ci.D и %Ci.F устанавливаются в 0. ● Предустановочное значение инициализируется значением, определенным при конфигурации.
Влияние теплого перезапуска (%S1=1) после остановки контроллера	Не влияет на текущее значение счетчика (%Ci.V).
Влияние изменения предустановки %Ci.P	Изменение предустановленного значения через инструкцию или при подстраивании имеет влияние, когда блок обрабатывается приложением (активизация одного из входов).

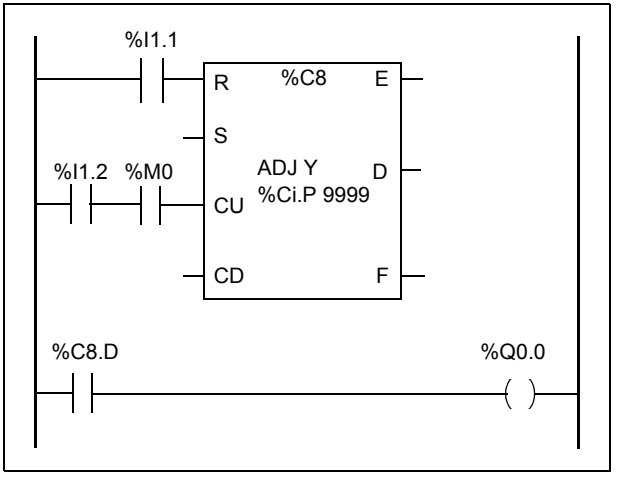
Программирование и конфигурирование счетчиков

Введение

В следующем примере приведен счетчик, который обеспечивает счет до 5000. Каждый импульс на входе %I1.2 (когда внутренний бит %M0 установлен в 1) увеличивает счетчик %C8 до конечного предустановленного значения (бит %C8.D=1). Счетчик сбрасывается входом %I1.1.

Пример программирования

На следующем рисунке показан функциональный блок счетчика с примерами обратимого и необратимого программирования.



Лестничная диаграмма

```
BLK    %C8
LD     %I1.1
R
LD     %I1.2
AND    %M0
CU
END_BLK
LD     %C8.D
ST     %Q0.0
```

Обратимое программирование

```
LD     %I1.1
R      %C8
LD     %I1.2
AND    %M0
CU     %C8
LD     %C8.D
ST     %Q0.0
```

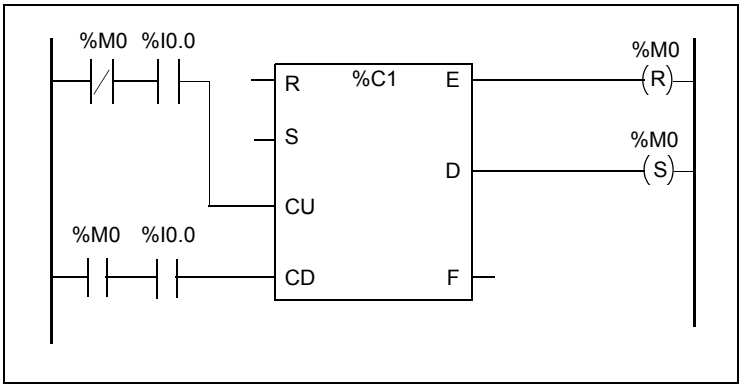
Необратимое программирование

Конфигурирование

Следующие параметры должны быть введены во время конфигурирования:
- Предустановленное значение (%Ci.P): установлено в 5000 в этом примере
- Регулирование: Да

Пример счетчика Up/Down

На следующем рисунке приведен пример функционального блока счетчика Up/Down.



Лестничная диаграмма

В этом примере, если мы возьмем %C1.P 4, текущее значение счетчика %C1.V будет увеличиваться от 0 до 3, затем уменьшаться от 3 до 0. Тогда как %I0.0=1 %C1.V колеблется между 0 и 3.

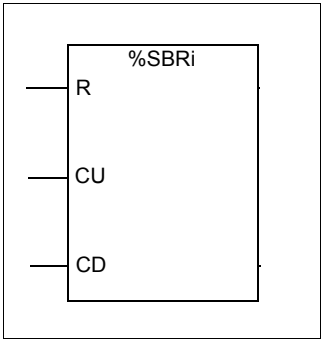
Функциональный блок сдвигающего регистра битов (%SBRi)

Введение

Функциональный блок сдвигающего регистра битов (%SBRi) обеспечивает сдвиг влево или вправо двоичных битов данных (0 или 1).

Иллюстрация

Ниже приведен пример функционального блока сдвигающего регистра битов.



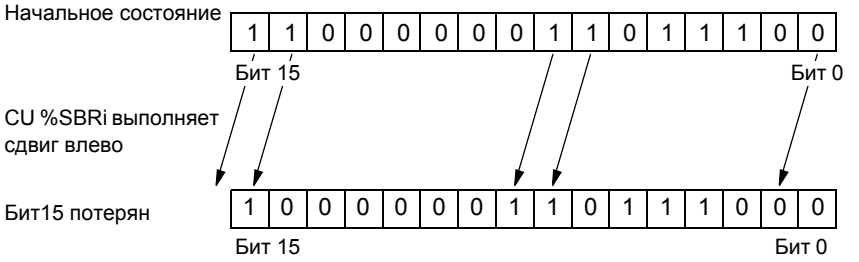
Параметры

Функциональный блок сдвигающего регистра битов имеет следующие параметры.

Параметр	Метка	Значение
Номер регистра	%SBRi	0 до 7
Бит регистра	%SBRi.j	Биты от 0 до 15 (j = 0 до 15) сдвигающего регистра могут проверяться инструкцией тестирования и записываться при помощи инструкции присваивания.
Вход сброса (или инструкция)	R	По фронту устанавливает биты регистра с 0 по 15 %SBRi.j в 0.
Вход сдвига влево (или инструкция)	CU	По фронту сдвигает биты регистра влево.
Вход сдвига вправо (или инструкция)	CD	По фронту сдвигает биты регистра вправо.

Функционирование

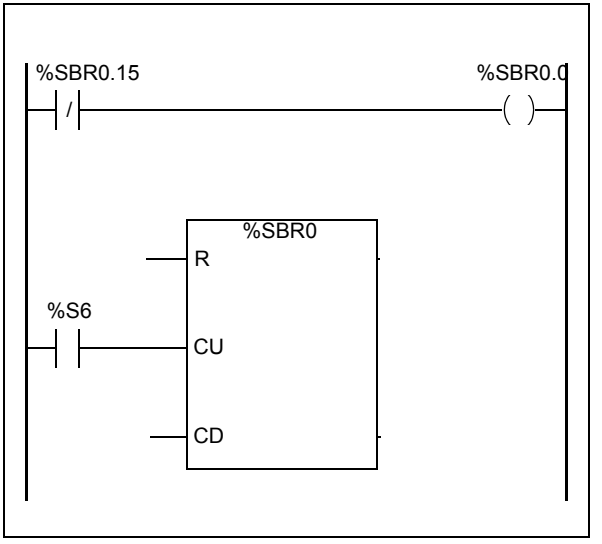
На следующем рисунке показан битовый образец до и после операции сдвига.



Это также справедливо по отношению к сдвигу вправо (Бит 15 до Бит 0), использующему инструкцию CD. Бит 0 теряется.
Если 16-битного регистра недостаточно, возможно использовать программу для каскадирования нескольких регистров.

Программирование

В следующем примере бит сдвигается влево каждую секунду пока Бит 0 не присвоит противоположное значение для Бит 15.



Обратимое программирование

```
LDN    %SBR0.15
ST     %SBR0.0
BLK    %SBR0
LD     %S6
CU
END_BLK
```

Необратимое программирование

```
LDN    %SBR0.15
ST     %SBR0.0
LD     %S6
CU     %SBR0
```

Особые случаи

Следующая таблица содержит список особых случаев для программирования функционального блока сдвигающего регистра битов.

Особый случай	Описание
Влияние холодного перезапуска (%S0=1)	Устанавливает все биты слова регистра в 0.
Влияние теплого перезапуска (%S1=1)	Не влияет на биты слова регистра.

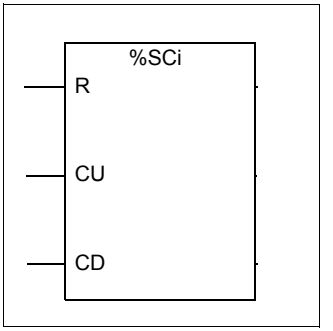
Функциональный блок счетчика шагов (%SCi)

Введение

Функциональный блок счетчика шагов (%SCi) обеспечивает набор шагов, которым могут быть приписаны действия. Перемещение от одного шага к другому зависит от внешних или внутренних событий. Каждый раз, когда шаг активен, связанный бит устанавливается в 1. Одновременно только один шаг счетчика шагов может быть активен.

Иллюстрация

Ниже приведен пример функционального блока счетчика шагов.



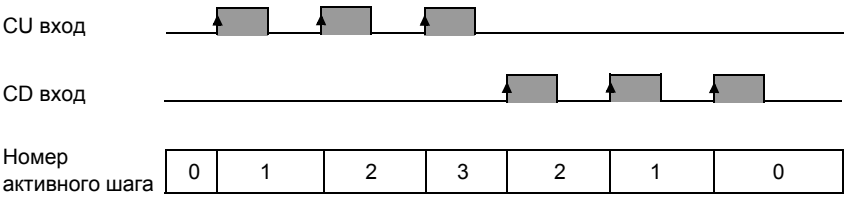
Параметры

Блок счетчика шагов имеет следующие параметры:

Параметр	Метка	Значение
Номер счетчика шагов	%SCi	0 - 7
Бит счетчика шагов	%SCi.j	Биты счетчика шагов с 0 по 255 (j = 0 - 255) могут проверяться логической операцией загрузки и записываться инструкцией присваивания.
Вход сброса (или инструкция)	R	По фронту сбрасывает счетчик шагов.
Вход увеличения (или инструкция)	CU	По фронту увеличивает счетчик шагов на один шаг.
Вход уменьшения (или инструкция)	CD	По фронту уменьшает счетчик шагов на один шаг.

Временная диаграмма

Следующая временная диаграмма иллюстрирует работу функционального блока шагов.

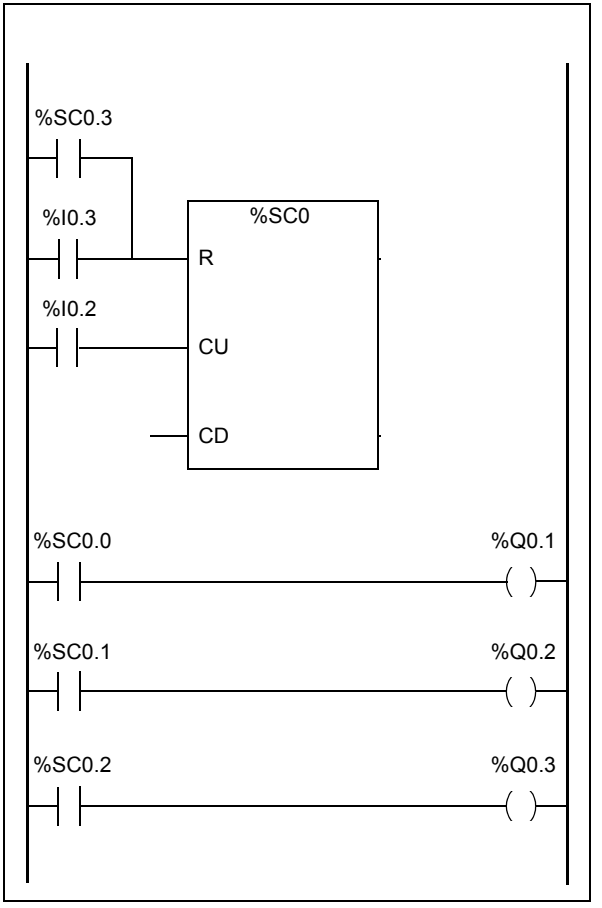


Программирование

Ниже приведен пример функционального блока счетчика шагов .

- Счетчик шагов 0 увеличивается входом %I0.2.
- Счетчик шагов 0 сбрасывается в 0 входом %I0.3 или при достижении шага 3.
- Шаг 0 управляет выходом %Q0.1, шаг 1 управляет выходом %Q0.2, и шаг 2 управляет выходом %Q0.3.

На следующем рисунке показано обратимое и необратимое программирование для этого примера.



Обратимое программирование

```
BLK     %SC0
LD      %SC0.3
OR      %I0.3
R
LD      %I0.2
CU
END_BLK
LD      %SC0.0
ST      %Q0.1
LD      %SC0.1
ST      %Q0.2
LD      %SC0.2
ST      %Q0.3
```

Необратимое программирование

```
LD      %SC0.3
OR      %I0.3
R       %SC0
LD      %I0.2
CU      %SC0
LD      %SC0.0
ST      %Q0.1
LD      %SC0.1
ST      %Q0.2
LD      %SC0.2
ST      %Q0.3
```

Особые случаи

Следующая таблица содержит список особых случаев для программирования функционального блока счетчика шагов.

Особый случай	Описание
Влияние хол. перезапуска (%S0=1)	Инициализирует счетчик шагов.
Влияние тепл. перезапуска (%S1=1)	Не влияет на счетчик шагов.

14.3 Цифровая обработка

Обзор

Цель этого раздела

Этот раздел представляет введение в цифровую обработку, включая описания и советы по программированию.

Содержание раздела

Этот раздел содержит следующие темы:

Введение в цифровые инструкции

Обзор

Цифровые инструкции обычно применяются к 16-битным словам (см. *Объекты-слова, стр. 28*) и к 32-битным двойным словам (См. *Объекты с плавающей точкой и двойные слова, стр. 31*). Они записываются между квадратными скобками. Если результат предшествующей логической операции был истинным (Булевский аккумулятор = 1), цифровая инструкция выполняется. Если результат предшествующей логической операции был ложным (Булевский аккумулятор = 0), цифровая инструкция не выполняется и операнд остается неизменным.

Инструкции присваивания

Введение

Инструкции присваивания используются для загрузки операнда Op2 в операнд Op1.

Присваивание

Синтаксис для инструкций присваивания.

[Op1:=Op2]	<=>	Op2 -> Op1

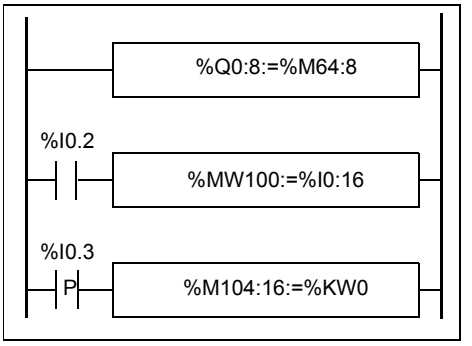
Инструкции присваивания могут выполняться над:
- Битовыми строками
- Словами
- Двойными словами
- Словами с ПТ
- Таблицами слов
- Таблицами двойных слов
- Таблицами слов с ПТ

Присваивание битовых строк

Операции могут выполняться над следующими битовыми строками (см. *Структурированные объекты, стр. 43*):
- Битовая строка -> битовая строка (Пример 1)
- Битовая строка -> слово (Пример 2) или двойное слово (индексированное)
- Слово или двойное слово (индексированное) -> битовая строка(Пример 3)
- Прямое значение -> битовая строка

Примеры

Примеры присваивания битовых строк.




```
LD      1
[%Q0:8:=%M64:8]          (Пр. 1)

LD      %I0.2
[%MW100:=%I0:16]         (Пр. 2)

LDR     %I0.3
[%M104:16:=%KW0]         (Пр. 3)
```

Правила использования:

- Для присваивания битовая строка -> слово: Биты в строке передаются в слово начиная справа (первый бит строки в бит 0 в слове), биты слова, которые не были вовлечены в передачу (длина J16) устанавливаются в 0.
- Для присваивания слово -> битовая строка: Биты слова передаются справа (бит слова 0 в первый бит строки).

Присваивание битовых строк

Синтаксис для присваивания битовых строк.

Оператор	Синтаксис	Операнд 1 (Op1)	Операнд 2 (Op2)
:=	[Op1: = Op2] **Операнд** (Op1) принимает значение операнда 2 (Op2)	%MWi,%QWi, %QWAi,%SWi %MWi[%MWi], %MDi, %MDi[%MWi] **%Mi:L, %Qi:L, %Si:L, %Xi:L**	Прямое значение, %MWi, %KWi, %IW,%IWAi, %INWi, %QWi, %QWAi %QNWi, %SWi, %BLK.x, %MWi[%MWi], %KWi[%MWi], %MDi[%MWi], %KDi[%MWi], **%Mi:L,%Qi:L, %Si:L, %Xi:L, %Ii:L**

Примечание: Аббревиатура %BLK.x (например, %C0.P) используется для описания любого слова функционального блока.

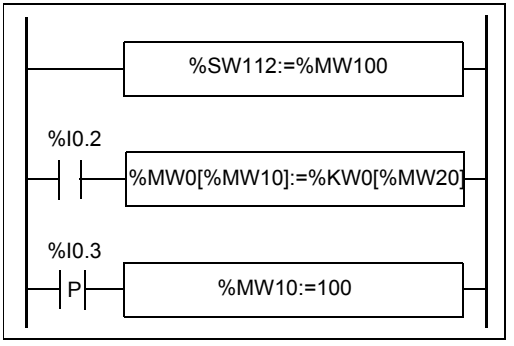
Присваивание слов

Операции присваивания могут выполняться над следующими словами и двойными словами:

- Слово (индексированное) -> слово (2, например) (индексированное или нет)
- Двойное слово (индексированное) -> двойное слово (индексированное или нет)
- Прямое целое значение -> слово (Пример 3) или двойное слово (индексированное или нет)
- Битовая строка -> слово или двойное слово
- Слово с ПТ (индексированное или нет-> слово с ПТ (индексированное или нет)
- Слово или двойное слово -> битовая строка
- Прямое значение с ПТ -> слово с ПТ (индексированное или нет)

Примеры

Примеры присваивания слов.




```
LD      1
[%SW112:=%MW100]    (Пр. 1)

LD      %I0.2
[%MW0[%MW10]:=      (Пр. 2)
%KW0[%MW20]]

LDR     %I0.3       (Пр. 3)
[%MW10:=100]
```

Синтаксис

Синтаксис для присваивания слов.

Оператор	Синтаксис
:=	[Op1: = Op2] Операнд 1 (Op1) принимает значение операнда 2 (Op2)

В следующей таблице представлены подробные операнды:

Тип	Операнд 1 (Op1)	операнд 2 (Op2)
слово, двойное слово, битовая строка	**%BLK.x, %MWi, %QWi, %QWAi, %SWi %MWi[MWi**, %MDi, %MDi[%MWj]], %Mi:L, %Qi:L, %Si:L, %Xi:L	**Прямое значение, %MWi, %KWi, %IW, %IWAi, %QWi, %QWAi, %SWi, %MWi[MWi], %KWi[MWi], %MDi, %MDi[%MWj], %KDi, %KDi[MWj]** , %INW, %Mi:L, %Qi:L, %QNW, %Si:L, %Xi:L, %Ii:L
ПТ	**%MFi, %MFi[%MWj]**	**Прямое значение с ПТ, %MFi, %MFi[%MWj], %KFi, %KFi[%MWj]**

Примечание: Аббревиатура %BLK.x (например, R3.I) используется для описания любого слова функционального блока. Для битовых строк %Mi:L, %Si:L, and %Xi:L, базовый адрес первой битовой строки должен быть кратен 8 (0, 8, 16, ..., 96, ...).

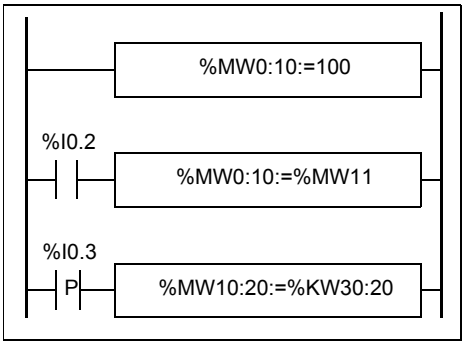
Присваивание Таблиц Слов, Двойных слов и Слов с ПТ

Операции присваивания могут выполняться над следующими таблицами объектов (См. *Таблицы слов, стр. 44*):

- Прямое целое значение -> таблица слов (Пример 1) или таблица двойных слов
- Слово -> таблица слов (Пример 2)
- Таблица слов -> таблица слов (Пример 3)
  Длина таблицы (L) должна быть одинаковая у обеих таблиц.
- Двойное слово -> таблица двойных слов
- Таблица двойных слов -> таблица двойных слов
  Длина таблицы (L) должна быть одинаковая у обеих таблиц.
- Прямое значение с ПТ -> таблица слов с ПТ
- ПТ -> таблица слов с ПТ
- Таблица слов с ПТ-> таблица слов с ПТ
  Длина таблицы (L) должна быть одинаковая у обеих таблиц.

Примеры

Примеры присваивания таблиц слов:




```
LD      1
[%MW0:10:=100]                  (Ex. 1)

LD      %I0.2
[%MW0:10:=%MW11]                (Ex. 2)

LDR     %I0.3
[%MW10:20:=%KW30:20]            (Ex. 3)
```

Синтаксис

Синтаксис для присваивания таблиц слов, двойных слов и слов с ПТ

Оператор	**Синтаксис**
:=	[Op1: = Op2] Операнд 1 (Op1) принимает значение операнда 2 (Op2)

В следующей таблице представлены подробные операнды:

Тип	**Операнд 1 (Op1)**	**Операнд 2 (Op2)**
Таблицы слов	**%MWi:L, %SWi:L**	**%MWi:L, %SWi:L**, прямое целое значение, %MWi, %KWi, %IW, %QW, %IWA, %QWA, %SWi, %BLK.x
Таблицы двойных слов	**%MDi:L**	Прямое целое значение**, %MDi, %KDi,%MDi:L, %KDi:L**
Таблицы слов с ПТ	**%MFi:L]**	Прямое значение с ПТ**, %MFi, %KFi, %MFi:L, %KFi:L**

Примечание: Аббревиатура %BLK.x (например, R3.I) используется для описания любого слова функционального блока.

Инструкции сравнения

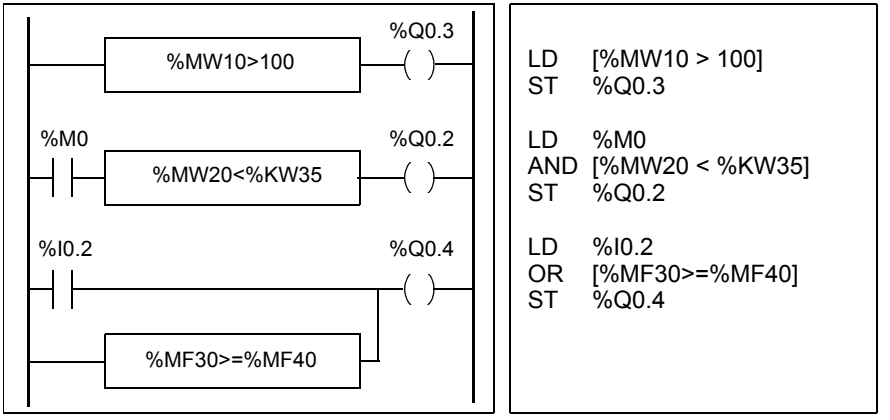
Введение

Инструкции сравнения используются для сравнения двух операндов. В следующей таблице перечислены типы инструкций сравнения.

Инструкция	Функция
>	Проверяет, больше ли операнд 1, чем операнд 2
>=	Проверяет, является ли операнд 1 больше или равен операнду 2
<	Проверяет, меньше ли операнд 1, чем операнд 2
<=	Проверяет, является ли операнд 1 меньше или равен операнду 2
=	Проверяет, равен ли операнд 1 операнду 2
<>	Проверяет, отличается ли операнд 1 от операнда 2

Структура

Сравнение выполняется внутри квадратных скобок, следующих за инструкциями LD, AND и OR. результат =1, когда сравнение истинно. Пример инструкций сравнения.



Синтаксис

Синтаксис инструкций сравнения:

Оператор	**Синтаксис**
>, >=, <, <=, =, <>	LD [Op1 Оператор Op2] AND [Op1 Оператор Op2] OR [Op1 Оператор Op2]

Операнды:

Тип	**Операнд 1 (Op1)**	**Операнд 2 (Op2)**
Слова	%MWi, %KWi, %INWi, %IW, %IWAi, %QNWi, %QWi, %QWAi, %QNWi, %SWi, %BLK.x	Прямое значение, %MWi, %KWi, %INWi, %IW, %IWAi, %QNWi, %QW, %QWAi, %SWi, %BLK.x, %MWi [%MWi], %KWi [%MWi]
Двойные слова	%MDi, %KDi	Прямое значение, %MDi, %KDi, %MDi [%MWi], %KD [%MWi]
Слова с ПТ	%MFi, %KFi	Прямое значение с ПТ, %MFi, %KFi, %MFi [%MWi], %KFi [%MWi]

Примечание: Инструкции сравнения могут использоваться внутри круглых скобок.

Пример использования инструкции сравнения внутри круглых скобок:

```
LD      %M0
AND(    [%MF20 > 10.0]
OR      %I0.0
)
ST      %Q0.1
```

Инструкции арифметических операций над целыми числами

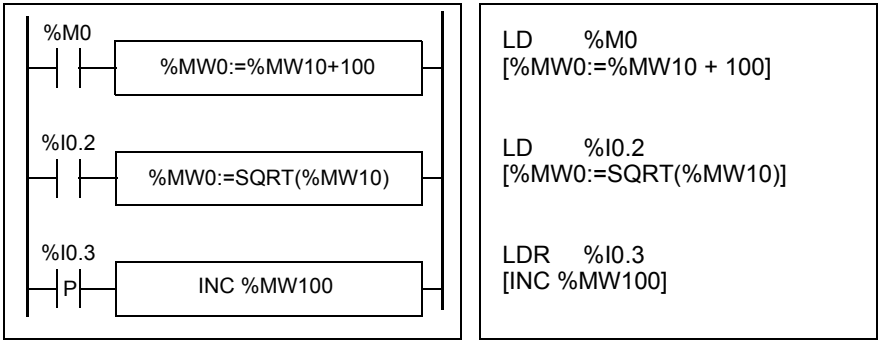
Введение

Арифметические инструкции используются для выполнения арифметических операций между двумя целыми операндами или над одним целым операндом. В следующей таблице перечислены типы арифметических инструкций.

Инструкция	Функция
+	Сложение двух операндов
-	Вычитание двух операндов
*	Умножение двух операндов
/	Деление двух операндов
REM	Остаток от деления двух операндов
SQRT	Квадратный корень из операнда
INC	Инкремент операнда
DEC	Декремент операнда
ABS	Модуль операнда

Структура

Арифметические операции выполняются следующим образом:



Синтаксис

Синтаксис зависит от используемых операторов, как показано в следующей таблице.

Оператор	**Синтаксис**
+,-,*,/,REM	[Op1: = Op 2 Оператор Op3]
INC, DEC	[Оператор Op1]
SQRT (1)	[Op1: = SQRT(Op2)]
ABS (1)	[Op1: = ABS(Op2)]

Операнды:

Тип	**Операнд 1 (Op1)**	**Операнды 2 и 3 (Op2 & 3) (1)**
Слова	%MWi, %QWi, %QWAi, %SWi	Прямое значение, %MWi, %KWi, %INW, %IW, %IAi, %QNW, %QW, %QWAi, %SWi, %BLK.x
Двойные слова	%MDi	Прямое значение, %MDi, %KDi

Примечание: (1) С этим оператором, Op2 не может быть прямым значением. Функция ABS может использоваться только с двойными словами (%MD и %KD) и словами с ПТ (%MF and %KF). Следовательно, OP1 и OP2 должны быть двойными словами или словами с ПТ.

Условия переполнения и ошибки

Сложение

- Переполнение во время операции над словами
  Если результат превышает емкость слова результата, бит %S18 (переполнение) устанавливается в 1 и результат не значим (см. Пример 1, следующая страница). Программа пользователя управляет битом %S18.

Примечание:
Для двойных слов, пределы -2147483648 и 21474836487.

Умножение

- Переполнение во время операции
  Если результат превышает емкость слова результата, бит %S18 (переполнение) устанавливается в 1 и результат не значим.

Деление / остаток

- Деление на 0
  Если делитель 0, деление невозможно и системный бит %S18 устанавливается в 1.
  Результат неправильный.
- Переполнение во время операции
  Если результат превышает емкость слова результата,бит %S18 устанавливается в 1.

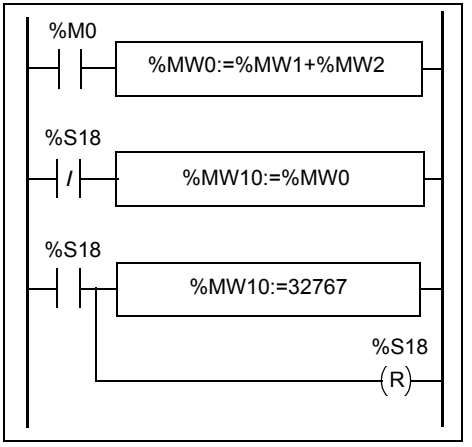
Извлечение квадратного корня

- Переполнение во время операции
  Извлечение квадратного корня выполняется только из положительных значений, поэтому, результат всегда положительный. Если операнд квадратного корня отрицательный, системный бит %S18 устанавливается в 1, и результат неправильный.

Примечание: Программа пользователя отвечает за управление системными битами %S17 и %S18.Они устанавливаются в 1 контроллером и должны сбрасываться программой для повторного использования (см. предыдущую страницу).

Примеры

Пример 1: переполнение во время сложения.




```
LD      %M0
[%MW0:=%MW1 + %MW2]

LDN     %S18
[%MW10:=%MW0]

LD      %S18
[%MW10:=32767]
R       %S18
```

Если %MW1 =23241 и %MW2=21853, действительный результат (45094) не может быть выражен одним 16-битным словом, бит %S18 устанавливается в 1 и полученный результат (-20442) неправильный. В этом примере, когда результат превышает 32767, его значение фиксируется в 32767.

Логические инструкции

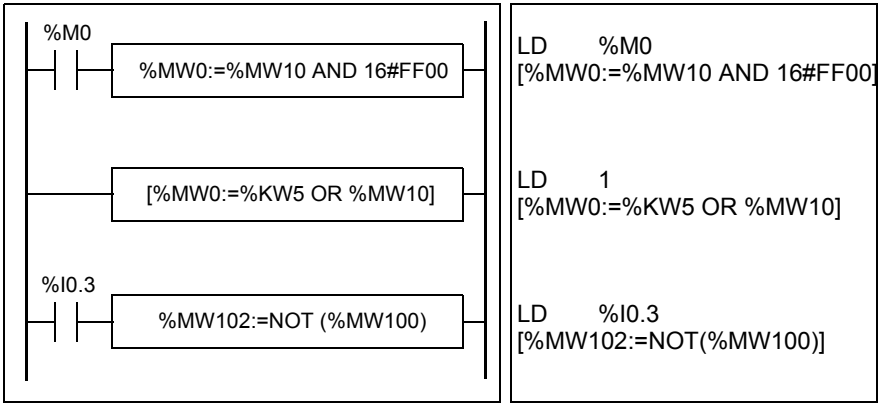
Введение

Логические инструкции используются для выполнения логических операций между двумя словами-операндами или над одним словом-операндом.
В следующей таблице перечислены типы логических инструкций.

Инструкция	Функция
AND	И (побитовое) между двумя операндами
OR	Логическое ИЛИ (побитовое) между двумя операндами
XOR	Исключающее ИЛИ (побитовое) между двумя операндами
NOT	Логическое отрицание (побитовое) операнда

Структура

Логические операции выполняются следующим образом:



Синтаксис

Синтаксис зависит от используемых операторов:

Оператор	**Синтаксис**	**Операнд 1 (Op1)**	**Операнды 2 и 3 (Op2 & 3)**
AND, OR, XOR	[Op1: = Op2 Оператор Op3]	%MWi, %QWi, %QWAi, %SWi	Прямое значение (1), %MWi, %KWi, %IW, %IWAi, %QW, %QWAi, %SWi, %BLK.x
NOT	[Op1:=NOT(Op2)]		

Примечание: (1) С NOT, Op2 не может быть прямым значением.

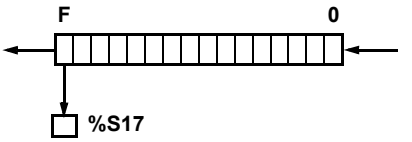
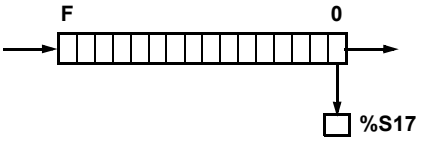
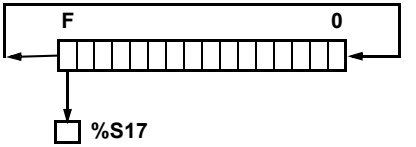
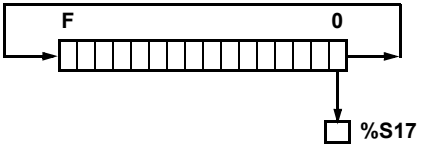
Пример

Ниже приведен пример инструкции логического И:
[%MW15:=%MW32 AND %MW12]

Инструкции сдвига

Введение

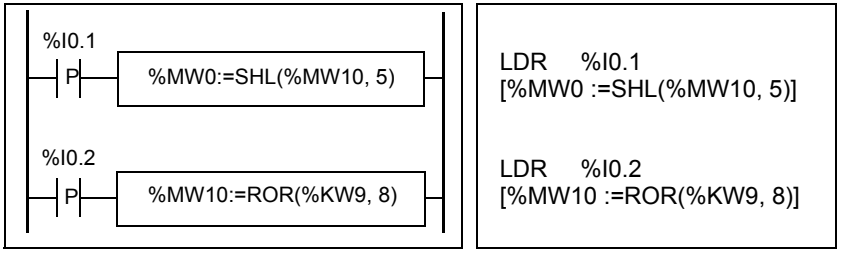
Инструкции сдвига перемещают биты операнда на заданное число позиций вправо или влево.
В следующей таблице перечислены типы инструкций сдвига.

Инструкция	Функция	
Логический сдвиг		
SHL(op2,i)	Логический сдвиг на i позиций влево.	
SHR(op2,i)	Логический сдвиг на i позиций вправо.	
Циклический сдвиг		
ROR(op2,i)	Циклический сдвиг на i позиций влево.	
ROL(op2,i)	Циклический сдвиг на i позиций вправо.	

Примечание: Системный бит %S17 (См. *Системные биты (%S), стр. 434*) выхода за границы емкости.

Структура

Операции сдвига выполняются следующим образом:



Синтаксис

Синтаксис зависит от используемых операторов, как указано в следующей таблице.

Оператор	**Синтаксис**
SHL, SHR	[Op1: = Оператор (Op2,i)]
ROL, ROR	

Операнды:

Типы	**Операнд 1 (Op1)**	**Операнд 2 (Op2)**
Слова	%MWi, %QWi, %QWAi, %SWi	%MWi, %KWi, %IW, %IWAi, %QW, %QWAi, %SWi, %BLK.x
Двойное слово	%MDi	%MDi, %KDi

Инструкции преобразования

Введение

Инструкции преобразования выполняют преобразование между различными формами представления чисел.
В следующей таблице перечислены типы инструкций преобразования.

Инструкция	Функция
BTI	Преобразование BCD --> двоичный код
ITB	Преобразование двоичный код --> BCD

Обзор кода BCD

Двоично-десятичный код (BCD) представляет десятичную цифру (от 0 до 9) кодированием четырех двоичных битов. 16-битное слово может содержать число из 4 цифр (0000 - 9999), а 32-битное двойное слово может содержать число из 8-ми цифр.
Во время преобразования системный бит %S18 устанавливается в 1, если значение не BCD. Этот бит должен проверяться и сбрасываться в 0 программой.
BCD представление десятичных цифр:

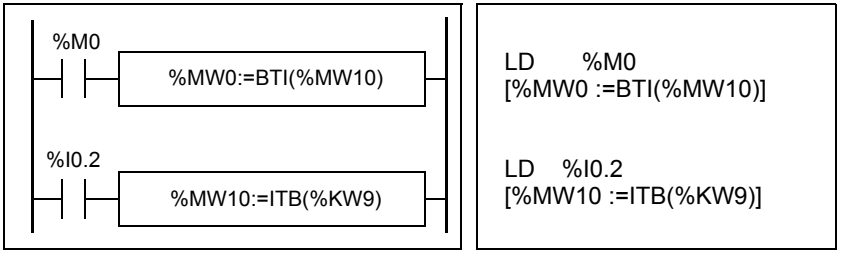
Десятичная	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

Примеры:
- Слово %MW5 выражает BCD значение "2450", которое соответствует двоичному значению: 0010 0100 0101 0000
- Слово %MW12 выражает десятичное значение "2450", которое соответствует двоичному значению: 0000 1001 1001 0010

Слово %MW5 преобразовывается в слово %MW12 инструкцией BTI.
Слово %MW12 преобразовывается в слово %MW5 инструкцией ITB.

Структура

Операции преобразования выполняются следующим образом:



Синтаксис

Синтаксис зависит от используемых операторов, как указано в следующей таблице.

Оператор	Синтаксис
BTI, ITB	[Op1: = Оператор (Op2)]

Операнды:

Тип	Операнд 1 (Op1)	Операнд 2 (Op2)
Слова	%MWi, %QWi, %QWAi, %SWi	%MWi, %KWi, %IW, %IWAi, %QW, %QWAi, %SWi, %BLK.x
Двойное слово	%MDi	%MDi, %KDi

Пример приложения:

Инструкция BTI используется для выдачи установочного значения на входы контроллера через BCD-кодированные колесики.
Инструкция ITB используется для отображения цифровых значений (например, результат вычислений, текущее значение функционального блока) на BCD-кодированных дисплеях.

Инструкции преобразования одинарных/двойных слов

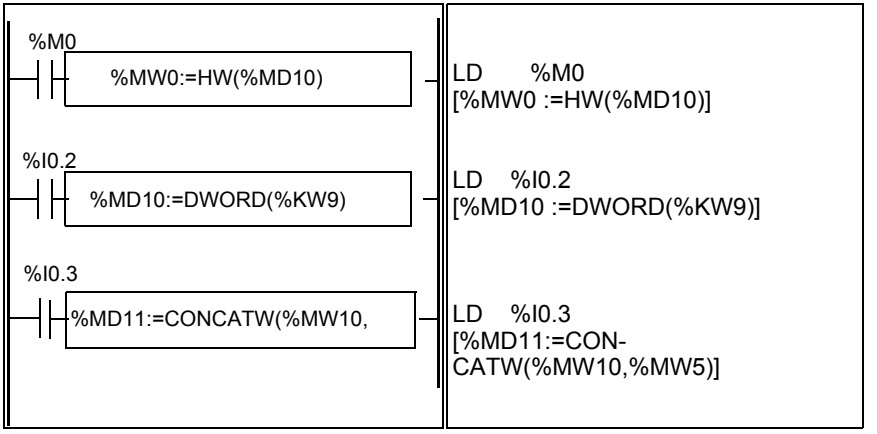
Введение

В следующей таблице описаны инструкции, используемые для выполнения преобразований между одинарными и двойными словами:

Инструкция	Функция
LW	Младший значащий байт двойного слова извлекается в слово.
HW	Старший значащий байт двойного слова извлекается в слово.
CONCATW	Соединение двух слов в двойное слово.
DWORD	Преобразование 16-битного слова в 32-битное двойное слово.

Структура

Операции преобразования выполняются следующим образом:



Синтаксис

Синтаксис зависит от используемых операторов, как указано в следующей таблице:

Оператор	Синтаксис	Операнд 1 (Op1)	Операнд 2 (Op2)	Операнд 3 (Op3)
LW, HW	Op1 = Оператор (Op2)	%MWi	%MDi, %KDi	[-]
CONCATW	Op1 = Оператор (Op2, Op3))	%MDi	%MWi, %KWi, прямое значение	%MWi, %KWi, прямое значение
DWORD	Op1 = Оператор (Op2)	%MDi	%MWi, %KWi	[-]

14.4 Инструкции программы

Обзор

Цель этого раздела

В этом разделе представлено введение в инструкции программы

Содержание раздела

Этот раздел содержит следующие темы:

Инструкции END

Введение

Инструкции End определяют конец выполнения сканирования программы.

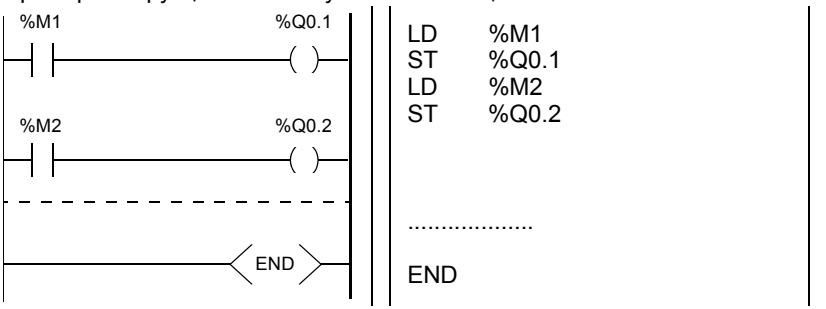
END, ENDC и ENDCN

Доступны три различные инструкции конца:
- END: безусловный конец программы
- ENDC: конец программы, если булевский результат предшествующей инструкции сравнения равен 1
- ENDCN: конец программы, если булевский результат предшествующей инструкции сравнения равен 0

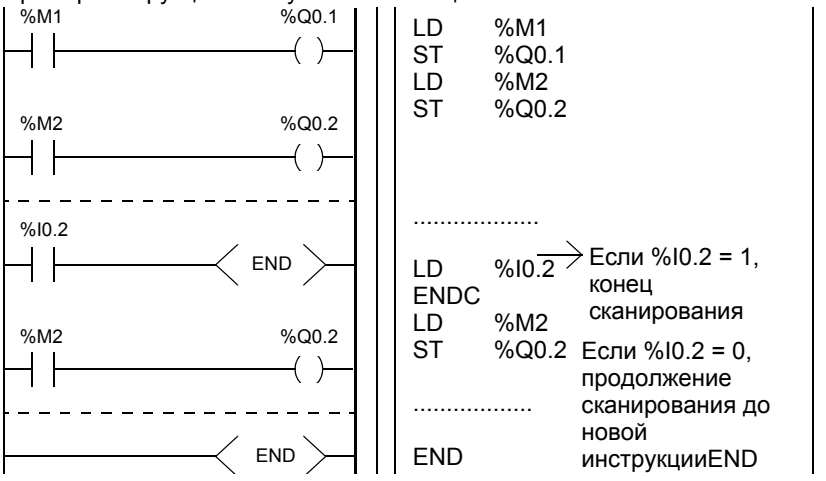
По умолчанию (нормальный режим), когда активируется конец программы, обновляются выходы, и начинается следующее сканирование.
Если сканирование является периодическим, когда достигается конец периода, обновляются выходы, и начинается следующее сканирование.

Примеры

Пример инструкции END безусловного конца .



Пример инструкции END условного конца.



Инструкция NOP

NOP

Инструкция NOP не выполняет никакой операции. Используйте её, чтобы "зарезервировать" строки программы, чтобы позднее вставлять туда инструкции без необходимости изменять номера строк.

Инструкции переходов

Введение

Инструкции переходов вызывают немедленное прерывание программы и выполнение её со строки, содержащей метку %Li
(i = с 1 до 16 для компактных и с 1 до 63 для других).

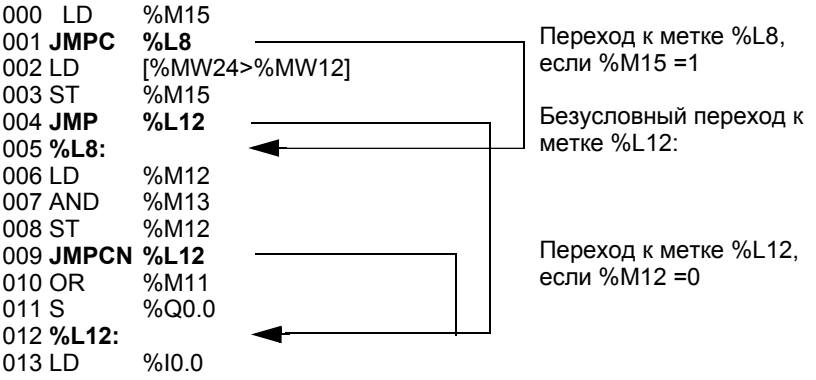
JMP, JMPC и JMPCN

Доступны три различные инструкции перехода:

- JMP: безусловный переход
- JMPC: переход, если булевский результат предшествующей инструкции сравнения равен 1
- JMPCN: переход, если булевский результат предшествующей инструкции сравнения равен 0

Примеры

Примеры инструкций перехода.



Советы

- Инструкции переходов не разрешены в пределах круглых скобок и не должны размещаться между инструкциями AND(, OR(и закрывающейся круглой скобкой ")".
- Метка может быть размещена только перед инструкциями LD, LDN, LDR, LDF или BLK
- Метка с № %Li должна быть определена в программе только один раз.
- Переход выполняется к строке программы выше или ниже текущей. Когда переход осуществляется выше, необходимо обратить внимание на время сканирования программы. Увеличенное время сканирования может вызвать срабатывание сторожевого таймера.

Инструкции подпрограмм

Введение

Инструкции подпрограмм вызывают выполнение подпрограммы и возврат к выполнению основной программы.

SRn, SRn: и RET.

Подпрограммы состоят из трех шагов:

- Инструкция **SRn** вызывает подпрограмму по метке SRn, если булевский результат предшествующей инструкции сравнения равен 1.
- Подпрограмма обозначается меткой **SRn:**, с n = 0 до 15 для TWDLCAA10DRF, TWDLCAA16DRF и 0 до 63 для остальных контроллеров.
- Инструкция **RET**, размещенная в конце подпрограммы, возвращает поток выполнения программы к основной программе.

Пример

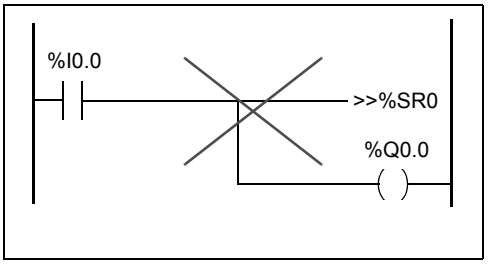
Пример инструкций подпрограмм.

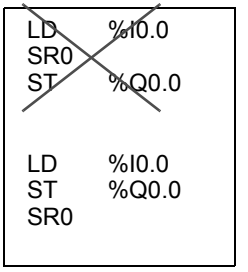
```
000  LD     %M15
001  AND    %M5
002  ST     %Q0.0
003  LD     [%MW24>%MW12]
004  SR8                          Переход к подпрограмме SR8
005  LD     %I0.4
006  AND    M13
007  _
008  _
009  _
010  END

011  SR8:
012  LD     1
013  IN     %TM0
014  LD     %TM0.Q
015  ST     %M15
010  RET                          Возврат к основной программе

.....................
```

Советы

- Подпрограмма не должна вызывать другую подпрограмму.
- Инструкции подпрограмм не разрешены в круглых скобках и не должны размещаться между инструкциями AND(, OR(и закрывающейся круглой скобкой ")".
- Метка может быть размещена только перед инструкциями LD или BLK, помечающей начало булевого уравнения (или ступени).
- Вызов подпрограммы не должен следовать за инструкцией присваивания, потому что подпрограмма может изменить содержимое булевского аккумулятора. Поэтому после возврата, он может иметь значение, отличное от того, которое было до вызова. См. следующий пример.

Пример программирования подпрограммы.





Дополнительные инструкции

15

Обзор

Предмет этой главы

В этой главе представлено подробное описание инструкций и функциональных блоков, которые используются для создания сложных управляющих программ для программируемых контроллеров Twido.

Содержание главы

Эта глава содержит следующие разделы:

15.1 Дополнительные функциональные блоки

Обзор

Цель этого раздела

В этом разделе представлено введение в дополнительные функциональные блоки, включая примеры программирования.

Содержание раздела

Этот раздел содержит следующие темы:

Битовые объекты и слова, связанные с дополнительными функциональными блоками

Введение

Дополнительные функциональные блоки используют типы выделенных слов и битов, аналогичные стандартным функциональным блокам.
Дополнительные функциональные блоки включают:

- Регистры LIFO/FIFO (%R)
- Барабанные контроллеры (%DR)
- Быстрые счетчики (%FC)
- Очень быстрые счетчики (%VFC)
- Выход импульсов широтной модуляции (%PWM)
- Выход генератора импульсов (%PLS)
- Сдвигающий регистр битов (%SBR)
- Сдвигающий счетчик (%SC)
- Блок контроля обмена (%MSG)

Объекты, достижимые программой

Следующая таблица содержит обзор слов и битов, достижимых программой, которые ассоциируются с разнообразными дополнительными функциональными блоками. Обратите внимание, что доступ к режиму записи в таблице зависит от настройки “Регулируемый”, выбранной во время конфигурации. Её установка позволяет или запрещает доступ к словам и битам через TwidoSoft или интерфейс оператора.

Дополнительный функциональный блок	**Связанные слова и биты**		**Адрес**	**Доступ к режиму записи**
%R	Слово	Регистровый вход	%Ri.I	Да
	Слово	Регистровый выход	%Ri.O	Да
	Бит	Регистровый выход Полон	%Ri.F	Нет
	Бит	Регистровый выход Пуст	%Ri.E	Нет
%DR	Слово	Номер текущего шага	%DRi.S	Да
	Бит	Последний шаг соответствует текущему шагу	%DRi.F	Нет
%FC	Слово	Текущее значение	%FCi.V	Да
	Слово	Предустановленное значение	%FCi.P	Да
	Бит	Готово	%FCi.D	Нет

Дополнительный функциональный блок	Связанные слова и биты		Адрес	Доступ к режиму записи
%VFC	Слово	Текущее значение	%VFCi.V	Нет
	Слово	Предустановленное значение	%VFCi.P	Да
	Бит	Направление счета	%VFCi.U	Нет
	Слово	Захваченное значение	%VFCi.C	Нет
	Слово	Пороговое значение 0	%VFCi.S0	Да
	Слово	Пороговое значение 1	%VFCi.S1	Да
	Бит	Переполнение	%VFCi.F	Нет
	Бит	Рассчитанная частота	%VFCi.M	Да
	Бит	Рефлексный выход 0 разрешен	%VFCi.R	Да
	Бит	Рефлексный выход 1 разрешен	%VFCi.S	Да
	Бит	Порог 0 выхода	%VFCi.TH0	Нет
	Бит	Порог 1 выхода	%VFCi.TH1	Нет
	Бит	Масштаб времени для измерения частоты	%VFCi.T	Да
%PWM	Слово	Процентное отношение импульса в 1 ко всему периоду.	%PWMi.R	Да
	Слово	Предустановленный период	%PWMi.P	Да
%PLS	Слово	Число импульсов	%PLSi.N	Да
	Слово	Предустановленное значение	%PLSi.P	Да
	Бит	Текущий выход разрешен	%PLSi.Q	Нет
	Бит	Генерирование выполнено	%PLSi.D	Нет
%SBR	Бит	Регистровый бит	%SBRi.J	Нет
%SC	Бит	Бит счетчика шагов	%SCi.j	Да
%MSG	Бит	Готово	%MSGi.D	Нет
	Бит	Ошибка	%MSGi.E	Нет

Принципы программирования для дополнительных функциональных блоков

Обзор

Все приложения Twido хранятся в форме Списка, даже если они написаны в редакторе лестничных диаграмм, поэтому, контроллеры Twido можно назвать "машинами списка." Термин "обратимость" относится к возможности TwidoSoft представлять приложение Списка инструкций в виде лестничной диаграммы и наоборот. По умолчанию, все лестничные программы обратимы.
Также как и основные функциональные блоки, дополнительные блоки должны учитывать правила обратимости. Структура обратимых функциональных блоков в языке Списка требует использования следующих инструкций:

- **BLK**: Помечает начало блока и часть ввода функционального блока
- **OUT_BLK**: Помечает начало части вывода функционального блока
- **END_BLK**: Помечает конец функционального блока

Примечание: Использование обратимых инструкций функционального блока является обязательным для правильного функционирования программ Списка. Возможно программировать некоторые инструкции на языке Списка без обратимости.

Выделенные входы и выходы

Дополнительные функции Быстрого счетчика, Очень быстрого счетчика, Генератора импульсов и ШИМ используют выделенные входы и выходы, но эти биты не зарезервированы для эксклюзивного использования каким то одним блоком. Лучше управлять использованием выделенных ресурсов. При использовании этих дополнительных функций, Вы должны управлять выделением входов и выходов. TwidoSoft помогает в конфигурировании этих ресурсов отображением подробностей конфигурирования входов/выходов и предупреждением, если выделенный вход или выход уже используются сконфигурированным функциональным блоком.
В следующей таблице суммируются зависимости выделенных входов и выходов и особых функций.
При использовании с функциями счета:

Входы	Использование
%I0.0.0	%VFC0: Управление направлением или Фаза B
%I0.0.1	%VFC0: Вход импульса или Фаза A
%I0.0.2	%FC0: Вход импульса или вход предустановки %VFC0
%I0.0.3	%FC1:Вход импульса или вход захвата %VFC0
%I0.0.4	%FC2: Вход импульса или вход захвата %VFC1
%I0.0.5	%VFC1: Вход предустановки

Входы	Использование
%I0.0.6	%VFC1: Управление направлением или Фаза B
%I0.0.7	%VFC1: Вход импульса или Фаза A

При использовании со счетчиками или особыми функциями:

Выход	Использование
%Q0.0.0	%PLS0 или выход PWM0
%Q0.0.1	%PLS1 или выход PWM1
%Q0.0.2	Рефлексные выходы для %VFC0
%Q0.0.3	
%Q0.0.4	Рефлексные выходы для %VFC1
%Q0.0.5	

Использование выделенных входов и выходов

TwidoSoft вынуждает использовать следующие правила для использования выделенных входов и выходов.

- Каждый функциональный блок, который использует выделенные Вх/Вых должен быть сконфигурирован до обращения в приложении. Выделенный Вх/Вых выделяется только при конфигурировании функционального блока, а не при использовании его в программе.
- После конфигурирования функционального блока его выделенные входы и выходы не могут использоваться приложением или другим функциональным блоком.
  Например, если Вы конфигурируете %PLS0, Вы не можете использовать %Q0.0.0 в %DR0 (барабанный контроллер) или в логике приложения (например, ST %Q0.0.0).
- Если функциональному блоку требуется выделенный вход или выход, который уже используется приложением или другим функциональным блоком, этот функциональный блок не может быть сконфигурирован.
  Например, если Вы конфигурируете %FC0 как счетчик вверх, Вы не можете сконфигурировать %VFC0 для использования %I0.0.2 как входа захвата.

Примечание: чтобы изменить использование выделенного вх/вых, I/O, освободите функциональный блок, установив тип объекта в "не используемый," и затем удалите обращения к функциональному блоку в приложении.

Регистровый функциональный блок LIFO/FIFO (%Ri)

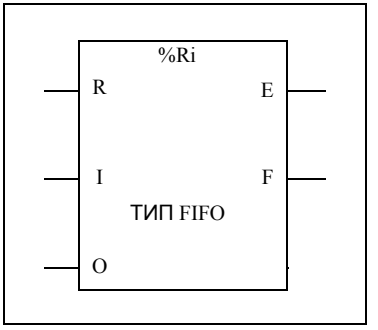
Введение

Регистр это блок памяти, который может хранить до 16 слов из 16 битов каждое двумя различными способами:

- Очередь (First In, First Out), известная как FIFO.
- Стек (Last In, First Out), известный как LIFO.

Иллюстрация

На следующем рисунке показан регистровый функциональный блок.



Регистровый функциональный блок

Параметры

Функциональный блок счетчика имеет следующие параметры:

Параметр	Метка	Значение
Номер регистра	%Ri	0 до 3.
Тип	FIFO или LIFO	Очередь или Стек.
Входное слово	%Ri.I	Входное слово регистра. Может быть прочитано, проверено и записано.
Выходное слово	%Ri.O	Выходное слово регистра. Может быть прочитано, проверено и записано.
Вход Сохранить (или инструкция)	I (In)	По фронту сохраняет содержимое слова %Ri.I в регистре.
Вход Извлечь (или инструкция)	O (Out)	По фронту загружает слово данных регистра в слово %Ri.O.
Вход Сброс (или инструкция)	R (Reset)	В состоянии 1, инициализирует регистр.
Выход Пуст	E (Empty)	Связанный бит %Ri.E показывает, что регистр пуст. Может быть проверен.
Выход Полон	F (Full)	Связанный бит %Ri.F показывает, что регистр полон. Может быть проверен.

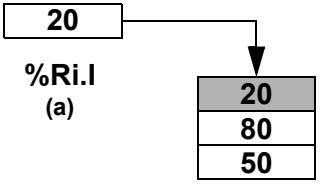
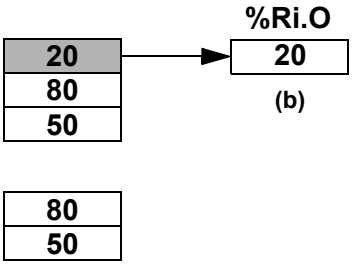
Функционирование LIFO

Введение

В LIFO (Last In, First Out), последний введенный элемент извлекается первым.

Функционирование

В следующей таблице описана работа LIFO.

Шаг	Описание	Пример
1	При получении запроса на сохранение (фронт на входе I или активизация инструкции I), содержимое входного слова %Ri.I (которое уже загружено) сохраняется в вершине стека (Рис. a).Когда стек полон (выход F=1), дальнейшее сохранение невозможно.	Сохранение содержимого %Ri.I в вершине стека. 
2	При получении запроса на извлечение (фронт на входе O или активизация инструкции O), верхнее слово данных (последнее введенное слово) загружается в слово %Ri.0 (Рис. b). Когда регистр пуст (выход E=1), дальнейшее извлечение невозможно. Выходное слово %Ri.O не изменяется и сохраняет свое значение.	Извлечение слова данных из вершины стека. 
3	Стек может быть сброшен в любое время (состояние 1 на входе R или активизация инструкции R). Элемент, который определяет указатель, становится верхним в стеке.	

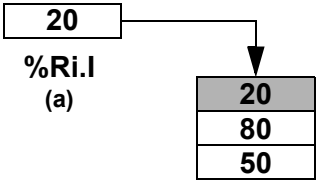
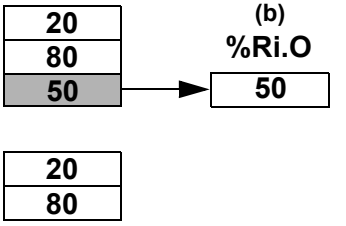
Функционирование FIFO

Введение

В FIFO (First In, First Out), первый введенный элемент извлекается первым.

Функционирование

В следующей таблице описана работа FIFO.

Шаг	Описание	Пример
1	При получении запроса на сохранение (фронт на входе I или активизация инструкции I), содержимое входного слова %Ri.I (которое уже загружено) сохраняется в вершине очереди (Рис. a).Когда очередь полна (выход F=1), дальнейшее сохранение невозможно.	Сохранение содержимого %Ri.I в вершине очереди. 
2	При получении запроса на извлечение (фронт на входе O или активизация инструкции O), нижнее слово данных (первое введенное слово) загружается в слово %Ri.0 (Рис. b). Когда регистр пуст (выход E=1), дальнейшее извлечение невозможно. Выходное слово %Ri.O не изменяется и сохраняет свое значение.	Извлечение первого элемента данных, который затем загружается в %Ri.O. 
3	Очередь может быть сброшена в любое время (состояние 1 на входе R или активизация инструкции R).	

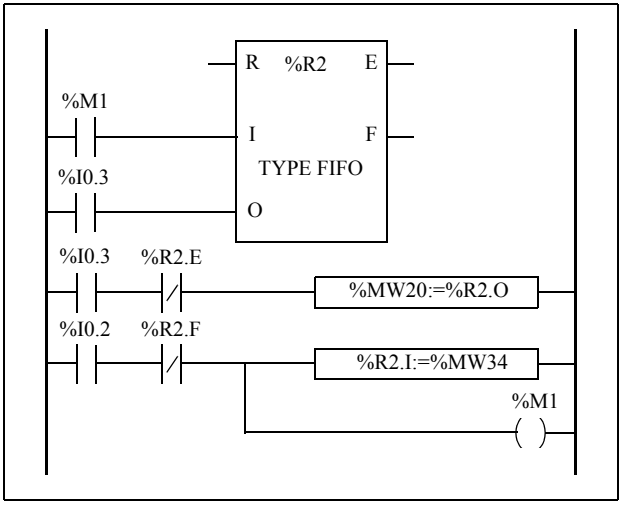
Программирование и конфигурирование регистров

Введение

В следующем примере программирования показано, как содержимое слова памяти (%MW34) загружается в регистр (%R2.I) при получении запроса сохранения (%I0.2), если регистр %R2 не полон (%R2.F = 0). Запрос сохранения в регистре производится %M1. Запрос извлечения производится входом %I0.3, и %R2.O загружается в %MW20, если регистр не пуст (%R2.E = 0).

Пример программирования

Ниже приведен пример регистрового функционального блока с примером обратимого и необратимого программирования.



Лестничная диаграмма

```
BLK          %R2
LD           %M1
I
LD           %I0.3
O
END_BLK
LD           %I0.3
ANDN         %R2.E
[%MW20:=%R2.O]
LD           %I0.2
ANDN         %R2.F
[%R2.I:=%MW34]
ST           %M1
```

Обратимая программа

```
LD           %M1
I            %R2
LD           %I0.3
O            %R2
ANDN         %R2.E
[%MW20:=%R2.O]
LD           %I0.2
ANDN         %R2.F
[%R2.I:=%MW34]
ST           %M1
```

Необратимая программа

Конфигурирование

Единственный параметр, который должен быть введен при конфигурировании, это тип регистра:

- FIFO (по умолчанию), или
- LIFO

Особые случаи

В следующей таблице содержится список особых случаев для программирования функционального блока Сдвигающего регистра битов:

Особый случай	Описание
Влияние холодного перезапуска (%S0=1)	Инициализирует содержимое регистра. Выходной бит %Ri.E, связанный с выходом E, установлен в 1.
Влияние теплого перезапуска (%S1=1) после остановки контроллера	Не влияет на текущее значение регистра и на состояние его выходных битов.

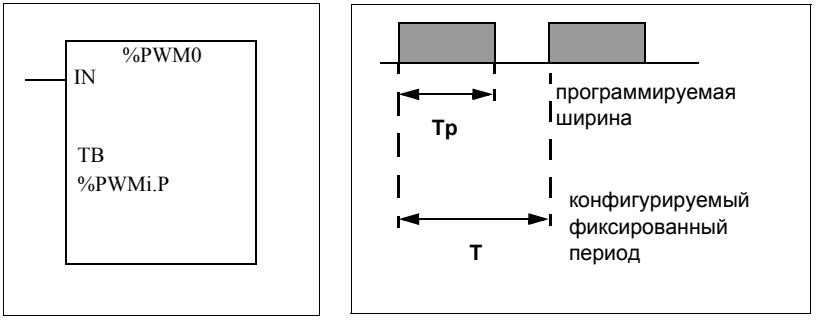
Функциональный блок генератора широтной модуляции (%PWM)

Введение

Функциональный блок Генератора широтной модуляции (%PWM) генерирует прямоугольные колебательные сигналы на выделенных выходных каналах %Q0.0.0 или %Q0.0.1, с переменной шириной, и следовательно, рабочим циклом. Контроллеры с релейными выходами для этих двух каналов не поддерживают эту функцию, в связи с ограничением частоты.
Доступны два блока %PWM. %PWM0 использует выделенный выход %Q0.0.0 и %PMW1 использует выделенный выход %Q0.0.1. Функциональные блоки %PLS конкурируют за использование тех же выходов, так что вы должны выбрать одну из двух функций.

Иллюстрация

Блок ШИМ и временная диаграмма:



Параметры

В следующей таблице перечислены параметры для функционального блока PWM.

Параметр	Метка	Описание
Масштаб по оси времени	TB	0.142 мс, 0.57 мс, 10 мс, 1 с (значение по умолчанию)
Выбранный период	%PWMi.P	0 < %PWMi.P <= 32767 с масштабом 10 мс или 1 с 0 < %PWMi.P <= 255 с масштабом 0.57 мс или 0.142 с 0 = Функция не используется
Рабочий цикл	%PWMi.R	Это значение задает отношение сигнала в состоянии 1 к периоду. Ширина Tp эквивалентна: Tp = T * (%PWMi.R/100). приложение пользователя записывает значение для %PWMi.R. Это слово контролирует рабочий цикл периода. Для определения T, см. "Диапазон периодов" ниже. Значение по умолчанию = 0, а значения, больше 100, считаются равными 100.
Вход генерации импульсов	IN	В состоянии 1, сигнал ШИМ генерируется на выходном канале. В состоянии 0, выходной канал установлен в 0.

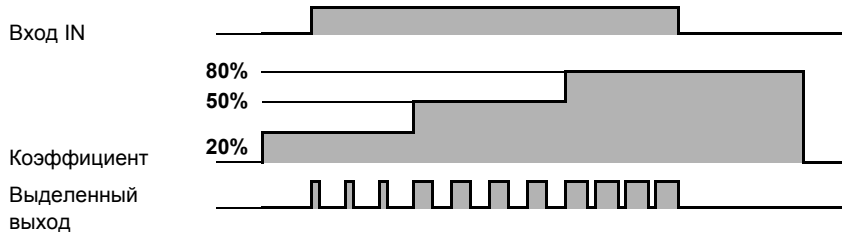
Диапазон периодов

Предустановленное значение и масштаб по оси времени могут быть изменены во время конфигурирования. Они используются для фиксирования периода сигнала T=%PWMi.P * TB. Чем ниже требуемый коэффициент, тем выше должно быть выбранное значение %PWMi.P. Диапазон доступных периодов:

- 0.142 мс до 36.5 мс с шагом 0.142 мс (27.4Гц до 7кГц)
- 0.57 мс до 146 мс с шагом 0.57 мс (6.84 Гц до 1.75 кГц)
- 10 мс до 5.45 мин с шагом 10 мс
- 1 сек до 9.1 часов с шагом 1 сек

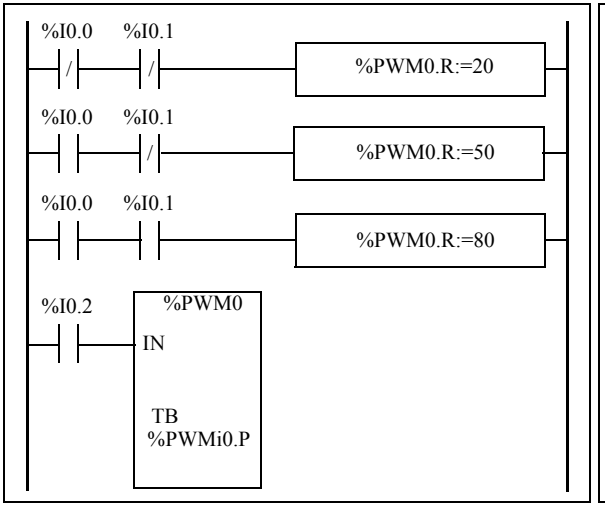
Функционирование

Частота выходного сигнала устанавливается во время конфигурирования, выбором масштаба TB и предустановленного %PWMi.P. Изменение % PWMi.R рабочего цикла в программе модулирует ширину сигнала. Ниже приведена иллюстрация диаграммы импульсов для ФБ PWM с изменением рабочего цикла.



Программирование и конфигурирование

В этом примере ширина сигнала изменяется программой в соответствии с состоянием входов контроллера %I0.0.0 и %I0.0.1.
Если %I0.0.1 и %I0.0.2 установлены в 0, коэффициент %PWM0.R установлен в 20%, длительность сигнала в состоянии 1 равна: 20 % x 500 мс = 100 мс.
Если %I0.0.0 установлен 0 и %I0.0.1 установлен 1, коэффициент %PWM0.R установлен в 50% (Длительность 250 мс).
Если %I0.0.0 и %I0.0.1 установлены в 1, коэффициент %PWM0.R установлен в 80% (длительность 400 мс).
Пример программирования:




```
LDN     %I0.0
ANDN    %I0.1
[%PWM0.R:=20]
LD      %I0.0
ANDN    %I0.1
[%PWM0.R:=50]
LD      %I0.0
AND     %I0.1
[%PWM0.R:=80]
BLK     %PWM0
LD      %I0.2
IN
END_BLK
```

Особые случаи

В следующей таблице приведен список особых случаев ФБ PWM.

Особый случай	Описание
Влияние холодного перезапуска (%S0=1)	Устанавливает %PWMi.R коэфф. в 0. Значение %PWMi.P сбрасывается в сконфигурированное, и это отменит любые изменения, сделанные в Редакторе анимационных таблиц или опциональном дисплее оператора.
Влияние теплого перезапуска (%S1=1)	Не влияет.
Результат того, что выходы выделены блоку %PWM	Форсирование выхода %Q0.0.0 или %Q0.0.1 при помощи программы не останавливает генерирование сигналов.

Функциональный блок генератора импульсов (%PLS)

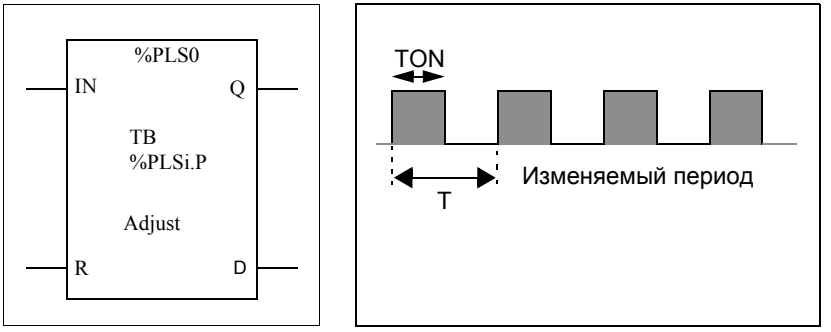
Введение

Функциональный блок %PLS используется для генерирования прямоугольных колебательных сигналов. Доступны две функции %PLS на выделенных выходных каналах %Q0.0.0 или %Q0.0.1. Функциональный блок %PLS допускает только одну ширину сигнала, или рабочий цикл, 50%. Вы можете выбрать либо ограничение количества импульсов или период, когда выполняется последовательность импульсов. Это может быть определено во время конфигурирования и/или обновлено приложением пользователя.

Примечание: Контроллеры с релейными выходами для этих двух каналов не поддерживают функцию %PLS.

Представление

Пример функционального блока генератора импульсов:



- TON=T/2 для 0.142мс и 0.57мс масштаб
  = (%PLSi.P*TB)/2
- TON=[целая часть(%PLSi.P)/2]*TB для 10мс до 1с масштаба.

Спецификации

В следующей таблице содержатся характеристики функционального блока PLS:

Функция	Объект	Описание
Масштаб	TB	0.142 мс, 0.57 мс, 10 мс, 1 сек
Предустановленный период	%PLSi.P	Импульсы на выходе %PLS1 не останавливаются, когда достигнут %PLS1.N для масштабов 0.142 мс и 0.57 мс. ● 1 < %PLSi.P <= 32767 для масштабов 10 мс или 1 с ● 0 < %PLSi.P <= 255 для масштабов 0.57 мс или 0.142 мс ● 0 = Функция не используется. Чтобы получить хорошую точность от рабочих циклов с масштабом 10мс и 1с, рекомендуется, чтобы %PLSi >= 100 , если P нечетное.
Число импульсов	%PLSi.N	Число импульсов, сгенерированных за период T, может быть ограничено 0 < %PLSi.N < 32767. Значение по умолчанию =0. Для генерирования неограниченного числа импульсов, установите %PLSi.N в ноль. Число импульсов можно всегда изменить, в не зависимости от настройки “Регулируемый “.
Регулируемый	Y/N	Если установлен в Y, возможно изменять предустановленное значение %PLSi.P через HMI или Редактор анимационных таблиц. Установка в N означает, что нет доступа к предустановке.
Вход генерации импульсов	IN	В состоянии 1, генерация импульсов производится на выделенный выходной канал. В состоянии 0, выходной канал установлен в 0.
Вход сброса	R	В состоянии 1, выходы %PLSi.Q и %PLSi.D установлены в 0. Число импульсов, генерируемых за период T, установлено в 0.
Текущий выход генерации импульсов	%PLSi.Q	В состоянии 1, указывает, что сконфигурировано генерирование импульсов на выделенный выходной канал.
Выход Генерация завершена	%PLSi.D	В состоянии 1, генерирование сигналов завершено. Число желаемых импульсов было достигнуто.

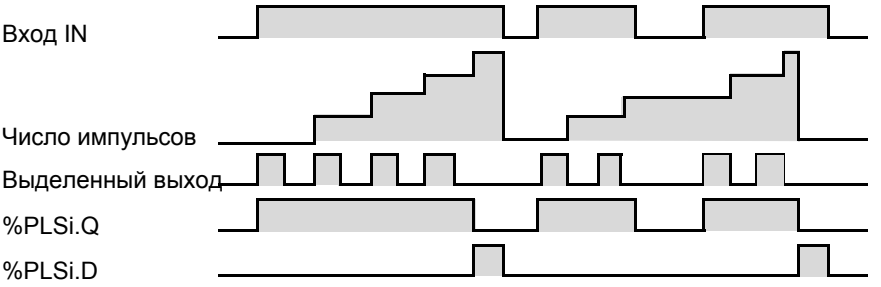
Диапазон периодов

Предустановленное значение и масштаб могут быть изменены во время конфигурирования. Они используются для фиксирования периода сигнала T=%PLSi.P * TB. Диапазон доступных периодов:

- 0.142 мс до 36.5 мс с шагом 0.142 мс (27.4Гц до 7кГц)
- 0.57 мс до 146 мс с шагом 0.57 мс (6.84 Гц до 1.75 кГц)
- 20 мс до 5.45 мин с шагом 10 мс
- 2 сек до 9.1 часов с шагом 1 сек

Функционирование

Ниже приведена иллюстрация функционального блока %PLS.



Особые случаи

Особый случай	Описание
Влияние холодного перезапуска (%S0=1)	Устанавливает %PLSi.P в определенное при конфигурировании значение
Влияние теплого перезапуска (%S1=1)	Не влияет.
Влияние изменения предустановки (%PLSi.P)	Оказывает немедленный результат
Результат того, что выходы выделены блоку %PLS	Форсирование выхода %Q0.0.0 или %Q0.0.1 при помощи программы не останавливает генерирование сигналов.

Примечание: %PLSx.D устанавливается, когда было достигнуто желаемое количество импульсов. Он сбрасывается установкой входа IN или R в 1.

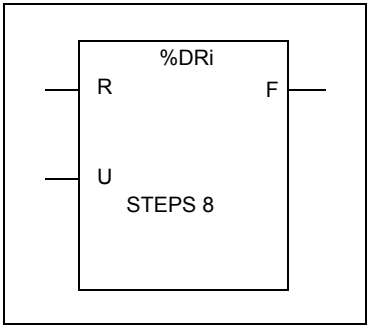
Функциональный блок барабанного контроллера (%DR)

Введение

Барабанный контроллер работает на принципах, похожих на принципы электромеханического барабанного контроллера, который изменяет шаг, в соответствии с внешними событиями. На каждом шаге, высшая точка кулачка дает команду, которая выполняется контроллером. В случае барабанного контроллера эти высшие точки выражаются состоянием 1 для каждого шага и связаны с выходными битами %Qi.j или внутренними битами %Mi, известными как управляющие биты.

Иллюстрация

Ниже приведена иллюстрация функционального блока барабанного контроллера.



функциональный блок барабанного контроллера.

Параметры

Функциональный блок барабанного контроллера имеет следующие параметры:

Параметр	Метка	Значение
Номер	%DRi	0 до 3 компактные контроллеры 0 до 7 модульные контроллеры
Номер текущего шага	%DRi.S	0<%DRi.S<7. Слово можно читать и записывать. Записываемое значение должно быть десятичным прямым значением. При записи эффект проявляется при следующем выполнении функционального блока.
Число шагов		1 до 8 (по умолчанию)
Вход для возврата в шаг 0 (или инструкция)	R (Reset)	В состоянии 1, устанавливает барабанный контроллер в шаг 0.
Вход продвижения (или инструкция)	U (Upper)	По фронту вызывает продвижение барабанного контроллера на один шаг и обновление управляющих битов.
Выход	F (Full)	Указывает, что текущий шаг равен последнему определенному шагу. Связанный бит %DRi.F может быть проверен (например, %DRi.F=1, если %DRi.S= число сконфигурированных шагов - 1).
Управляющие биты		Выходы внутренних битов, связанных с шагом (16 управляющих битов) и определенных в Редакторе конфигурации.

Работа функционального блока барабанного контроллера %DRi

Введение

Барабанный контроллер состоит из:

- Матрицы постоянных данных (кулачков), организованных в 8 шагов (0 до7), и 16 битов данных (состояния шагов), организованных в столбцы с номерами от 0 до F.
- Список управляющих битов связан с конфигурируемым выходом (%Qi.j.k) или словом памяти (%Mi). Во время текущего шага, управляющие биты принимают двоичные состояния, определенные для этого шага.

Пример в следующей таблице суммирует основные характеристики барабанного контроллера.

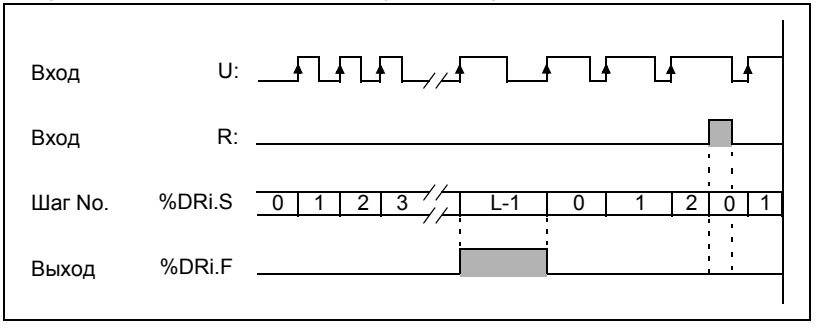
Столбец	**0**	**1**	**2**		**D**	**O**	**F**
Управляющие биты	%Q0.1	%Q0.3	%Q1.5		%Q0.6	%Q0.5	%Q1.0
0 шаг	0	0	1		1	1	0
1 шаг	1	0	1		1	0	0
5 шаг	1	1	1		0	0	0
6 шаг	0	1	1		0	1	0
7 шаг	1	1	1		1	0	0

Функционирование

В верхнем примере шаг 5 является текущим шагом, управляющие биты %Q0.1, %Q0.3 и %Q1.5 установлены в 1; управляющие биты%Q0.6, %Q0.5 и %Q1.0 установлены в 0. Номер текущего шага увеличивается при каждом фронте на входе U (или при активации инструкции U). Текущий шаг может быть изменен в программе.

Временная диаграмма

Следующая диаграмма иллюстрирует работу барабанного контроллера.



Особые случаи

Следующая таблица содержит список особых случаев функционирования барабанного контроллера.

Особый случай	Описание
Влияние холодного перезапуска (%S0=1)	Сброс барабанного контроллера в шаг 0 (обновление управляющих битов).
Влияние теплого перезапуска (%S1=1)	Обновление управляющих битов после текущего шага.
Влияние программного перехода	Факт, что барабанный контроллер больше не сканируется, означает, что управляющие биты не сброшены.
Обновление управляющих битов	Происходит только тогда, когда произошла смена шага, или при теплом или холодном перезапуске.
Влияние инструкций главного контрольного реле MCS/MCR	Если барабанный контроллер находится между двумя инструкциями MCS/MCR, это значит, что управляющие биты сбрасываются в 0, если булевский результат инструкции, предшествующей MCS, равен 0.

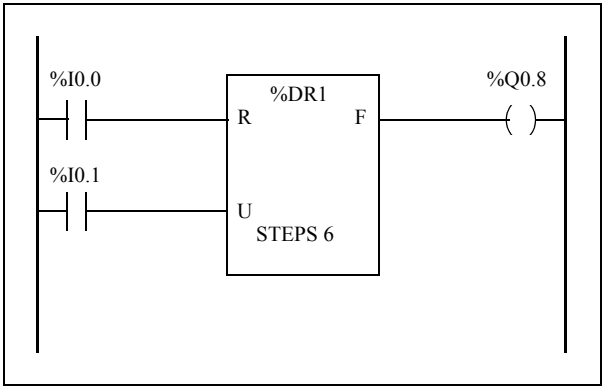
Программирование и конфигурирование барабанных контроллеров

Введение

Ниже приведен пример программирования и конфигурирования барабанного контроллера. Первые шесть входов с %Q0.0 по %Q0.5 активируются подряд каждый раз, когда вход %I0.1 устанавливается в 1. Вход I0.0 сбрасывает выходы в 0.

Пример программирования

На следующем рисунке приведен функциональный блок барабанного контроллера с примерами обратимого и необратимого программирования.



Лестничная диаграмма

```
BLK     %DR1
LD      %I0.0
R
LD      %I0.1
U
OUT_BLK
LD      F
ST      %Q0.8
END_BLK
```

Конфигурирование

Следующая информация определяется при конфигурировании:

- Число шагов: 6
- Состояния выходов (управляющих битов) для каждого шага барабанного контроллера.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Шаг 1:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Шаг 2:	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Шаг 3:	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Шаг 4:	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Шаг 5:	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Шаг 6:	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

- Присваивание управляющих битов.

1:	%Q0.0	4:	%Q0.1
2:	%Q0.2	5:	%Q0.3
3:	%Q0.4	6:	%Q0.5

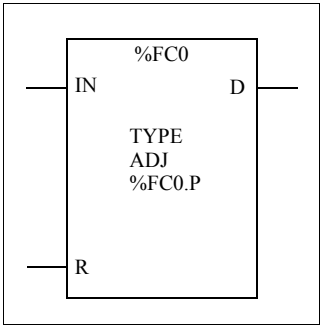
Функциональный блок быстрого счетчика (%FC)

Введение

Функциональный блок быстрого счетчика (%FC) может быть счетчиком вверх и счетчиком вниз. Он может считать фронты на цифровых входах на частотах до 5кГц. Быстрые счетчики управляются специальными аппаратными прерываниями, поэтому поддерживаемая максимальная частота может изменяться в зависимости от Вашего специфического приложения и аппаратной конфигурации.
Компактные контроллеры могут быть сконфигурированы на использование максимум трех быстрых счетчиков, модульные контроллеры могут использовать максимум два. Функциональные блоки быстрых счетчиков %FC0, %FC1 и %FC2 используют выделенные входы %I0.0.2, %I0.0.3 и %I0.0.4, соответственно. Эти биты не зарезервированы для использования исключительно этими блоками . Должно учитываться их выделение другим функциональным блокам.

Иллюстрация

Ниже приведен пример функционального блока счетчика.



Параметры

В следующей таблице перечислены параметры функционального блока счетчика.

Параметр	Метка	Описание
Функция	TYPE	Устанавливается при конфигурировании, может устанавливаться на счет вверх или вниз.
Предуст. значение	%FCi.P	Начальное значение устанавливается между 1 и 65635.
Регулируемый	Y/N	Если установлен в Y, возможно изменять предустановленное значение %FCi.P и %FCi.V через дисплей оператора и редактор анимационных таблиц. Если установлен в N, нет доступа к уставке.
Текущее значение	%FCi.V	Текущее значение увеличивается или уменьшается в зависимости от выбранной функции счета вверх или вниз. Для счета вверх текущее значение счета обновляется и может достигнуть 65536. Для счета вниз текущее значение это предустановленное значение %FCi.P и может уменьшаться до нуля.
Разрешение ввода	IN	В состоянии 1, текущее значение обновляется в соответствии с импульсами, приложенными к физическому входу. В состоянии 0, текущее состояние удерживается в последнем значении.
Сброс	%FCi.R	Используется для инициализации блока. В состоянии 1,текущее значение сбрасывается в 0, если сконфигурирован счет вверх, или устанавливается в %FCi.P, если сконфигурирован счет вниз. Бит готовности %FCi.D устанавливается в значение по умолчанию.
Готово	%FCi.D	Этот бит устанавливается в 1, когда %FCi.V достигает %FCi.P, при конфигурировании счета вверх, %FCi.V достигает нуля при конфигурировании счета вниз. Этот бит только для чтения сбрасывается только установкой %FCi.R в 1.

Особая заметка

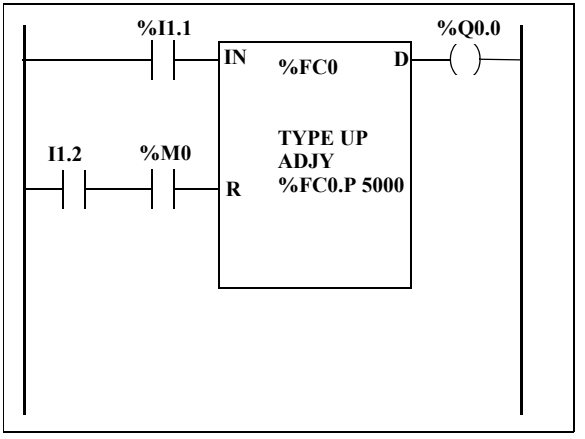
Если сконфигурирован регулируемым, приложение может изменять предустановленное значение %FCi.P и текущее значение %FCi.V в любое время. Но новое значение учитывается только, если активен вход сброса или по фронту на выходе %FCi.D. Это позволяет успешный различный счет без потери единичных импульсов.

Функционирование

Если сконфигурирован для счета вверх, когда фронт появляется на выделенном входе, текущее значение увеличивается на 1. Когда достигается предустановленное значение %FCi.P, выходной бит готовности %FCi.D устанавливается в 1, и ноль загружается в текущее значение %FCi.V.
Если сконфигурирован для счета вниз, когда фронт фрон появляется на выделенном входе, текущее значение уменьшается на 1. Когда достигается значение 0, выходной бит готовности %FCi.D устанавливается в 1, а предустановленное значение загружается в текущее значение %FCi.P.

Конфигурирование и программирование

В этом примере приложение считает количество единиц до 5000, пока %I1.1 установлен в 1. Вход для %FC0 это выделенный вход %I0.0.2. Когда достигается предустановленное значение, %FC0.D устанавливается в 1 и сохраняет это значение, пока %FC0.R не управляется результатом "AND" на %I1.2 и %M0.




```
BLK     %FC0
LD      %I1.1
IN
LD      %I1.2
AND     %M0
R
OUT_BLK
LD D
ST %Q0.0
END_BLK
```

Особые случаи

Следующая таблица содержит список особых случаев функционирования функционального блока %FC:

Особый случай	Описание
Влияние холодного перезапуска (%S0=1)	Сбрасывает все атрибуты %FC к значениям, сконфигурированным пользователем или приложением.
Влияние теплого перезапуска (%S1=1)	Не влияет.
Влияние остановки контроллера	Блок %FC продолжает считать, настройки параметров те же, которые были разрешены в момент остановки.

Функциональный блок очень быстрого счетчика (%VFC)

Введение

Функциональный блок очень быстрого счетчика (%VFC) может быть сконфигурирован TwidoSoft на выполнение любой из ниже перечисленных функций:

- Счетчик Up/down
- 2-х фазный счетчик Up/down
- Счетчик Up
- Счетчик Down
- Частотомер

Блок %VFC поддерживает счет на цифровых входах на частотах до 20кГц. Компактные контроллеры могут сконфигурировать один очень быстрый счетчик (%VFC), модульные контроллеры могут сконфигурировать до двух очень быстрых счетчиков (%VFC).

Присваивание выделенных вх/вых

Функциональные блоки очень быстрых счетчиков (%VFC)используют выделенные входы и вспомогательные входы и выходы. Эти входы и выходы не резервируются для их исключительного использования. Их выделение должно учитывать использование другими функциональными блоками этих выделенных ресурсов. В следующей таблице суммируется это присваивание:

		Основные входы		**Вспомогательные входы**		**Рефлексные выходы**	
%VFC 0	Выбранное использование	Вход IA	Вход IB	IPres	Ica	Выход 0	Выход 1
	Счетчик Up/down	%I0.0.1	%I0.0.0 (UP=0/DO=1)	%I0.0.2 (1)	%I0.0.3 (1)	%Q0.0.2 (1)	%Q0.0.3 (1)
	2-х фазный счетчик Up/Down	%I0.0.1	%I0.0.0 (Импульс)	%I0.0.2 (1)	%I0.0.3 (1)	%Q0.0.2 (1)	%Q0.0.3 (1)
	Счетчик Up	%I0.0.1	(2)	%I0.0.2 (1)	%I0.0.3 (1)	%Q0.0.2 (1)	%Q0.0.3 (1)
	Счетчик Down	%I0.0.1	(2)	%I0.0.2 (1)	%I0.0.3 (1)	%Q0.0.2 (1)	%Q0.0.3 (1)
	Частотомер	%I0.0.1	(2)	(2)	(2)	(2)	(2)
%VFC 1	Выбранное использование	Вход IA	Вход IB	IPres	Ica	Выход 0	Выход 1
	Счетчик Up/down	%I0.0.7	%I0.0.6 (UP = 0/DO = 1)	%I0.0.5 (1)	%I0.0.4 (1)	%Q0.0.4 (1)	%Q0.0.5 (1)
	2-х фазный счетчик Up/Down	%I0.0.7	%I0.0.6 (Импульс)	%I0.0.5 (1)	%I0.0.4 (1)	%Q0.0.4 (1)	%Q0.0.5 (1)
	Счетчик Up	%I0.0.7	(2)	%I0.0.5 (1)	%I0.0.4 (1)	%Q0.0.4 (1)	%Q0.0.5 (1)
	Счетчик Down	%I0.0.7	(2)	%I0.0.5 (1)	%I0.0.4 (1)	%Q0.0.4 (1)	%Q0.0.5 (1)
	Частотомер	%I0.0.7	(2)	(2)	(2)	(2)	(2)

Комментарии:

(1) = опционально

(2) = не используется

Ipres = вход предустановки

Ica= вход захвата

Вход IA = вход импульсов

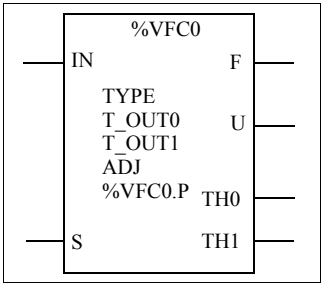
Вход IB = импульсы или UP/DO

UP/DO = счет Up / Down

Когда не используется, вход или выход остается обычным цифровым вх/вых, которым можно управлять приложением в основном цикле.

Если %I0.0.2 используется, %FC0 не допускается.
Если %I0.0.3 используется, %FC2 не допускается.
Если %I0.0.4 используется, %FC3 не допускается.

Иллюстрация Ниже приведен блок очень быстрого счетчика (%VFC):



Спецификации В следующей таблице перечислены характеристики функционального блока очень быстрого счетчика (%VFC).

Функция	**Описание**	**Значения**	**Исп. %VFC**	**Доступ во время выпол-я**
Текущее значение (%VFCi.V)	Текущее значение увеличивается или уменьшается, в соответствии с физическими входами и выбранной функцией. Это значение может быть установлено или сброшено, используя вход предустановки (%VFCi.S).	0 -> 65535	CM	Чтение
Предустановленное значение (%VFCi.P)	Используется только функцией счета up/down и счета up или down.	0 -> 65535	CM или FM	Чтение и Запись (1)
Захваченное знач. (%VFCi.C)	Используется только функцией счета up/down и счета up или down.	0 -> 65535	CM	Чтение
Направление счета (%VFCi.U)	Устанавливается системой. Этот бит используется функцией счета up/down для указания направления счета: Для счетчика up или счетчика down, %I0.0.0 указывает напрявление для %VFC0 и %I0.0.6 для %VFC1. Для 2-х фазного счетчика up/down, это разница фаз между двумя сигналами, которая указывает направление. Для %VFC0, %I0.0 выделен для IB и %I0.1 для IA. Для %VFC1, %I0.6 выделен для IB и %I0.7 для IA.	0 (Счет вниз) 1 (Счет вверх)	CM	Чтение
Разрешение рефл. выхода 0 (%VFCi.R)	Разрешение рефлексного выхода 0	0 (Запрещ.) 1 (Разреш.)	CM	Чтение и Запись (2)
Разрешение рефл. выхода 1 (%VFCi.S)	Разрешение рефлексного выхода 1	0 (Запрещ.) 1 (Разреш.)	CM	Чтение и Запись (2)

Функция	Описание	Значения	Исп. %VFC	Доступ во время выпол-я
Пороговое значение S0 (%VFCi.S0)	Это слово содержит значение порога 0. Значение определяется во время конфигурирования функционального блока. Примечание: Это значение должно быть меньше, чем %VFCi.S1.	0 -> 65535	CM	Чтение и Запись (1)
Пороговое значение S1 (%VFCi.S1)	Это слово содержит значение порога 1. Значение определяется во время конфигурирования функционального блока. Примечание: Это значение должно быть больше, чем %VFCi.S0.	0 -> 65535	CM	Чтение и Запись (1)
Масштаб измер. частоты (%VFCi.T)	Конфигурируемый элемент для 100 или 1000 мс масштаба по оси времени.	1000 или 100	FM	Чтение и Запись (1)
Регулируемый (Y/N)	Конфигурируемый элемент, который, в случае выбора, позволяет пользователю изменять уставку, пороги и масштаб измерения частоты во время работы.	N (Нет) Y (Да)	CM или FM	Нет
Вход разрешения (IN)	Используется для разрешения или запрещения текущей функции.	0 (Нет)	CM или FM	Чтение и Запись (3)
Вход предустановки (S)	В зависимости от конфигурации, в состоянии 1: ● Счет Up/Down или Down: инициализирует текущее значение предустановленным. ● Счет Up: сбрасывает текущее значение в 0. Он также инициализирует работу выходов порога и учитывает любые изменения пороговых значений, установленные дисплеем оператора или программой пользователя.	0 или 1	CM или FM	Чтение и Запись
Вых.переполн. (F)	0 до 65535 или от 65535 до 0	0 или 1	CM	Чтение
Пороговый Бит 0 (%VFCi.TH0)	Установлен в 1, когда текущее значение >= пороговому значению %VFCi.S0. Рекомендуется проверять этот бит в программе только один раз, так как он изменяется в реальном времени. Приложение пользователя отвечает за действительность значения в момент использования.	0 или 1	CM	Чтение
Пороговый Бит 1 (%VFCi.TH1)	Установлен в 1, когда текущее значение >= пороговому значению %VFCi.S1. Рекомендуется проверять этот бит в программе только один раз, так как он изменяется в реальном времени. Приложение пользователя отвечает за действительность значения в момент использования.	0 или 1	CM	Чтение

(1) Только запись, если Регулируемый = 1.
(2) Доступ возможен, только если это сконфигурировано.
(3) Доступ для чтения и записи только в приложении, но не в Дисплее оператора или Редакторе анимационных таблиц.
CM = Режим счета FM = Режим частотомера

Описание функции счета

Очень быстрые счетчики (%VFC) работают на максимальной частоте 20 КГц, с диапазоном от 0 до 65535. Подсчитываемые импульсы прикладываются следующим способом:
Таблица:

Функция	Описание	%VFC0		%VFC1	
		IA	IB	IA	IB
Счетчик Up/Down	Импульсы прикладываются к физическому входу IA, текущая операция (up/down) задается состоянием физического входа IB.	%I0.0.1	%I0.0.0	%I0.0.7	%I0.0.6
2-х фазный счетчик Up/Down	Две фазы кодера прикладываются к физическим входам IA и IB.	%I0.0.1	%I0.0.0	%I0.0.7	%I0.0.6
Счетчик Up	Импульсы прикладываются к физическому входу IA. IB не используется.	%I0.0.1	ND	%I0.0.7	ND
Счетчик Down	Импульсы прикладываются к физическому входу IA. IB не используется.	%I0.0.1	ND	%I0.0.7	ND

Замечания по функциональным блокам

Счет вверх и вниз выполняется по фронту импульса, и только если блок счетчика разрешен.
В режиме счетчика используются два опциональных входа: ICa и IPres. ICa используется для захвата текущего значения (%VFCi.V) и хранит его в %VFCi.C. Входы Ica определены как %I0.0.3 для %VFC0 и %I0.0.4 для %VFC, если доступно.
Когда вход IPres активен, текущее значение изменяется следующими способами:

- Для счета up, %VFCi.V сбрасывается в 0
- Для счета down, %VFCi.V записывается содержимым %VFCi.P
- Для частотомера, %VFCi.V устанавливается в 0

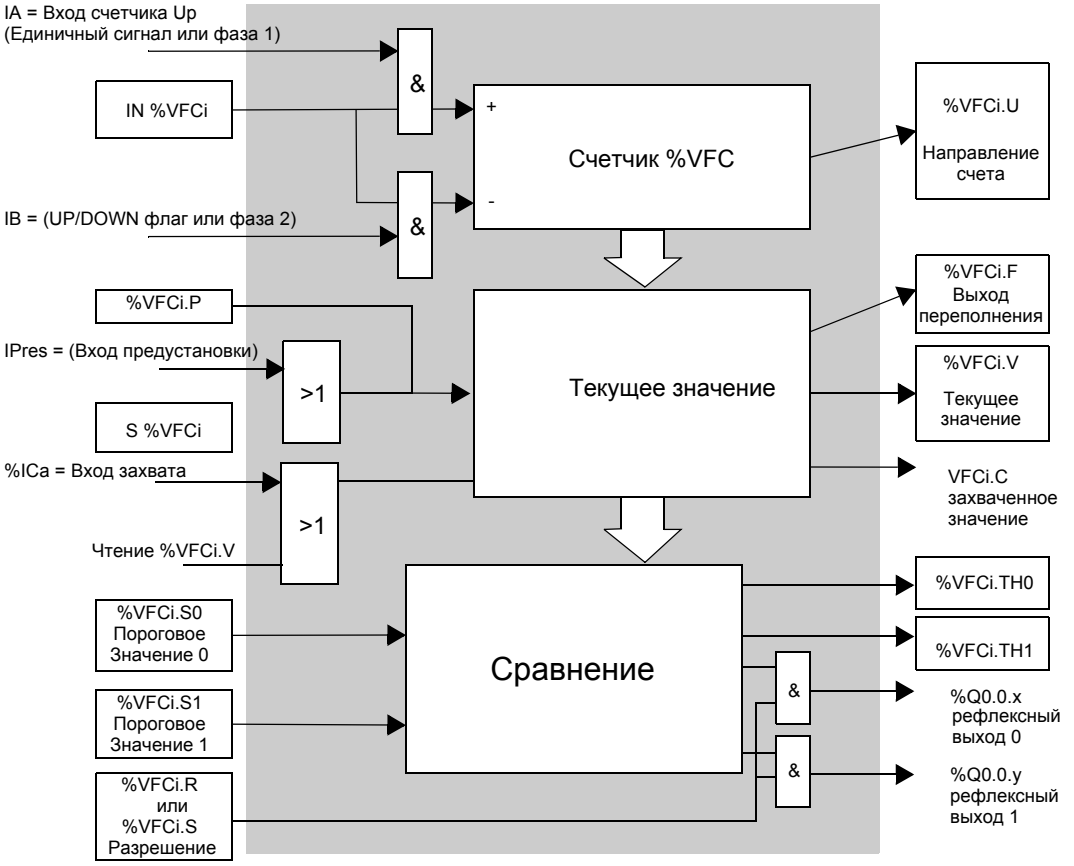
Предупреждение: %VFCi.F также устанавливается в 0. Входы IPres определены как %I0.0.2 для %VFC0 и %I0.0.5 для %VFC1, если доступно.

Замечания по выходам функциональных блоков

Для всех функций текущее значение сравнивается с двумя порогами (%VFCi.S0 и % VFCi.S1). В соответствии с результатом сравнения два битовых объекта (%VFCi.TH0 и %VFCi.TH1) устанавливаются в 1, если текущее значение больше или равно текущему пороговому, или сбрасываются в 0 в обратном случае. Рефлексные выходы (если сконфигурированы) устанавливаются в 1 в соответствии с этими сравнениями. Примечание: 0, 1 или 2 выхода могут быть сконфигурированы.
%VFC.U это выход ФБ, он указывает направление связанного счетчика (1 для UP, 0 для DOWN).

Функциональная схема счетчика

Ниже приведена функциональная схема счетчика:



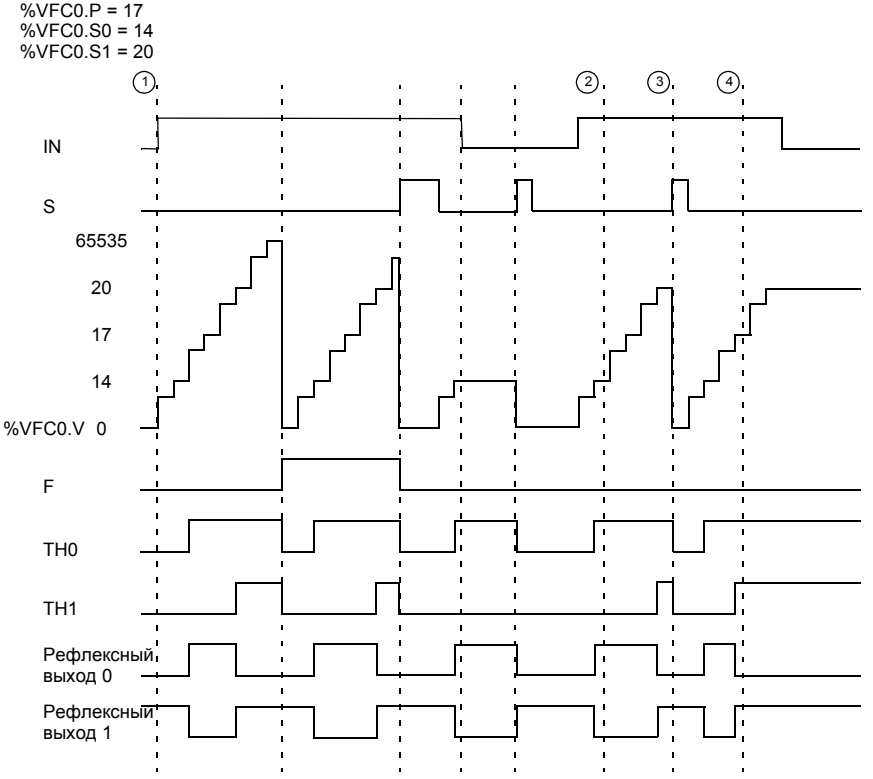
Примечание: Выходы управляются независимо от времени цикла контроллера. Время отклика от 0 до 1мс.

Функционирование счетчика Up

Ниже приведен пример использования %VFC в режиме счетчика up. Для этого примера использовалась следующая конфигурация элементов: %VFC0.P предустановленное значение =17, нижний порог %VFC0.S0 =14, верхний порог %VFC0.S1 =20.

Рефлексный выход	**<%VFC.S0**	**%VFC0.S0 <=< %VFC0.S1**	**>= %VFC0.S1**
%Q0.0.2		X	
%Q0.0.3	X		X

Временная диаграмма:



(1) : %VFC0.U = 1, потому что %VFC это счетчик up

(2) : изменение %VFC0.S1 в 17

(3) : Активный вход S указывает, что будет дано новое значение порогу S1 на следующем счете

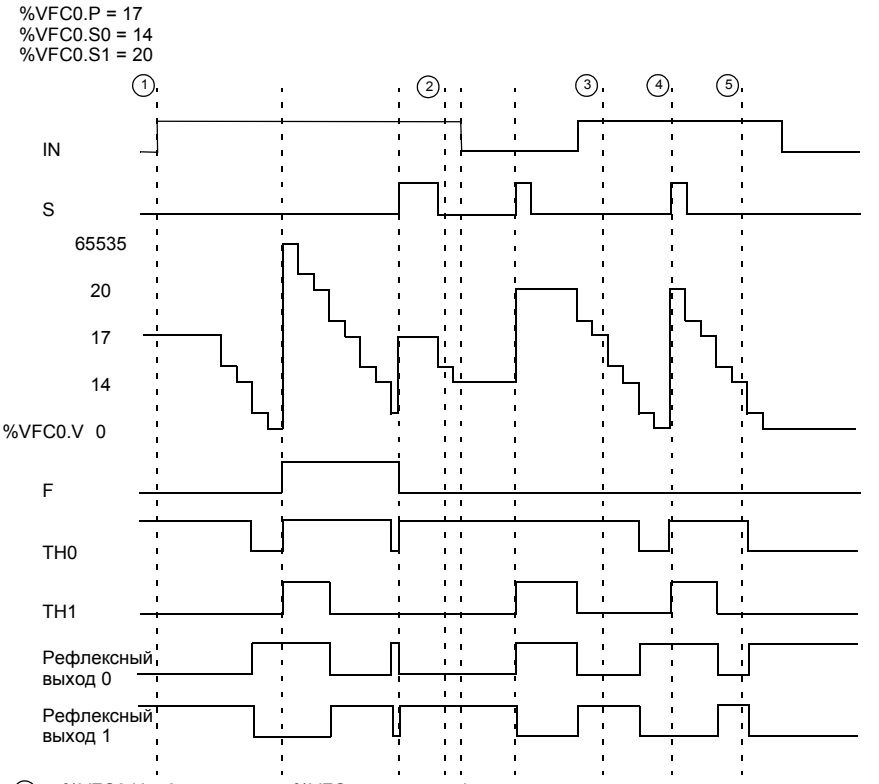
(4) : произошел захват текущего значения, поэтому %VFC0.C = 17

Функционирование счетчика Down

Ниже приведен пример использования %VFC в режиме счетчика down. Для этого примера использовалась следующая конфигурация элементов: %VFC0.P предустановленное значение =17, нижний порог %VFC0.S0 =14, верхний порог %VFC0.S1 =20.

Рефлексный выход	<%VFC.S0	%VFC0.S0 <=< %VFC0.S1	>= %VFC0.S1
%Q0.0.2	X		X
%Q0.0.3		X	

Пример:



① : %VFC0.U = 0 , потому что %VFC это счетчик down

② : изменение %VFC0.P в 20

③ : изменение %VFC0.S1 в 17

④ : Активный вход S указывает, что будет дано новое значение порогу S1 на следующем счете

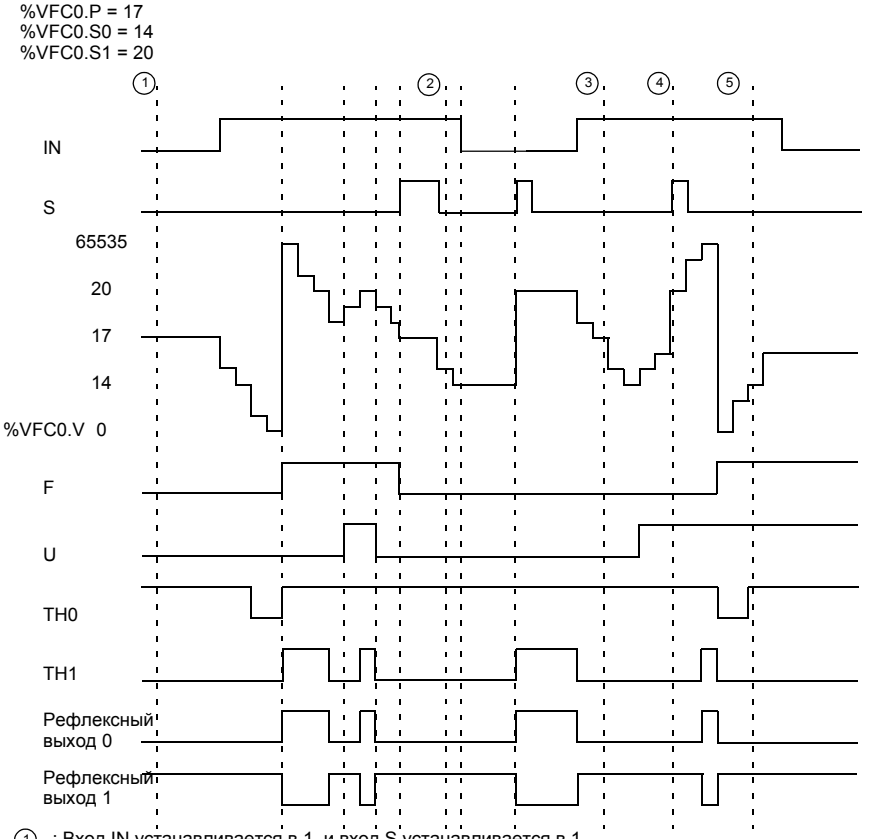
⑤ : произошел захват текущего значения, поэтому %VFC0.C = 17

Функционирование счетчика Up-Down

Ниже приведен пример использования %VFC в режиме счетчика up-down. Для этого примера использовалась следующая конфигурация элементов: %VFC0.P предустановленное значение =17, нижний порог %VFC0.S0 =14, верхний порог %VFC0.S1 =20.

Рефлексный выход	<%VFC.S0	%VFC0.S0 <=< %VFC0.S1	%VFC0.S1
%Q0.0.2			X
%Q0.0.3	X	X	

Пример:



(1) : Вход IN устанавливается в 1, и вход S устанавливается в 1

(2) : изменение %VFC0.P в 20

(3) : изменение %VFC0.S1 в 17

(4) : Активный вход S указывает, что будет дано новое значение порогу S1 на следующем счете

(5) : произошел захват текущего значения, поэтому %VFC0.C = 17

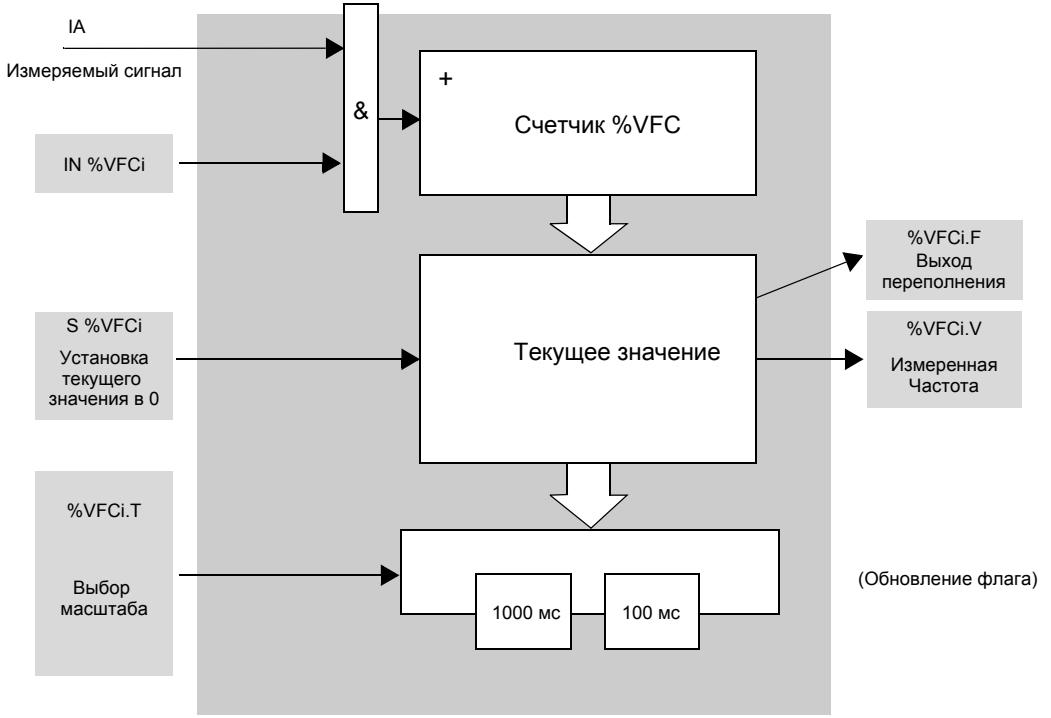
Описание функции частотомера

Функция частотомера %VFC используется для измерения частоты периодического сигнала в Гц на входе IA. Диапазон частот, которые можно измерить, от 10 до 20КГц. Пользователь может выбирать между 2 масштабами, выбор производится новым объектом %VFC.T (масштаб по оси времени). Значение 100 = масштаб 100 мс, значение 1000 = масштаб 1 с.

Масштаб	Диапазон измерения	Точность	Обновление
100 мс	100 Hz to 20 kHz	0.05 % для 20 КГц, 10 % для 100 Гц	10 раз в секунду
1 с	10 Hz to 20 kHz	0.005 % для 20 КГц, 10 % для 10 Гц	1 раз в секунду

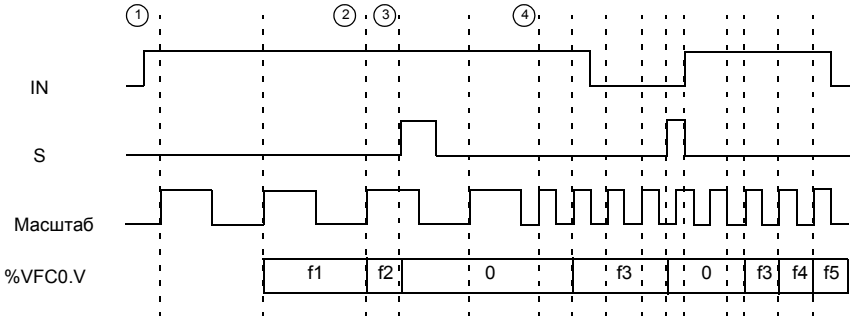
Функциональная схема частотомера

Ниже приведена функциональна схема частотомера:



Функционирование частотомера

Ниже приведена временная диаграмма примера использования %VFC в режиме частотомера.



① : Первое измерение частоты начинается здесь.

② : Обновляется текущее значение частоты.

③ : Вход IN в 1, вход S в 1

④ : Изменение %VFC0.T в 100 мс: это изменение отменяет текущее иизмерение и начинает другое

Особые случаи

В следующей таблице перечислены особые случаи функционирования функционального блока %VFC.

Особый случай	**Описание**
Влияние холодного перезапуска (%S0=1)	Сбрасывает все атрибуты %VFC в значения, сконфигурированные пользователем или пользовательским приложением.
Влияние теплого перезапуска (%S1=1)	Не влияет.
Влияние MCR/MCS	Все программные выходы не изменяются. Все сконфигурированные рефлексные выходы сбрасываются в 0.
Влияние остановки контроллера	%VFC останавливается и выходы остаются в текущем состоянии.

Передача/прием сообщений - инструкция обмена (EXCH)

Введение

Контроллер Twido может быть сконфигурирован для связи со slave устройствами Modbus или может посылать и/или получать сообщения в символьном режиме (ASCII).
TwidoSoft предоставляет следующие функции для этих коммуникаций:

- Инструкция EXCH для передачи/приема сообщений
- Функциональный блок контроля обмена (%MSG) для управления передачей данных

Контроллер Twido использует протокол, сконфигурированный для специального порта, при выполнении инструкции EXCH. Каждому коммуникационному порту может быть назначен свой протокол. Коммуникационные порты достигаются дополнением номера порта инструкции EXCH или функции %MSG (EXCH1, EXCH2, %MSG1, %MSG2).

Инструкция EXCH

Инструкция EXCH позволяет контроллеру Twido отправлять и/или принимать информацию к/от устройств ASCII. Пользователь определяет таблицу слов (%MWi:L или %KWi:L), содержащую данные для отправки и/или получения (до 250 байт данных). Формат для таблицы слов описывается в параграфах, посвященных каждому протоколу. Обмен сообщениями выполняется при помощи инструкции EXCH.

Синтаксис

Ниже приведен формат инструкции EXCH:
[EXCHx %MWi:L] или [EXCHx %KWi:L]
Где: x = номер порта (1 или 2); L = общее число слов в таблице (максимум 121).
Значения внутренней таблицы слов %MWi:L таковы, что i+L <= 255.
Контроллер Twido должен закончить обмен по первой инструкции EXCHx перед началом второй инструкции. Функциональный блок %MSG должен использоваться при отправке нескольких сообщений.

Функциональный блок контроля обмена (%MSGx)

Введение

Примечание: Символ "x" в %MSGx обозначает порт контроллера.

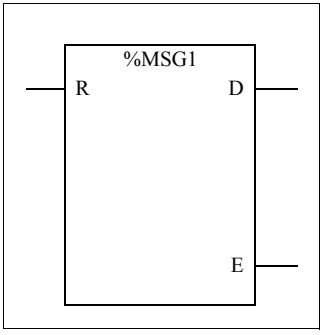
Функциональный блок %MSGx управляет обменом данными и имеет три функции:

- Проверка ошибок коммуникаций:
  Проверка ошибок контролирует, что длина блока (таблицы слов), запрограммированная инструкцией EXCH, достаточно большая для хранения длины отправляемого сообщения (сравнивается с длиной, запрограммированной в младшем значащем байте первого слова таблицы слов).
- Координирование множества сообщений:
  Для обеспечения согласованности при посылке множества сообщений, функциональный блок %MSGx обеспечивает информацию, требуемую для определения, когда завершено предыдущее сообщение.
- Передача приоритетных сообщений:
  Функциональный блок %MSGx позволяет останавливать текущую передачу сообщения, чтобы позволить немедленную отправку срочного сообщения.

Программирование функционального блока %MSGx является необязательным.

Иллюстрация

Ниже приведен пример функционального блока %MSGx.



Параметры В следующей таблице перечислены параметры для функционального блока %MSGx.

Параметр	Метка	Значение
Вход сброса (или инструкция)	R	В состоянии 1, реинициализирует коммуникацию: %MSGx.E = 0 и %MSGx.D = 1.
Выход Связь завершена	%MSGx.D	Состояние 1, связь завершена, если: ● Конец передачи (при передаче) ● Конец приема (получен символ конца) ● Ошибка ● Сброс блока Состояние 0, запрос выполняется.
Выход ошибки	%MSGx.E	Состояние 1, ошибка, если: ● Неправильная команда ● Таблица неправильно сконфигурирована ● Получен неправильный символ (скорость, четность и т.д.) ● Таблица приема полна (не обновляется) Состояние 0, длина сообщения OK, связь OK.

Если ошибка происходит во время использования инструкции EXCH, биты %MSGx.D и %MSGx.E устанавливаются в 1, и системное слово %SW63 содержит код ошибки для Порта 1, и %SW64 содержит код ошибки для Порта 2. см. *Системные слова (%SW), стр. 442*.

Вход сброса (R) Когда вход сброса установлен в 1:
- Любые передаваемые сообщения останавливаются.
- Вход ошибки сбрасывается в 0.
- Бит готовности устанавливается в 1.

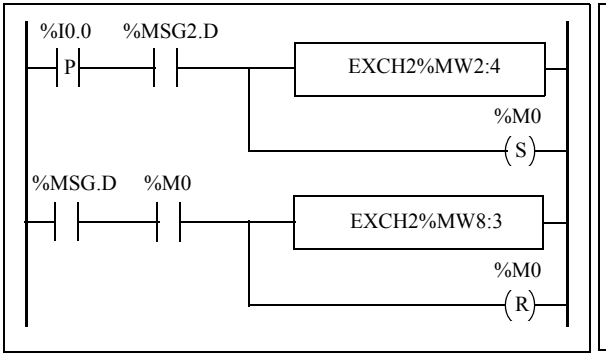
Новое сообщение не может быть отправлено.

Выход ошибки (%MSGx.E) Выход ошибки устанавливается в 1 либо из-за ошибки программирования связи, либо из-за ошибки передачи сообщения. Выход ошибки устанавливается в 1, если число байтов, определенных в блоке данных, связанном с инструкцией EXCH (слово 1, младший байт) больше 128 (+80 в 16-ричном формате).
Выход ошибки также устанавливается в 1, если существует проблема отсылки сообщения Modbus к устройству Modbus. В этом случае пользователь должен проверить соединение и то, что устройство назначения поддерживает связь Modbus.

Выход Связь завершена (%MSGx.D) Когда выход готовности установлен в 1, контроллер Twido готов к отправке другого сообщения. Использование бита %MSGx.D рекомендуется, когда посылается множество сообщений. Иначе сообщения могут быть утеряны.

Передача нескольких последовательных сообщений

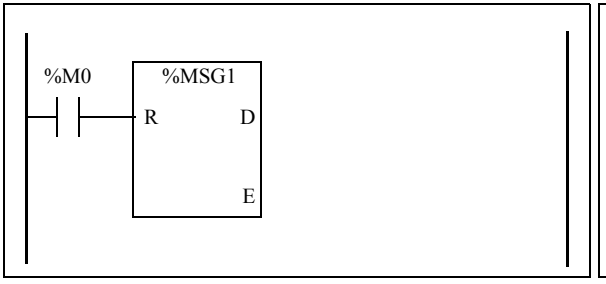
Выполнение инструкции EXCH активирует блок сообщения в прикладной программе. Сообщение передается, если блок сообщений ещё не активен (%MSGx.D = 1). Если несколько сообщений посылаются в одном цикле, передается только первое сообщение. Пользователь отвечает за управление отправкой нескольких сообщений в программе.
Пример передачи двух последовательных сообщений через Порт 2:




```
LDR     %I0.0
AND     %MSG2.D
[EXCH2 %MW2:4]
S       %M0
LD      %MSG2.D
AND     %M0
[EXCH2 %MW8:3]
R       %M0
```

Реинициализация обмена

Обмен отменяется активированием входа (или инструкции) R. Этот вход инициализирует связь и сбрасывает выход %MSGx.E в 0 и выход %MSGx.D в 1. Возможно реинициализировать обмен, если обнаружена ошибка.
Пример реинициализации обмена:




```
BLK     %MSG1
LD      %M0
R
END_BLK
```

Особые случаи

В следующей таблице перечислены особые случаи для ФБ %MSGx.

Особый случай	Описание
Влияние холодного перезапуска (%S0=1)	Приводит к реинициализации связи.
Влияние теплого перезапуска (%S1=1)	Не влияет.
Влияние остановки контроллера	Если происходит передача сообщения, контроллер останавливет пересылку и реинициализирует выходы %MSGx.D и %MSGx.E.

15.2 Функции часов

Обзор

Цель этого раздела

В этом разделе описаны функции управления временем для контроллеров Twido.

Содержание раздела

Этот раздел содержит следующие темы:

Функции часов

Введение

Контроллеры Twido имеют функцию часов, которая требует опции часов реального времени (RTC) и предоставляет следующее:

- **Блоки планировщика** используются для выполнения действий в предопределенное или рассчитанное время.
- **Печать времени/даты** используется для присваивания времени и даты событиям и измерения продолжительности событий.

К часам реального времени можно получить доступ, выбрав **Schedule Blocks** из меню TwidoSoft **Software**. Дополнительно, часы могут быть установлены из программы. Установки часов сохраняются до 30 дней после выключения контроллера, если батарея заряжалась как минимум шесть часов подряд перед отключением контроллера.

Часы реального времени имеют 24-часовой формат и учитывают високосные годы.

Корректирующее значение RTC

Корректирующее значение RTC необходимо для правильной работы RTC. Каждый блок RTC имеет свое корректирующее значение, записанное в блоке. Это значение может конфигурироваться из TwidoSoft, используя опцию **Configure RTC** из диалогового окна **Controller Operations**.

Блоки планировщика

Введение

Блоки планировщика используются для управления событиями в предопределенные месяц, день и время. Максимум 16 блоков планировщика могут использоваться и не требовать запуска программы.

Примечание: Проверьте системный бит %S51 и системное слово%SW118, чтобы подтвердить, что опция RTC установлена. См. *Системные биты (%S), стр. 434*. Опция RTC требуется для использования блоков планировщика.

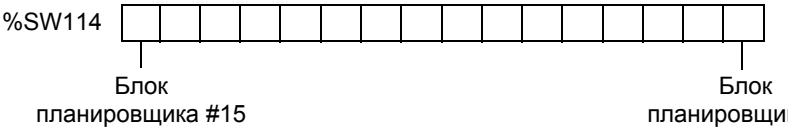
Параметры

В следующей таблице перечислены параметры блока планировщика:

Параметр	Формат	Функция/диапазон
Schedule block number (номер блока планировщика)	n	n = 0 до 15
Configured (сконфигурирован)	Флажок	Поставьте флажок для конфигурирования выбранного блока планировщика.
Output bit (выходной бит)	%Qx.y.z	Присваивание выходам активируется блоком планировщика: %Mi или %Qj.k. Этот выход устанавливается в 1, когда текущие дата и время находятся между началом и концом активного периода.
Start month (месяц начала)	January - December	Месяц для начала блока планировщика.
End month (месяц завершения)	January - December	Месяц для завершения блока планировщика.
Start date (дата начала)	1 - 31	День для месяца начала блока планировщика.
End date (дата завершения)	1 - 31	День для месяца завершения блока планировщика.
Start time (время начала)	hh:mm	Время дня, часы (0 до 23) и минуты (0 до 59), для начала блока планировщика.
Stop time (время завершения)	hh:mm	Время дня, часы (0 до 23) и минуты (0 до 59), для завершения блока планировщика.
Day of week (день недели)	Monday - Sunday	Флажки, указывающие на день недели для активации блока планировщика.

Разрешение блоков планировщика

Биты системного слова %SW114 позволяют (бит установлен в 1) или запрещают (бит установлен в 0) работу каждого из 16 блоков планировщика. Присваивание блоков планировщика битам %SW114:



По умолчанию (или после “холодного” перезапуска) все биты этого системного слова установлены в 1. Использование этих битов в программе не обязательно.

Выход блоков планировщика

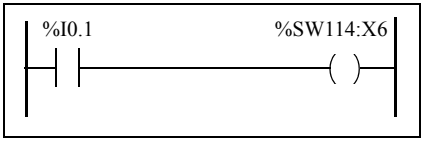
Если один выход(%Mi или %Qj.k) присвоен нескольким блокам, он вычисляется как функция ИЛИ результатов каждого блока, которая окончательно присваивается объекту (возможно иметь несколько "рабочих диапазонов" для одного выхода).

Пример

В следующей таблице показаны параметры для примера программы функции, работающей в летние месяцы:

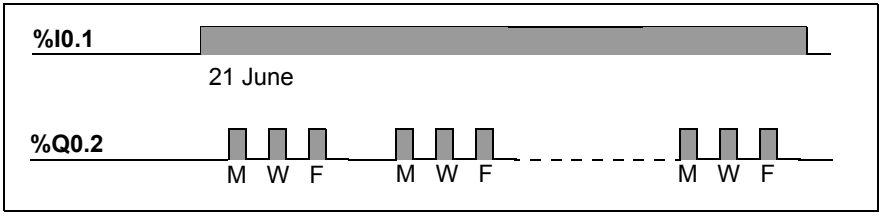
Параметр	Значение	Описание
Schedule block	6	Блок планировщика номер 6
Output bit	%Q0.2	Активировать выход %Q0.2
Start month	June	Начать работу в июне
End month	September	Завершить работу в сентябре
Start date	21	Начать работу 21-го июня
End date	21	Завершить работу 21-го сентября
Day of week	Monday, Wednesday, Friday	Выполнять работу по понедельникам, средам и пятницам
Start time	21:00	Начать работу в 21:00
Stop time	22:00	Завершить работу в 22:00

Используя следующую программу, блок планировщика может быть запрещен при помощи переключателя или детектора влажности, соединенного со входом %I0.1.




```
LD    %I0.1
ST    %SW114:X6
```

Следующая временная диаграмма показывает активацию выхода %Q0.2.



Датирование программой

Дата и время доступны в системных словах с %SW50 по %SW53 (см. *Системные слова (%SW), стр. 442*). Поэтому возможно выполнять печать времени и даты в программе контроллера, выполняя арифметическое сравнение текущей даты и времени и прямых значений слов %MWi (или %KWi), которые содержат уставки.

Печать времени/даты

Введение

Системные слова с %SW49 по %SW53 содержат текущую дату и время в формате BCD (См. *Обзор кода BCD, стр. 314*), который бесполезен для отображения или передачи на периферийное устройство. Эти системные слова могут использоваться для хранения времени и даты событий (См. *Системные слова (%SW), стр. 442*).

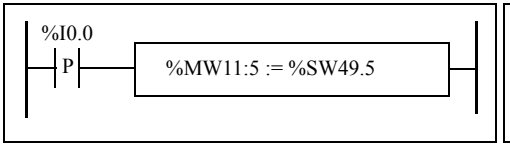
Примечание: дата и время также могут быть установлены через опциональный дисплей оператора (См. *Часы, стр. 205*).

Датирование события

Для датирования события достаточно использовать операции присваивания, переслать содержимое системных слов во внутренние слова, и обработать эти внутренние слова (например, переслать на дисплей инструкцией EXCH).

Пример программирования

В следующем примере показано, как датировать фронт входа %I0.1.




```
LDR      %I0.0
[%MW11:5 := %SW49:5]
```

После определения события таблица слов содержит:

Кодирование	Старший значащий байт	Младший значащий байт
%MW11		День недели[1]
%MW12	00	Секунда
%MW13	Час	Минута
%MW14	Месяц	День
%MW15	Век	Год

Примечание: (1) 1 = Понедельник, 2 = Вторник, 3 = Среда, 4 = Четверг, 5 = Пятница, 6 = Суббота, 7 = Воскресенье.

Пример таблицы слов

Пример для даты: 13:40:30 Понедельник, 19 Апреля, 2002:

Слово	Значение (16-ричное)	Содержание
%MW11	0001	Понедельник
%MW12	0030	30 секунд
%MW13	1340	13 часов, 40 минут
%MW14	0419	04 = Апрель, 19-е
%MW15	2002	2002

Дата и время последней остановки

Системные слова с %SW54 по %SW57 содержат дату и время последней остановки, слово %SW58 содержит код, показывающий причину последней остановки в формате BCD (см. *Системные слова (%SW), стр. 442*).

Установка даты и времени

Введение

Вы можете обновить установки даты и времени одним из следующих методов:

- TwidoSoft
  Используйте диалоговое окно **Set Time**. Это диалоговое окно доступно из диалогового окна **Controller Operations**. Оно отображается после выбора **Controller Operations** из меню **Controller**.
- Системные слова
  Используйте системные слова с %SW49 по %SW53 или системное слово %SW59.

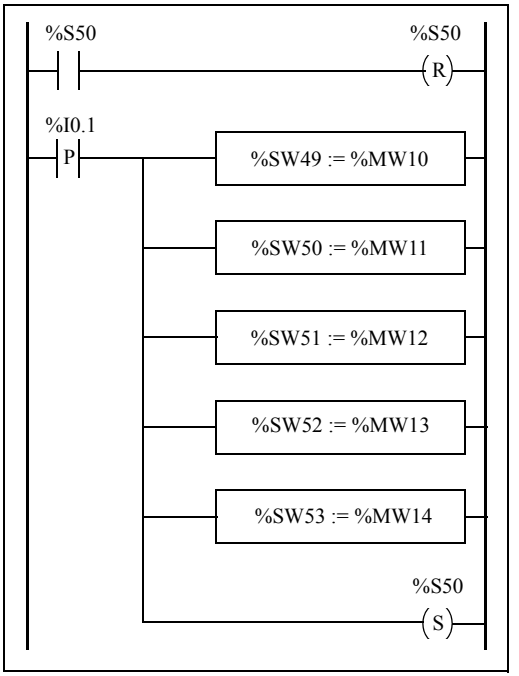
Установки даты и времени могут быть обновлены только, когда опциональный картридж RTC (TWDXCPRTC) установлен в контроллере.

Использование %SW49 по %SW53

Для использования системных слов с %SW49 по %SW53для установки даты и времени, бит %S50 должен быть установлен в 1. Это имеет следующие последствия:

- Отмена обновления слов с %SW49 по %SW53 через внутренние часы.
- Передача значений, записанных во внутренние слова с %SW49 по %SW53, во внутренние часы.

Пример программирования:




```
LD      %S50
R       %S50

LDR     %I0.1
[%SW49 := %MW10]
[%SW50 := %MW11]
[%SW51 := %MW12]
[%SW52 := %MW13]
[%SW53 := %MW14]
S       %S50
```

Слова с %MW10 по %MW14 будут содержать новую дату и время в формате BCD (см. *Обзор кода BCD, стр. 314*) и будут соответствовать кодам слов с %SW49 по %SW53.

Таблица слов должна содержать новую дату и время:

Кодирование	Старший значащий байт	Младший значащий байт
%MW10		День недели[1]
%MW11	00	Секунда
%MW12	Час	Минута
%MW13	Месяц	День
%MW14	Век	Год

Примечание: (1) 1 = Понедельник, 2 = Вторник, 3 = Среда, 4 = Четверг, 5 = Пятница, 6 = Суббота, 7 = Воскресенье.

Пример для даты: Понедельник, 19 Апреля, 2002:

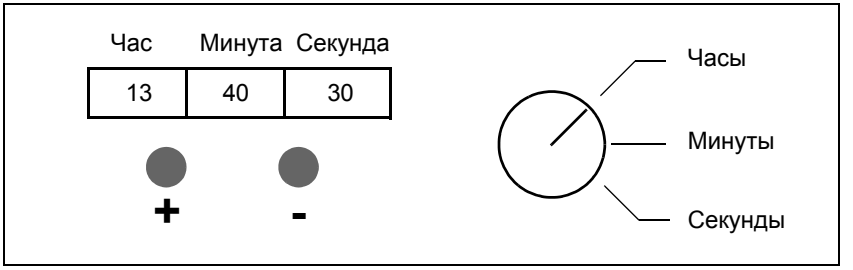
Слово	Значение (16-ричное)	Содержание
%MW10	0001	Понедельник
%MW11	0030	30 секунд
%MW12	1340	13 часов, 40 минут
%MW13	0419	04 = Апрель, 19-е
%MW14	2002	2002

Использование %SW59

Другой метод обновления даты и времени состоит в использовании системного бита %S59 и системного слова установки даты %SW59. Установка бита %S59 в 1 позволяет устанавливать текущую дату и время при помощи слова %SW59 (см. *Системные слова (%SW), стр. 442*). %SW59 увеличивает или уменьшает каждый компонент даты или времени по фронту сигнала.

Пример приложения

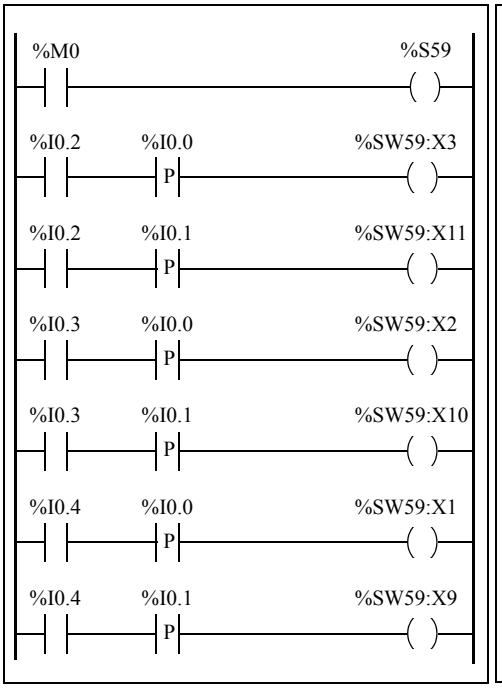
Следующая передняя панель создана для изменения значений часов, минут и секунд внутренних часов.



Описание команд:

- Переключатель Часы/Минуты/Секунды выбирает, какой дисплей времени изменять, используя входы %I0.2, %I0.3 и %I0.4 соответственно.
- Кнопка "+" увеличивает выбранный дисплей, используя вход %I0.0.
- Кнопка "-" уменьшает выбранный дисплей, используя вход %I0.1.

Следующая программа читает входы с панели и устанавливает внутренние часы.




```
LD    %M0
ST    %S59
LD    %I0.2          (Час)
ANDR  %I0.0
ST    %SW59:X3
LD    %I0.2
ANDR  %I0.1
ST    %SW59:X11
LD    %I0.3          (Минута)
ANDR  %I0.0
ST    %SW59:X2
LD    %I0.3
ANDR  %I0.1
ST    %SW59:X10
LD    %I0.4          (Секунда)
ANDR  %I0.0
ST    %SW59:X1
LD    %I0.4
ANDR  %I0.1
ST    %SW59:X9
```

15.3 Функция PID

Обзор

Цель этого раздела

В этом разделе описано поведение, функциональные возможности и применение функции PID.

Содержание раздела

Этот раздел содержит следующие темы:

Общее представление

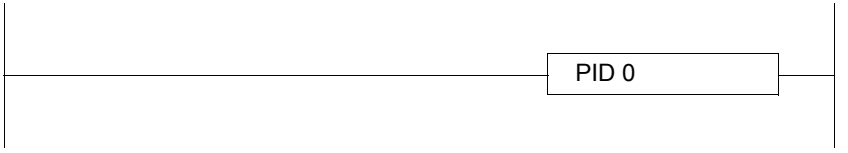
Общие положения

Регулирующая функция PID -это функция языка программирования TwidoSoft. Она позволяет программировать регулирующие циклы PID на контроллерах Twido версии 1.2 или выше.

Эта функция особенно:
- Соответствует потребностям последовательной обработки, для которой требуются вспомогательные функции регулировки (примеры: машина для упаковки пластиковой пленки, машина для окончательной обработки, прессы и т.д.)
- Соответствует потребностям простых процессов регулировки (примеры: печи для металла, печи для керамики, небольшие рефрижераторы и т.д.)

Очень легко устанавливать, так как это выполняется в экранах:
- Configuration
- и Debug,

связанных со строками программы (операционный блок в языке лестничной логики или простой вызов инструкции PID в языке списка инструкций), указывающими номер используемых PID.
Пример строки программы на языке лестничной логики:



Примечание: В любом приложении автоматизации Twido максимальное число конфигурируемых функций PID 14.

Ключевые возможности

Ключевые возможности следующие:
- Аналоговый вход,
- Линейное преобразование конфигурируемого измерения,
- Конфигурируемый сигнал тревоги по высокому или низкому уровню,
- Аналоговый выход или выход широтно-импульсного модулятора,
- Отключение конфигурируемого выхода,
- Конфигурируемое прямое или инверсное действие.

Принцип регулирующего цикла

Обзор

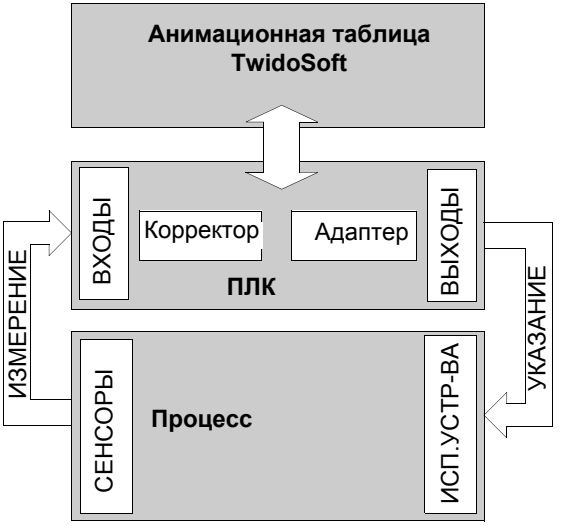
Работа регулирующего цикла состоит из трех различных фаз:
- Сбор данных:
  - Измерения сенсоров (аналоговых, кодировщиков)
  - Уставки из внутренних переменных контроллера или из данных анимационной таблицы TwidoSoft
- Выполнение регулирующего алгоритма PID
- Отсылка указаний, адаптированных к характеристикам исполнительных механизмов, передаваемых через дискретные (PWM) или аналоговые выходы

Алгоритм PID генерирует управляющий сигнал из:
- Выборочных измерений входного модуля
- Установочных значений, зафиксированных оператором или программой
- Значениями различных корректирующих параметров

Сигнал от корректирующего устройства либо прямо обрабатывается платой аналогового выхода контроллера, связанного с исполнительным устройством, либо обрабатывается через контролирующую PWM плату на дискретном выходе контролера.

Иллюстрация

На следующем рисунке схематично изображен принцип регулирующего цикла.



Развитие методологии регулирующего приложения

Принцип

Следующая диаграмма описывает все задачи, выполняемые во время создания и отладки регулирующего приложения.

Примечание: Порядок зависит от Ваших методов работы и представлен, как пример.

Совместимость и производительность

Обзор

Функция PID доступна для Twido версии 1.2 и выше, поэтому ее установка подчиняется правилам программной и аппаратной совместимости, описанным в следующих параграфах.
Кроме того, эта функция требует наличия ресурсов, представленных в параграфе **Производительность**.

Совместимость

Функция PID доступна для Twido с ПО версии 1.2 или выше.
Если Ваши Twido имеют более раннюю версию ПО, Вы можете обновить аппаратно-программные средства для использования функции PID.

Примечание: Модули аналоговых вх/вых версии 1.0 могут использоваться как модули вх/вых PID без необходимости обновления.

Для конфигурирования и программирования PID для этих аппаратных версий, Вы должны иметь **ПО TwidoSoft версии 1.2.**

Производительность

Регулирующие циклы PID имеют следующую производительность:

Описание	Время
Время выполнения цикла	0.4 мс

Подробные характеристики функции PID

Общие положения

Функция PID выполняет коррекцию PID через аналоговые измерения и установочные значения в формате по умолчанию [0-10000] и обеспечивает аналоговую команду в том же формате или генератор широтной модуляции (PWM) на цифровом выходе.
Все параметры PID поясняются в окнах, используемых для их конфигурации . Здесь мы просто резюмируем доступные функции, укажем значения измерений и опишем, как они интегрируются в функциональную диаграмму процесса PID.

Примечание: Для использования на полной шкале (оптимальное разрешение), Вы можете сконфигурировать соединение аналогового входа с секцией измерений PIDв формате 0-10000. Однако, если Вы используете конфигурацию по умолчанию (0-4095), контроллер будет функционировать правильно.

Примечание: Для того, чтобы регулирование выполнялось правильно, необходимо, чтобы ПЛК Twido **находился в периодическом режиме**. Функция PID выполняется периодически в каждом цикле, и выборки входных данных PID соответствуют периоду, установленному при конфигурации (см. таблицу ниже).

Подробное описание доступных функций

В следующей таблице указаны различные доступные функции и их шкалы:

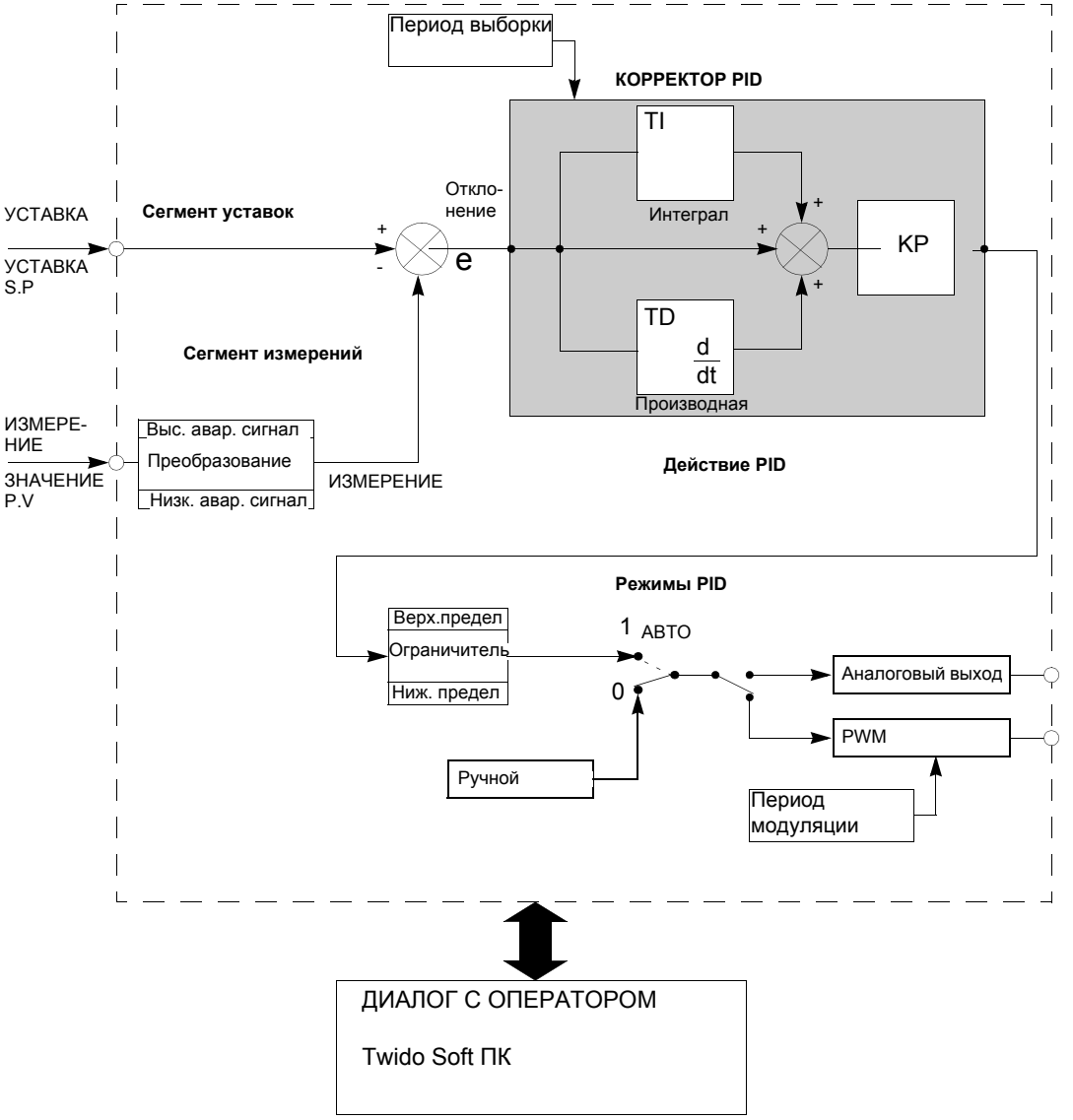
Функция	Шкала и комментарии
Линейное преобразование входов	Позволяет преобразовывать значение в формате от 0 до 10000 (разрешение модуля аналогового входа) в значение между -32768 и 32767
Пропорциональное увеличение	Используя множитель 100, значение между 1 и 10000. Это соответствует увеличенному числу от 0.01 до 100.
Интеграл времени	Используя масштаб 0.1 секунды, значение между 0 и 32767. Это соответствует интегралу времени от 0 до 3276.7 секунд.
Производная времени	Используя масштаб 0.1 секунды, значение между 0 и 32767. Это соответствует производной времени от 0 до 3276.7 секунд.

Функция	Шкала и комментарии
Период выборки	Используя масштаб 0.01 секунды, значение между 1 и 10000. Это соответствует периоду выборки от 0.01 до 100 секунд.
Выход PWM	Используя масштаб 0.1 секунды, значение между 1 и 500. Это соответствует периоду модуляции от 0.1 до 50 секунд.
Аналоговый выход	Значение между 0 и +10,000
Аварийный сигнал высокого уровня для параметра процесса	Этот сигнал устанавливается после преобразования. Он устанавливается в значение от -32768 до 32767, если преобразование активировано, и от 0 до 10000, если не активировано.
Аварийный сигнал низкого уровня для параметра процесса	Этот сигнал устанавливается после преобразования. Он устанавливается в значение от -32768 до 32767, если преобразование активировано, и от 0 до 10000, если не активировано.
Значение верхней границы выхода	Эта граница между 0 и 10000 для значения аналогового выхода. Когда PWM активен, граница соответствует процентной части периода модуляции. 0% для 0 и 100% для 10000.
Значение нижней границы выхода	Эта граница между 0 и 10000 для значения аналогового выхода. Когда PWM активен, граница соответствует процентной части периода модуляции. 0% для 0 и 100% для 10000.
Ручной режим	Когда активен ручной режим, выходу присваивается фиксированное значение, задаваемое пользователем. Эти выходное значение находится между 0 и 10000 (от 0 до 100% для выхода PWM).
Прямое или обратное действие	Прямое или обратное действие доступно и влияет непосредственно на выход.

Примечание: Для более подробного объяснения того, как работает каждая описанная выше функция, обратитесь к диаграмме ниже.

Принципы работы

На следующем рисунке показаны принципы работы функции PID.



Примечание: Использованные параметры описаны в предыдущей таблице и в экранах конфигурации.

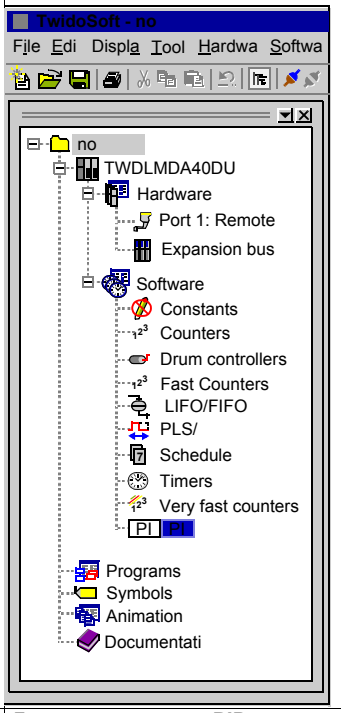
Как получить доступ к конфигурации PID

Обзор

В следующих параграфах описано, как получить доступ к экранам конфигурации PID для контроллеров TWIDO.

Процедура

В следующей таблице описана процедура доступа к экранам конфигурации PID:

Шаг	Действие
1	Убедитесь, что Вы находитесь в режиме **offline**.
2	Откройте браузер. 
3	Двойной щелчок на **PID**. **Результат:** Откроется окно конфигурации PID и по умолчанию отобразится вкладка **General** (См. *Вкладка General функции PID, стр. 387*). **Примечание**: Вы можете также щелкнуть правой кнопкой на **PID** и выбрать опцию **Edit,** или выбрать **Software** Ж **PID** из меню, или использовать **Program** Ж **Configuration Editor** Ж **PID Icon** меню, или, при использовании последнего метода, выбрать PID и щелкнуть на иконке Увеличительного стекла и выбрать особый PID.

Вкладка General функции PID

Обзор

Когда Вы открываете PID из браузера, Вы открываете окно конфигурации PID. Это окно позволяет Вам:

- конфигурировать каждый TWIDO PID,
- отлаживать каждый TWIDO PID,

Когда Вы открываете этот экран, если Вы:

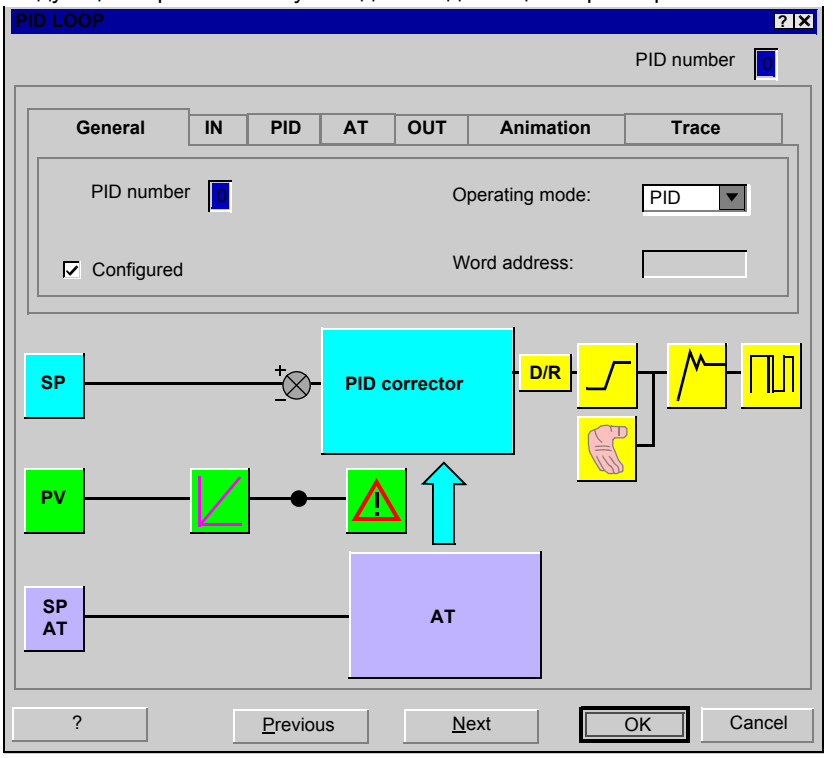
- в режиме offline: Вы попадете во вкладку **General** по умолчанию и получите доступ к параметрам конфигурации,
- в режиме online: Вы попадете во вкладку **Animation** и получите доступ к параметрам отладки и регулировки.

Примечание: Вкладки и поля, выделенные серым, недоступны, либо потому что они будут представлены в следующей версии (например, автонастройка), либо потому что текущий активный режим(offline или online) не позволяет получить доступ к этим параметрам.

В следующих параграфах описана вкладка **General**.

Вкладка General функции PID

Следующий экран используется для ввода общих параметров PID.



Описание

В следующей таблице описаны настройки, которые Вы можете задать.

Поле	Описание
PID number (**Номер PID**)	Укажите номер PID, который Вы хотите здесь сконфигурировать. Значение между 0 и 13, максимум 14 PID на приложение.
Configured (Сконфигурирован)	Для конфигурирования PID этот флажок должен быть поставлен. Иначе нельзя будет выполнять никакие действия на этих экранах и PID, хотя он и существует в приложении, не может быть использован.
Operating mode (Режим работы)	Укажите желаемый режим работы. В этой версии Twido, единственная доступная опция это PID. Другие опции будут доступны в следующих версиях.
Word address	Не доступно в этой версии, так как относится к автонастройке.
Diagram (Диаграмма)	Диаграмма позволяет Вам посмотреть различные возможности, доступные для конфигурирования PID.

Вкладка IN функции PID

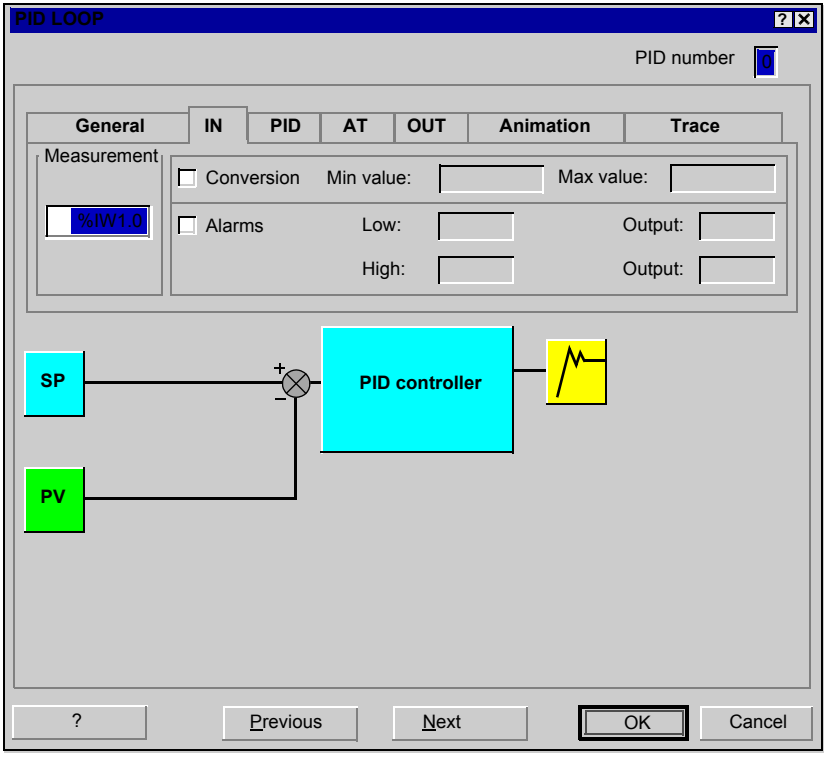
Обзор

Вкладка используется для ввода входных параметров PID.

Примечание: Вкладка доступна в режиме offline.

Вкладка IN функции PID

Следующий экран используется для ввода входных параметров PID.



Описание В следующей таблице описаны настройки, которые Вы можете задать.

Поле	Описание
PID number (Номер **PID)**	Укажите номер PID, который Вы хотите здесь сконфигурировать. Значение между 0 и 13, максимум 14 PID на приложение.
Measurement (Измерение)	Укажите переменную, которая будет содержать контролируемый параметр процесса. шкала по умолчанию от 0 до 10000. Вы можете ввести либо внутреннее слово (%MW0 до %MW2999), либо аналоговый вход (%IWx.0 до %IWx.1).
Conversion (Преобразова-ние)	Поставьте этот флажок, если Вы хотите преобразовать параметр процесса, заданный как вход PID. Если этот флажок поставлен, оба поля **Min value** и **Max value** доступны. Преобразование является линейным. Преобразовываются значения от 0 до 10,000 в значения, для которых минимумы и максимумы находятся между -32768 и +32767.
Min value (Минимальное значение) **Max value** (Максимальное значение)	Укажите минимальные и максимальные допустимые входные значения PID . Параметр процесса затем автоматически преобразовывается PID так, чтобы он не выходил за пределы минимального и максимального значений. **Примечание**: минимальное значение должно всегда быть меньше максимального значения. **Минимальное и максимальное значения могут быть внутренними словами** (от %MW0 до %MW2999), внутренними константами (от %KW0 до %KW255) или значениями между -32768 и +32767.
Alarms (Аварийные сигналы)	Поставьте этот флажок, если Вы хотите активировать аварийные сигналы по высоким или низким входным значениям. **Примечание**: Аварийные сигналы должны быть определены в соответствии с параметром процесса, полученным после фазы преобразования. Они должны быть между минимальным и максимальным значениями, когда преобразование активно. Иначе они будут между 0 и 10000.
Low (Нижнее значение) **Output** (Выход)	Укажите значение аварийного сигнала низкого уровня. Это значение может быть внутренним словом (от %MW0 до %MW2999), внутренней константой (от %KW0 до %KW255) или прямым значением. **Выход должен содержать адрес бита, который будет установлен в** 1, когда будет достигнута нижняя граница. Выход может быть либо внутренним битом (от %M0 до %M255) или выходом (от %Qx.0 до %Qx.32).
High (Верхнее значение) **Output** (Выход)	Укажите значение аварийного сигнала высокого уровня. Это значение может быть внутренним словом (от %MW0 до %MW2999), внутренней константой (от %KW0 до %KW255) или прямым значением. **Выход должен содержать адрес бита, который будет установлен в** 1, когда будет достигнута нижняя граница. Выход может быть либо внутренним битом (от %M0 до %M255) или выходом (от %Qx.0 до %Qx.32).
Diagram (Диаграмма)	Диаграмма позволяет Вам посмотреть различные возможности, доступные для конфигурирования PID.

Вкладка PID функции PID

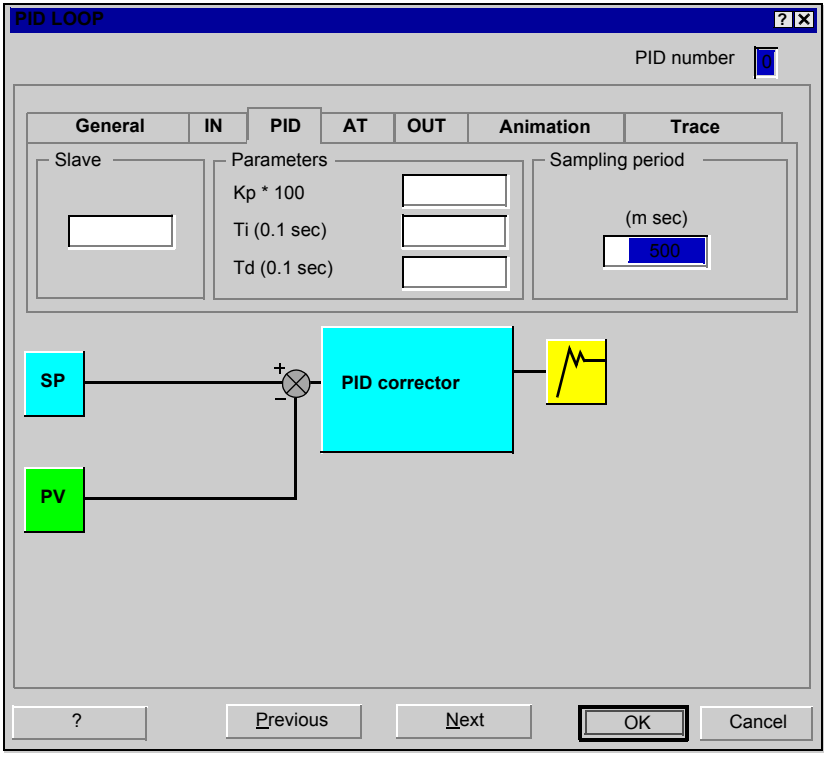
Обзор

Вкладка используется для ввода внутренних параметров PID.

Примечание: Вкладка доступна в режиме offline.

Вкладка PID функции PID

Следующий экран используется для ввода внутренних параметров PID.



Описание

В следующей таблице описаны настройки, которые Вы можете задать.

Поле	Описание
PID number (Номер PID)	Укажите номер PID, который Вы хотите здесь сконфигурировать. Значение между 0 и 13, максимум 14 PID на приложение.
Slave	Укажите значение уставки PID. Это значение может быть внутренним словом (от %MW0 до %MW2999), внутренней константой (от %KW0 до %KW255) или прямым значением. Это значение должно быть между 0 и 10000, когда преобразование запрещено. Иначе оно должно быть между минимальным и максимальным значениями для преобразования.
Kp * 100	Укажите пропорциональный коэффициент PID умноженный на 100. Это значение может быть внутренним словом (от %MW0 до %MW2999), внутренней константой (от %KW0 до %KW255) или прямым значением. Оно должно быть между 0 и 127.
TI (0.1 sec)	Укажите коэффициент воздействия по интегралу для масштаба по оси времени 0.1 секунды. Это значение может быть внутренним словом (от %MW0 до %MW2999), внутренней константой (от %KW0 до %KW255) или прямым значением. Оно должно быть между 0 и 32,767.
Td (0.1 sec)	Укажите коэффициент воздействия по производной для масштаба по оси времени 0.1 секунды. Это значение может быть внутренним словом (от %MW0 до %MW2999), внутренней константой (от %KW0 до %KW255) или прямым значением. Оно должно быть между 0 и 32,767.
Sampling period (Период выборки)	Укажите период выборки PID для масштаба по оси времени 10^{-2} секунд. Это значение может быть внутренним словом (от %MW0 до %MW2999), внутренней константой (от %KW0 до %KW255) или прямым значением. Оно должно быть между 0 и 127.
Diagram (Диаграмма)	Диаграмма позволяет Вам посмотреть различные возможности, доступные для конфигурирования PID.

Вкладка OUT функции PID

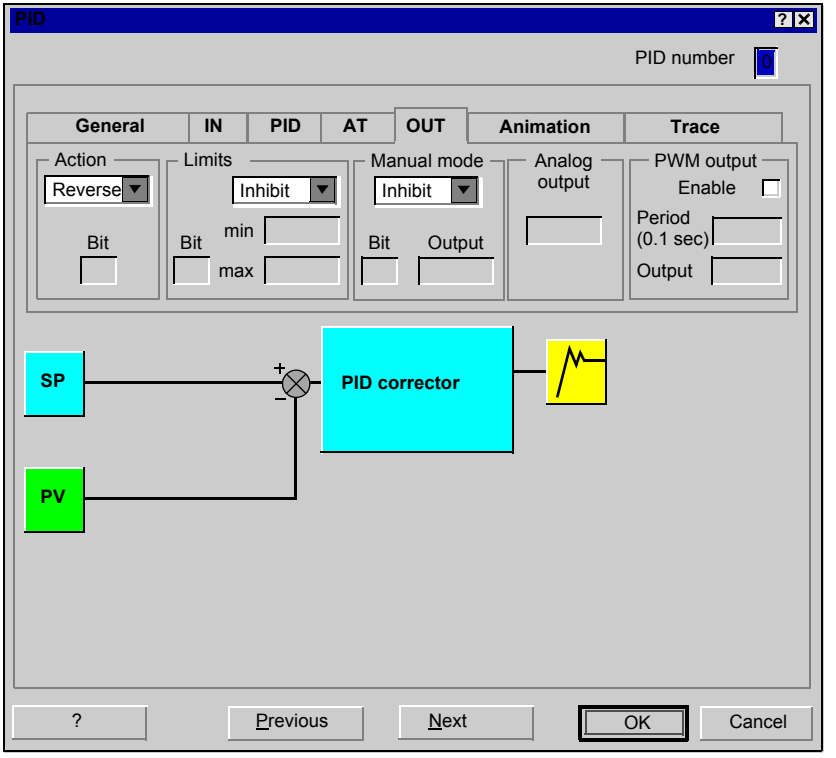
Обзор

Вкладка используется для ввода выходных параметров PID.

Примечание: Вкладка доступна в режиме offline.

Вкладка OUT функции PID

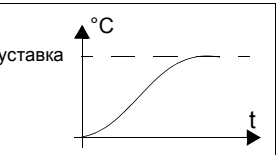
Следующий экран используется для ввода выходных параметров PID.



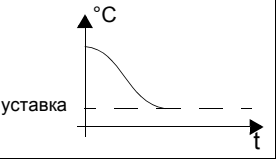
Описание В следующей таблице описаны настройки, которые Вы можете задать.

Поле	Описание
PID number (Номер PID)	Укажите номер PID, который Вы хотите здесь сконфигурировать. Значение между 0 и 13, максимум 14 PID на приложение.
Action (Действие)	Укажите тип действия PID. Доступны три опции: **Обратный**, **Прямой** или адрес бита. Если Вы выбрали адрес бита, Вы можете изменить этот тип через программу, изменяя связанный бит, который является либо внутренним битом (от %M0 до %M255) либо входом (от %Ix.0 до %Ix.32). Действие прямое, если бит установлен в 1, и обратное, если нет.
Limits (Пределы) **Bit** (Бит)	Укажите, хотите ли Вы поставить ограничения на выход PID. Доступны три опции: **Разрешено**, **Запрещено** или адрес бита. Если Вы выбрали адрес бита, Вы можете разрешить (бит = 1) или запретить (бит = 0) управление пределами в программе, изменяя связанный бит, который является либо внутренним битом (от %M0 до %M255) либо входом (от %Ix.0 до %Ix.32).
Min. (Минимум) **Max.** (Максимум)	Укажите пороговые значения выхода PID. **Примечание**: Минимум должен быть меньше максимума. **Минимум и максимум могут быть внутренними словами** (от %MW0 до %MW2999), внутренними константами (от %KW0 до %KW255) или значениями между 1 и 10,000.
Manual mode (Ручной режим) **Bit** (Бит) **Output** (Выход)	Укажите, хотите ли Вы изменить режим PID на ручной режим. Доступны три опции: **Разрешено**, **Запрещено** или адрес бита. Если Вы выбрали адрес бита, Вы можете переключиться в ручной режим (бит = 1) или переключиться в автоматический режим (бит = 0) через программу, изменяя связанный бит, который является либо внутренним битом (от %M0 до %M255) либо входом (от %Ix.0 до %Ix.32). Поле Выход ручного режима должно содержать значение, которое вы хотите присвоить аналоговому выходу, когда PID в ручном режиме. Выход может быть словом (от %MW0 до %MW2999) или прямым значением.
Analog output (Аналоговый выход)	Укажите выход PID в автоматическом режиме. Аналоговый выход может быть %MW-типа (от %MW0 до %MW2999) или %QW-типа (%QWx.0).
PWM output (Выход PWM) enabled (разрешен) **Period (0.1s)** (Период) **Output** (Выход)	Поставьте флажок, если вы хотите использовать функцию PWM PID. Укажите период модуляции в поле **Period (0.1s)**. Этот период должен быть между 1 и 500 и может быть внутренним словом (от %MW0 до %MW2999) или внутренней константой (от %KW0 до %KW255). Укажите бит выхода PWM в поле **Output**.Это может быть внутренний бит (от %M0 до %M255) или выход (от %Qx.0 до %Qx.32).
Diagram (Диаграмма)	Диаграмма позволяет Вам посмотреть различные возможности, доступные для конфигурирования PID.

Примечание: Термин Обратный в поле Действие используется для достижения высокой уставки (например: для нагревания).



Термин Прямой в поле Действие используется для достижения низкой уставки (например: для охлаждения).



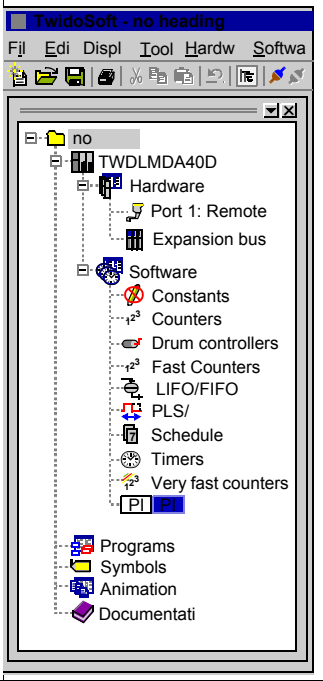
Как получить доступ к отладке PID

Обзор

В следующих параграфах описано, как получить доступ к экранам отладки PID для контроллеров TWIDO.

Процедура

В следующей таблице описана процедура доступа к экранам отладки PID:

Шаг	действие
1	Убедитесь, что Вы находитесь в режиме **online**.
2	Откройте браузер. **Результат:** 
3	Двойной щелчок на **PID**. **Результат: Откроется окно конфигурации PID и по умолчанию отобразится вкладка Animation (См. *Вкладка Animation функции PID, стр. 397*).** **Примечание**: Вы можете также щелкнуть правой кнопкой на **PID** и выбрать опцию **Edit,** или выбрать **Software** Ж **PID** из меню, или использовать **Program** Ж **Configuration Editor** Ж **PID Icon** меню, или, при использовании последнего метода, выбрать PID и щелкнуть на иконке Увеличительного стекла и выбрать особый PID.

Вкладка Animation функции PID

Обзор

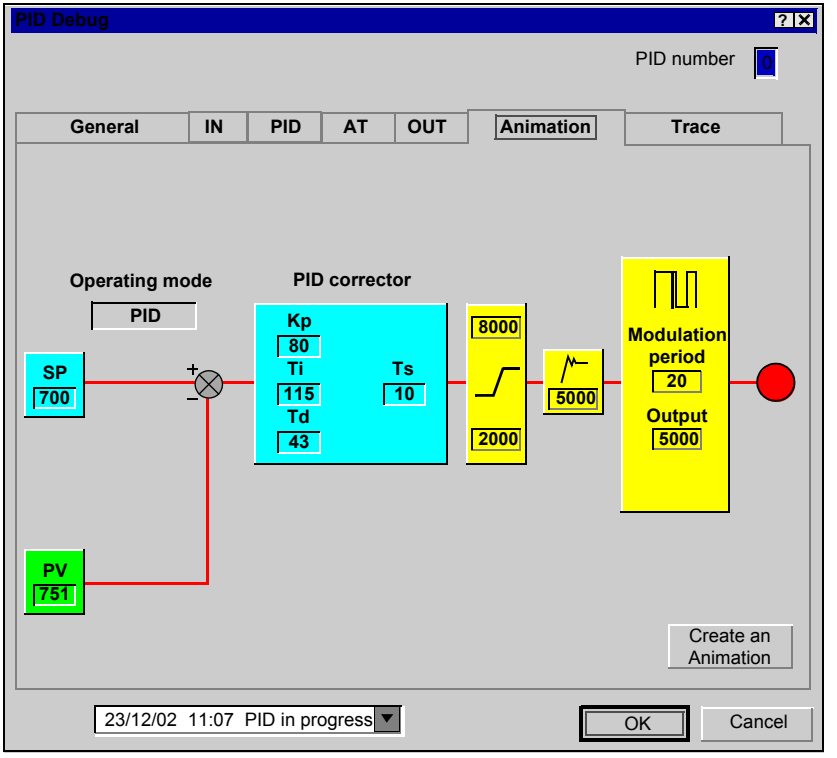
Вкладка используется для отладки PID.
Диаграмма, зависящая от типа PID, контролирует то, что Вы создали.
Отображаются только сконфигурированные элементы.

Экран является динамическим. Активные ссылки показаны красным цветом, а неактивные черным.

Примечание: Вкладка доступна в режиме online.

Вкладка Animation функции PID

Следующий экран используется для просмотра и отладки PID.



Описание

В следующей таблице описаны зоны окна.

Поле	Описание
PID number (Номер PID)	Укажите номер PID, который Вы хотите здесь сконфигурировать. Значение между 0 и 13, максимум 14 PID на приложение.
Operating mode (Режим работы)	Это поле показывает текущий режим работы PID.
Create an Animation Table (Создать анимационную таблицу)	Щелкните на “**Create an Animation Table”**, для создания файла, содержащего все переменные, показанные на диаграмме, что позволяет Вам изменять их в режиме onlineи отлаживать PID.
list (список)	Список в нижней части экрана позволяет просмотреть последние 15 состояний PID в реальном времени. Этот список обновляется после каждого изменения, указывает дату и время состояния, также как и текущее состояние.

Вкладка Trace функции PID

Обзор

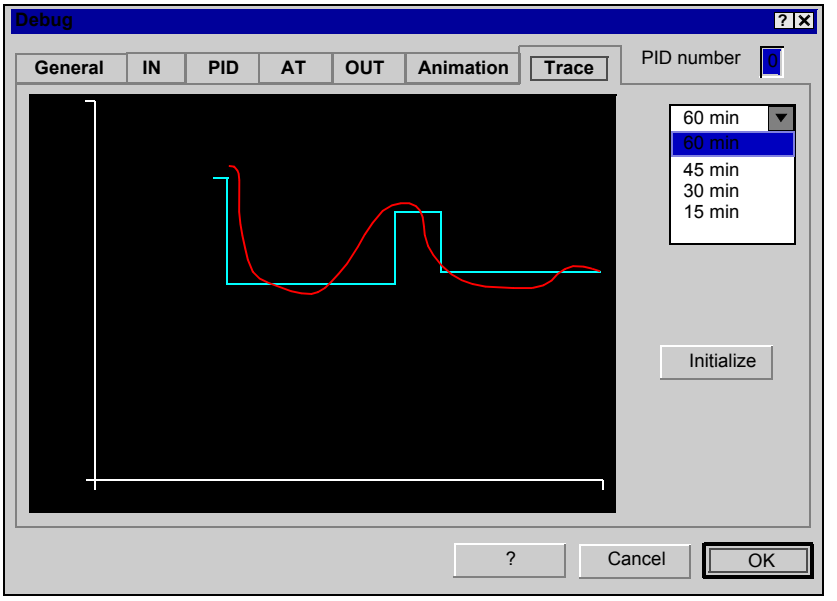
Эта вкладка позволяет просматривать функционирование PID и подстраивать его работу.

График начинает вычерчиваться, как только отображен экран отладки.

Примечание: Вкладка доступна в режиме online.

Вкладка Trace функции PID

Следующий экран используется для просмотра управления PID.



Описание

В следующей таблице описаны зоны окна.

Поле	Описание
PID number (Номер PID)	Укажите номер PID, который Вы хотите здесь сконфигурировать. Значение между 0 и 13, максимум 14 PID на приложение.
Chart (График)	В этой зоне изображаются графики уставки и параметра процесса. Шкала горизонтальной оси (X) определяется при помощи меню в верхней правой части окна. Шкала вертикальной оси определяется при помощи входных конфигурационных значений PID (с или без преобразования). Она автоматически оптимизируется так, чтобы добиться лучшего просмотра графиков.
Horizontal axis scale menu (Меню шкалы горизонтальной оси)	Это меню позволяет изменять шкалу горизонтальной оси. Вы можете выбрать из 4 значений: 15, 30, 45 или 60 минут.
Initialize (Инициализация)	Эта кнопка очищает график и перезапускает вычерчивание.

Метод регулировки параметров PID

Введение

Существует множество методов регулировки параметров PID, мы предлагаем метод Ziegler и Nichols, который имеет два варианта:

- регулировка закрытой петли,
- регулировка открытой петли.

Перед применением одного из этих методов, Вы должны установить направление действия PID:

- если увеличение выхода OUT вызывает увеличение измерения PV, сделайте PID обратным (KP > 0),
- если вызывает уменьшение PV, сделайте PID прямым (KP < 0).

Регулировка закрытой петли

Этот принцип заключается в использовании пропорциональной команды (Ti = 0, Td = 0) для запуска процесса увеличением производства, пока он не начнет колебаться снова после применения уровня к корректирующей уставке PID. Все это требуется для увеличения критического уровня производства (Kpc), который вызвал незатухающие колебания и период колебаний (Tc), для уменьшения значений, дающих оптимальную регуляцию регулятора.
В соответствии с типом регулятора (PID или PI), подгонка коэффициентов выполняется следующими значениями:

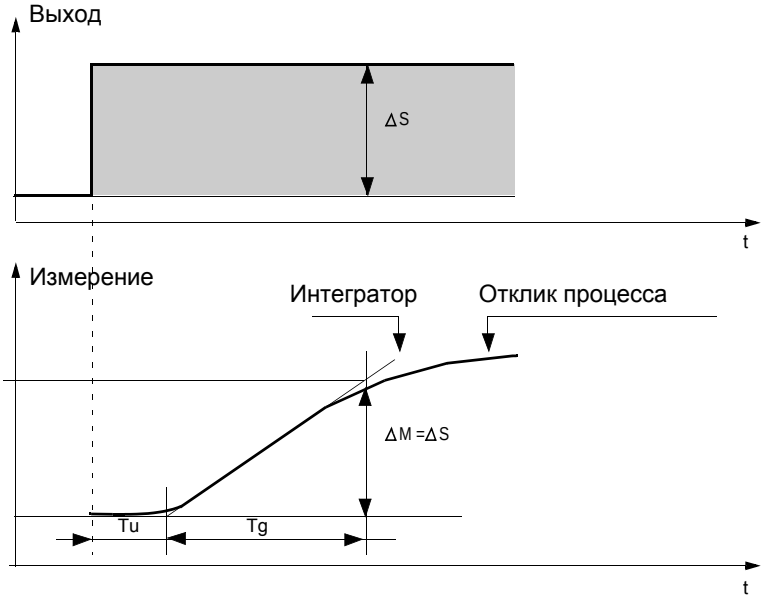
-	Kp	Ti	Td
PID	Kpc/1,7	Tc/2	Tc/8
PI	Kpc/2,22	0,83 x Tc	-

где Kp = пропорциональное выполнение, Ti = время интегрирования и TD = время производной.

Примечание: Этот метод регулировки обеспечивает очень динамичную команду, которая может выразить себя через не желаемые завышения во время изменения установочных пульсаций. В этом случае, понижайте значение производства, пока не получите требуемое поведение.

Регулировка открытой петли

Когда регулятор в ручном режиме, примените уровень к выходу и сделайте запуск отклика процедуры таким же как интегратор с чистым временем задержки.



Точка пересечения в правой части, которая отражает интегратор с осями времени, определяет время Tu. Время Tg определяется как время, необходимое контролируемой переменной (измерению) для достижения того же размера изменения (% шкалы), как у выхода регулятора.
В соответствии с типом регулятора (PID или PI), подгонка коэффициентов выполняется следующими значениями:

-	**Kp**	**Ti**	**Td**
PID	-1,2 Tg/Tu	2 x Tu	0,5 x Tu
PI	-0,9 Tg/Tu	3,3 x Tu	-

где Kp = пропорциональное выполнение, Ti = время интегрирования и TD = время производной.

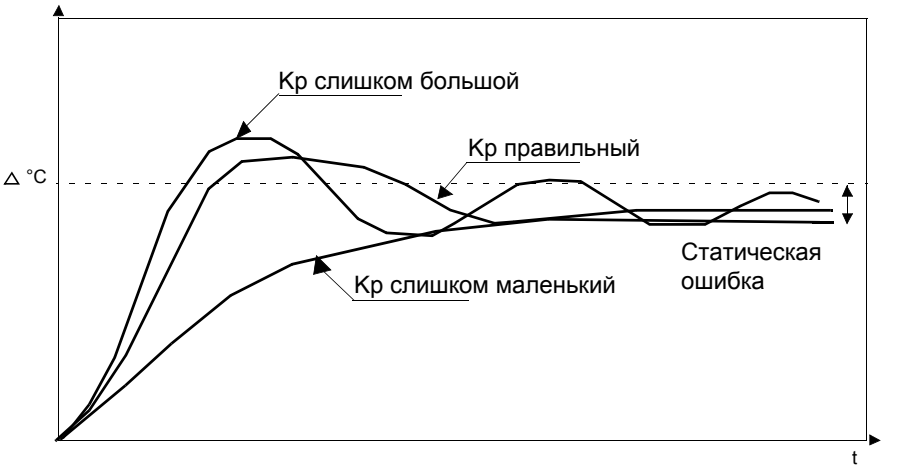
Примечание: Обращайте внимание на единицы измерения. Если подгонка выполняется в PL7, умножайте полученное значение для KP на 100.

Этот метод регулировки также обеспечивает очень динамичную команду, которая может выразить себя через не желаемые завышения во время изменения установочных пульсаций. В этом случае, понижайте значение производства, пока не получите требуемое поведение. Метод интересен, потому что он не требует допущений о природе и порядке процедуры. Вы можете применять его также для стабилизации процедур, таких ка интегрирующие процедуры. Он действительно интересен в случае медленных процедур (стекольная промышленность...), потому что от пользователя требуется только начало отклика для регулирования коэффициентов Kp, Ti и Td.

Роль и влияние параметров PID

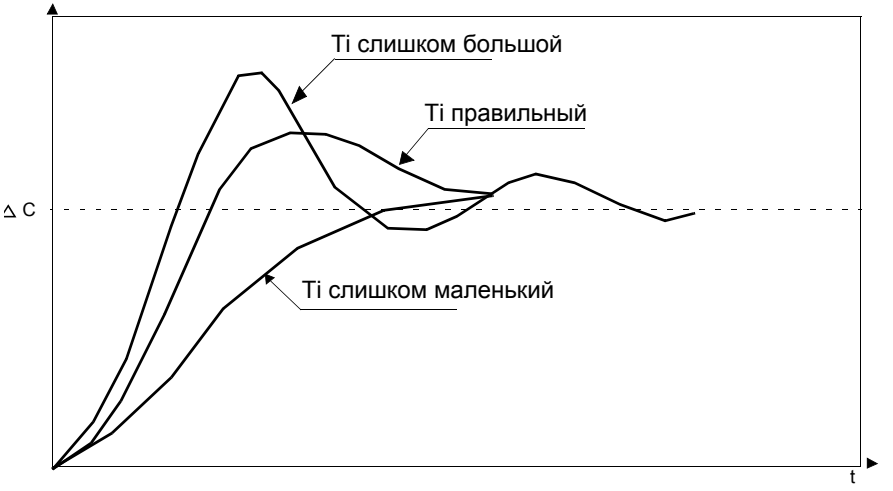
Влияние пропорционального действия

Пропорциональное действие используется для влияния на скорость отклика процесса. Чем выше коэффициент, тем быстрее отклик, и ниже статическая ошибка (в прямом соотношении), хотя уменьшается стабильность. Должен быть найден подходящий компромисс между стабильностью и скоростью. Влияние интегрального действия на отклик процесса к делению шкалы следующий:



Влияние интегрального действия

Интегральное действие используется для уничтожения статической ошибки (отклонение между параметром процесса и уставкой). Чем выше уровень интегрального действия (ниже Ti),тем быстрее отклик и больше ухудшается стабильность. Также необходимо найти подходящий компромисс между скоростью и стабильностью. Влияние интегрального действия на отклик процесса к делению шкалы следующее:

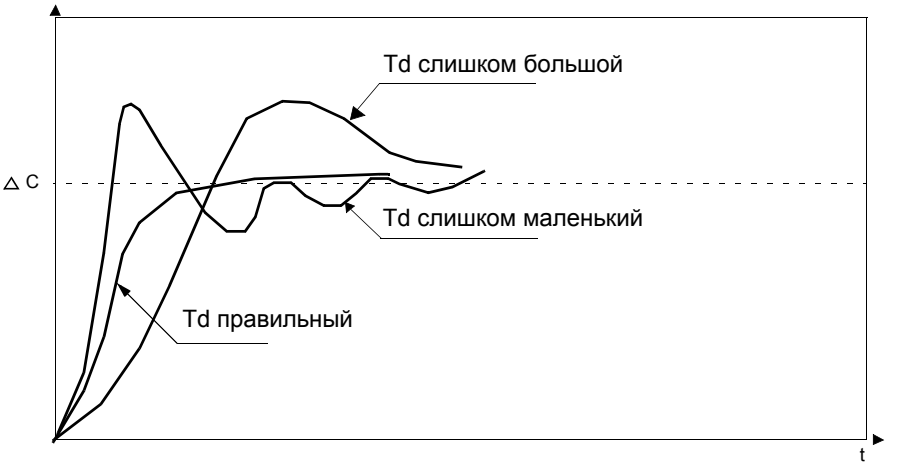


Примечание: Низкий Ti означает высокий уровень интегрального действия.

где Kp = пропорциональное увеличение, Ti = время интегрирования и TD = время производной

Влияние действия по производной

Действие по производной является предупреждающим. На практике, оно добавляет продолжительность, которая принимает во внимание скорость изменения в производной, которая делает возможным предупредить изменения ускорив время отклика процесса, когда производная увеличивается, и их замедлением, когда производная уменьшается. Чем выше уровень действия по производной (выше Td), тем быстрее отклик. должен быть найден подходящий компромисс между скоростью и стабильностью. Влияние действия по производной на отклик процесса к делению шкалы следующее:



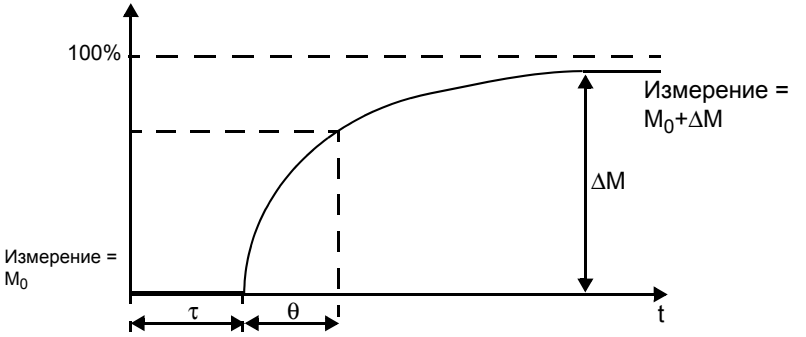
Пределы контролирующей петли PID

Если процесс приспосабливается к чистой задержке первого порядка передаточной функции: $(H(p)) = K\frac{(e^{(-\tau)p})}{(1+\theta p)}$

где:

τ =задержка модели,

θ = константа времени модели,



Производительность процесса контроля зависит от отношения $\frac{\tau}{\theta}$

Подходящий процесс управления PID достигается в следующем интервале: 2-$\frac{\tau}{\theta}$-20

Для $\frac{\tau}{\theta}$ <2, другими словами для быстрых регулирующих циклов (ниже θ) или для процессов с большими задержками (выше t), процесс управления PID не подходит. В этих случаях должны использоваться более сложные алгоритмы.

для $\frac{\tau}{\theta}$ >20, подходит процесс управления при помощи порогов плюс гистерезиса.

15.4 Инструкции для работы с числами с плавающей точкой

Обзор

Цель этого раздела

В этом разделе описаны дополнительные инструкции языка TwidoSoft для работы с числами с плавающей точкой (См. *Объекты с плавающей точкой и двойные слова, стр. 31*).
инструкции сравнения и присваивания описаны в разделе *Цифровая обработка, стр. 297*

Содержание раздела

Этот раздел содержит следующие темы:

Арифметические инструкции для чисел с плавающей точкой

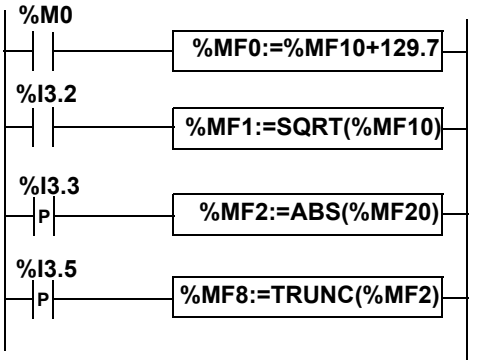
Общие положения

Эти инструкции используются для выполнения арифметических операций над двумя операндами или над одним операндом.

+	сложение двух операндов	**SQRT**	квадратный корень операнда
-	вычитание двух операндов	**ABS**	модуль операнда
*****	умножение двух операндов	**TRUNC**	целая часть числа с ПТ
/	деление двух операндов	**EXP**	натуральная экспоненциальная функция
LOG	десятичный логарифм	**EXPT**	порядок числа
LN	натуральный логарифм		

Структура

Язык лестничной логики



Язык списка инструкций

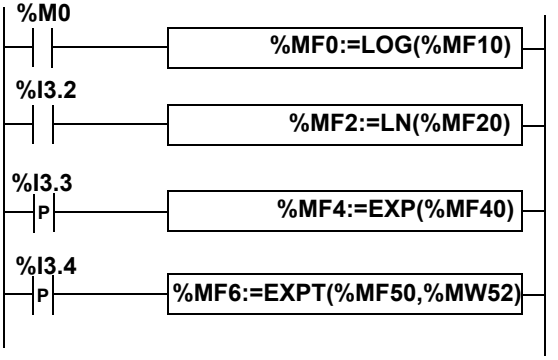
LD %M0
[%MF0:=%MF10+129.7]

LD %I3.2
[%MF1:=SQRT(%MF10)]

LDR %I3.3
[%MF2:=ABS(%MF20)]

LDR %I3.5
[%MF8:=TRUNC(%MF2)]

Язык лестничной логики



Язык списка инструкций
LD %M0
[%MF0:=LOG(%MF10]

LD %I3.2
[%MF2:=LN(%MF20)]

LDR %I3.3
[%MF4:=EXP(%MF40)]

LDR %I3.4
[%MF6:=EXPT(%MF50,%MW52)]

Синтаксис

Операторы и синтаксис арифметических инструкций над числами с ПТ

Операторы	**Синтаксис**
+, - *, /	Op1:=Op2 Оператор Op3
SQRT, ABS, TRUNC, LOG, EXP, LN	Op1:=Оператор (Op2)
EXPT	Op1:=Оператор (Op2,Op3)

Примечание: Когда Вы выполняете сложение или вычитание двух чисел с ПТ, два операнда должны соответствовать условию : $Op1 > Op2 \times 2^{-24}$, где Op1>Op2. Если это условие не выполняется, результат будет равен Операнду 1 (Op1). Этот феномен не имеет большого значения в случае изолированной операции, потому что ошибка результата очень маленькая (2^{-24}), но он может иметь непредсказуемые последствия, если вычисления повторяются.
Например, в случае, когда инструкция **%MF2:= %MF2 + %MF0** повторяется бесконечно. Если начальные значения %MF0 = 1.0 и %MF2 = 0, значение %MF2 блокируется на 16777216.
Мы рекомендуем Вам быть внимательными при программировании повторяющихся вычислений. Однако, если Вы хотите запрограммировать этот вид вычислений, задача клиентского приложения управлять ошибками усечения.

Операнды арифметических инструкций над числами с ПТ:

Операторы	**Операнд 1 (Op1)**	**Операнд 2 (Op2)**	**Операнд 3 (Op3)**
+, - *, /	%MFi	%MFi, %KFi, прямое значение	%MFi, %KFi, прямое значение
SQRT, ABS, LOG, EXP, LN	%MFi	%MFi, %KFi	[-]
TRUNC	%MFi	%MFi, %KFi	[-]
EXPT	%MFi	%MFi, %KFi	%MWi, %KWi, прямое значение

Правила использования

- Операции над числами с ПТ и целыми числами не могут смешиваться. Операции преобразования (См. *Инструкции преобразования чисел Целое <-> С плавающей точкой, стр. 416*) конвертирует в один или другой формат.)
- Системный бит %S18 управляется так же, как для операций над целыми (См. *Инструкции арифметических операций над целыми числами, стр. 306*), слово %SW17 (См. *системные слова (%SW), стр. 442*) указывает на причину ошибки.
- Когда операнд функции является недействительным числом (например: логарифм отрицательного числа), результат не определен или равен бесконечности, бит %S18 =1, слово %SW17 указывает причину ошибки.

Тригонометрические инструкции

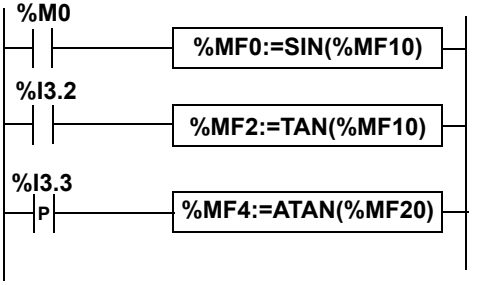
Общие положения

Эти инструкции позволяют пользователю выполнять тригонометрические операции.

SIN	синус угла, выраженного в радианах,	**ASIN**	арксинус (результат между $-\frac{\pi}{2}$ и $\frac{\pi}{2}$)
COS	косинус угла, выраженного в радианах,	**ACOS**	арккосинус (результат между 0 и π)
TAN	тангенс угла, выраженного в радианах,	**ATAN**	арктангенс (результат между $-\frac{\pi}{2}$ и $\frac{\pi}{2}$)

Структура

Язык лестничной логики



Язык списка инструкций

LD %M0
[%MF0:=SIN(%MF10)]

LD %I3.2
[%MF2:=TAN(%MF10)]

LDR %I3.3
[%MF4:=ATAN(%MF20)]

Язык структурированного текста

```
IF %M0 THEN
 %MF0:=SIN(%MF10);
END_IF;
IF %I3.2 THEN
 %MF2:=TAN(%MF10);
END_IF;
IF %I3.3 THEN
 %MF4:=ATAN(%MF20);
END_IF;
```

Синтаксис

Операторы, операнды и синтаксис инструкций тригонометрических операций

Операторы	Синтаксис	Операнд 1 (Op1)	Операнд 2 (Op2)
SIN, COS, TAN, ASIN, ACOS, ATAN	Op1:=Operator(Op2)	%MFi	%MFi, %KFi

Правила использования

- когда операнд функции является недействительным числом (например: арккосинус числа большего 1), результат не определен или является бесконечностью, бит %S18 =1, слово %SW17 (См. *Системные слова (%SW), стр. 442*) указывает на причину ошибки.
- функции SIN/COS/TAN допускают в качестве параметра угол от -4096π до 4096π, но их точность постепенно уменьшается для углов вне периода -2π и $+2\pi$, потому что неточность вносится приведением параметра к модулю 2π перед каждой операцией.

Инструкции преобразования

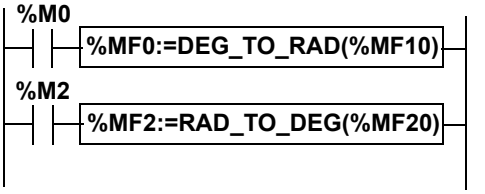
Общие положения

Эти инструкции используются для выполнения операций преобразования.

DEG_TO_RAD	преобразование градусов в радианы, результат - угол между 0 и 2π
RAD_TO_DEG	косинус угла, выраженного в радианах, результат - угол между 0 и 360 градусами

Структура

Язык лестничной логики



Язык списка инструкций

LD %M0
[%MF0:=DEG_TO_RAD(%MF10)]

LD %M2
[%MF2:=RAD_TO_DEG(%MF20)]

Язык структурированного текста

IF %M0 THEN
%MF0:=DEG_TO_RAD(%MF10);
END_IF;
IF %M2 THEN
%MF2:=RAD_TO_DEG(%MF20);
END_IF;

Синтаксис

Операторы, операнды и синтаксис инструкций преобразования

Операторы	Синтаксис	Операнд 1 (Op1)	Операнд 2 (Op2)
DEG_TO_RAD **RAD_TO_DEG**	Op1:=Operator(Op2)	%MFi	%MFi, %KFi

Правила использования

Угол для преобразования должен быть между -737280.0 и +737280.0 (для преобразования DEG_TO_RAD) или между -4096π и 4096π (для преобразования RAD_TO_DEG).
Для значений за пределами этих диапазонов будет отображаться результат + 1.#NAN, биты %S18 и %SW17:X0 устанавливаются в 1.

Инструкции преобразования чисел Целое <-> С плавающей точкой

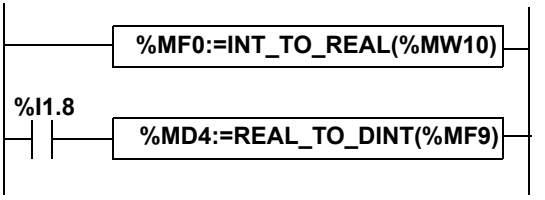
Общие положения

Предлагается четыре инструкции преобразования.
Список инструкций преобразования чисел Целое <-> С плавающей точкой :

INT_TO_REAL	преобразование Целое слово --> ПТ
DINT_TO_REAL	преобразование Двойное целое слово --> ПТ
REAL_TO_INT	преобразование ПТ --> Целое слово (результат ближайшее алгебраическое значение)
REAL_TO_DINT	преобразование ПТ --> Двойное целое слово (результат ближайшее алгебраическое значение)

Структура

Язык лестничной логики



Язык списка инструкций

```
LD TRUE
[%MF0:=INT_TO_REAL(%MW10)]

LD I1.8
[%MD4:=REAL_TO_DINT(%MF9)]
```

Язык структурированного текста

```
%MF0:=INT_TO_REAL(%MW10);
IF %I1.8 THEN
 %MD4:=REAL_TO_DINT(%MF9);
END_IF;
```

Синтаксис

Операторы и синтаксис (преобразование Целое слово --> ПТ):

Операторы	**Синтаксис**
INT_TO_REAL	Op1=INT_TO_REAL(Op2)

Операнды (преобразование Целое слово --> ПТ):

Операнд 1 (Op1)	**Операнд 2 (Op2)**
%MFi	%MWi,%KWi

Пример: преобразование Целое слово --> ПТ: 147 --> 1.47e+02

Операторы и синтаксис (преобразование Двойное целое слово --> ПТ):

Операторы	**Синтаксис**
DINT_TO_REAL	Op1=DINT_TO_REAL(Op2)

Операнды (преобразование Двойное целое слово --> ПТ):

Операнд 1 (Op1)	**Операнд 2 (Op2)**
%MFi	%MDi,%KDi

Пример: преобразование Двойное целое слово --> ПТ: 68905000 --> 6.8905e+07

Операторы и синтаксис (преобразование ПТ -->Целое слово или Целое двойное слово):

Операторы	**Синтаксис**
REAL_TO_INT	Op1=Operator(Op2)
REAL_TO_DINT	

Операнды (преобразование ПТ -->Целое слово или Целое двойное слово):

Тип	**Операнд 1 (Op1)**	**Операнд 2 (Op2)**
Слова	%MWi	%MFi, %KFi
Двойные слова	%MDi	%MFi, %KFi

Пример:
преобразование ПТ --> Целое слово: 5978.6 --> 5979
преобразование ПТ --> Целое двойное слово: -1235978.6 --> -1235979

Примечание: Если во время преобразования действительного числа в целое (или действительного в целое двойное) значение с ПТ выходит за пределы слова (или двойного слова), бит %S18устанавливается в 1.

Точность округления

Стандарт IEEE 754 определяет 4 режима округления для операций с ПТ. В описанных выше инструкциях используется режим "округления к ближайшему":
"если ближайшие представимые числа находятся на одинаковом расстоянии от теоретического результата, число округляется к значению, у которого младший значащий бит равен 0".
В определенных случаях, результат округления может быть меньше или больше округляемого числа.

Например:
Округление значения 10.5 -> 10
Округление значения 11.5 -> 12

15.5 Инструкции для работы с таблицами

Обзор

Цель этого раздела

В этом разделе описываются специальные инструкции для таблиц:
- двойных слов,
- объектов с плавающей точкой.

инструкции присваивания для таблиц описаны в главе "Базовые инструкции" (См. *Присваивание таблиц слов, двойных слов и слов с ПТ, стр. 302*).

Содержание раздела

Этот раздел содержит следующие темы:

Функция суммирования таблиц

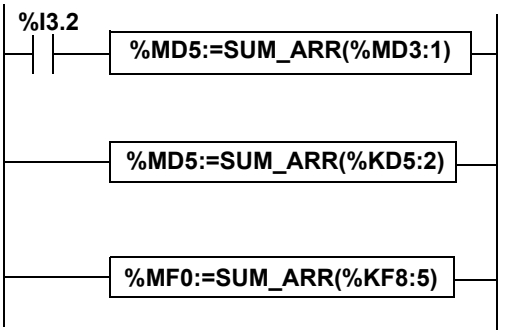
Общие положения

Функция SUM_ARR складывает вместе все элементы таблицы объектов:

- если таблица состоит из двойных слов, результат возвращается в форме двойного слова
- если таблица состоит из слов с плавающей точкой, результат возвращается в форме слова с плавающей точкой

Структура

Язык лестничной логики



Язык списка инструкций

LD %I3.2
[%MD5:=SUM_ARR(%MD3:1)]
%MD5:=SUM_ARR(%KD5:2)
%MF0:=SUM_ARR(%KF8:5)

Синтаксис

Синтаксис инструкции суммирования таблиц:

```
Res:=SUM_ARR(Tab)
```

Параметры инструкции суммирования таблиц

Тип	Результат (res)	Таблица (Tab)
Таблицы двойных слов	%MDi	%MDi:L,%KDi:L
Таблицы слов с ПТ	%MFi	%MFi:L,%KFi:L

Примечание: Когда результат выходит за пределы допустимого диапазона двойного слова, системный бит %S18 устанавливается в 1.

Пример

%MD5:=SUM(%MD30:4)
where %MD30=10, %MD31=20, %MD32=30, %MD33=40
%MD5=10+20+30+40=100

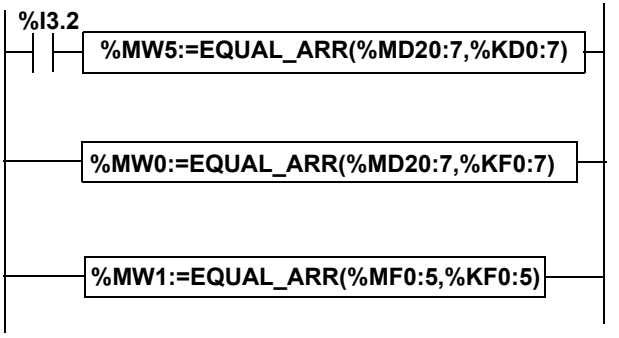
Функция сравнения таблиц

Общие положения

Функция EQUAL _ARR выполняет поэлементное сравнение двух таблиц. Если существует различие, ранг первого отличного элемента возвращается в формате слова, иначе, возвращаемое значение равно -1.
сравнение выполняется над всей таблицей.

Структура

Язык лестничной логики



Язык списка инструкций
LD %I3.2
[%MW5:=EQUAL_ARR(%MD20:7,KD0:7)]

Structured Text language
%MW0:=EQUAL_ARR(%MD20:7,%KF0:7)

%MW1:=EQUAL_ARR(%MF0:5,%KF0:5)

Синтаксис

Синтаксис инструкции сравнения таблиц:

```
Res:=EQUAL_ARR(Tab1,Tab2)
```

Параметры инструкции сравнения таблиц:

Тип	Результат (Res)	Таблицы (Tab1 и Tab2)
Таблицы двойных слов	%MWi	%MDi:L,%KDi:L
Таблицы слов с ПТs	%MWi	%MFi:L,%KFi:L

Примечание:
- таблицы должны быть одной длины и одинакового типа.

Пример

%MW5:=EQUAL_ARR(%MD30:4,%KD0:4)
Сравнение двух таблиц:

Ранг	Таблица слов	Таблица констант	Различие
0	%MD30=10	%KD0=10	=
1	%MD31=20	%KD1=20	=
2	%MD32=30	%KD2=60	Отличается
3	%MD33=40	%KD3=40	=

Значение слова %MW5 =2 (ранг первого отличного)

Функции поиска в таблице

Общие положения

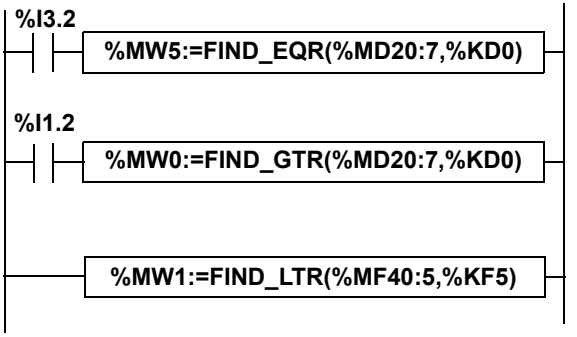
Существует 3 функции поиска:

- **FIND_EQR**: ищет позицию двойного слова или слова с ПТ первого элемента, равного данному значению
- **FIND_GTR**: ищет позицию двойного слова или слова с ПТ первого элемента, большего данного значения
- **FIND_LTR**: ищет позицию двойного слова или слова с ПТ первого элемента, меньшего данного значения

Результат этих инструкций равен рангу первого найденного элемента или -1, если ничего не было найдено.

Структура

Язык лестничной логики



Язык списка инструкций

LD %I3.2
[%MW5:=FIND_EQR(%MD20:7,KD0)]
LD %I1.2
[**%MW0:=FIND_GTR(%MD20:7,%KD0)**]
%MW1:=FIND_LTR(%MF40:5,%KF5)

Синтаксис

Синтаксис инструкций поиска в таблице:

Функция	**Синтаксис**
FIND_EQR	Res:=Function(Tab,Val)
FIND_GTR	
FIND_LTR	

Параметры поиска в таблице двойных слов и слов с ПТ:

Тип	**Результат (Res)**	**Таблица (Tab)**	**Значение (val)**
Таблицы слов с ПТ	%MWi	%MFi:L,%KFi:L	%MFi,%KFi
Таблицы двойных слов	%MWi	%MDi:L,%KDi:L	%MDi,%KDi

Пример

%MW5:=FIND_EQR(%MD30:4,%KD0)
Поиск позиции первого двойного слова =%KD0=30 в таблице:

Ранг	**Таблица слов**	**результат**
0	%MD30=10	-
1	%MD31=20	-
2	%MD32=30	Значение (val), ранг
3	%MD33=40	-

Функции поиска в таблице максимального и минимального значений

Общие положения

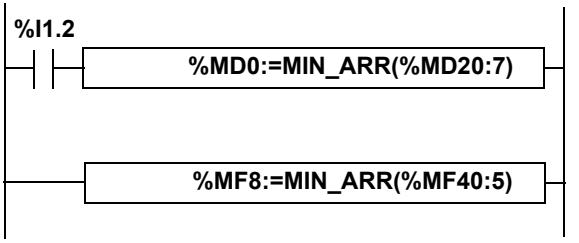
Существуют 2 функции поиска:

- **MAX_ARR**: ищет максимальное значение в таблице двойных слов и слов с ПТ
- **MIN_ARR**: ищет минимальное значение в таблице двойных слов и слов с ПТ

Результат этих инструкций равен максимальному значению (или минимальному), найденному в таблице.

Структура

Язык лестничной логики



Язык списка инструкций

LD %I1.2
[**%MD0:=MIN_ARR(%MD20:7)**]
%MF8:=MIN_ARR(%MF40:5)

Синтаксис

Синтаксис инструкций поиска в таблице максимального и минимального значений:

Функция	**Синтаксис**
MAX_ARR	Res:=Function(Tab)
MIN_ARR	

Параметры инструкций поиска в таблице максимального и минимального значений:

Тип	**Результат (Res)**	**Таблица (Tab)**
Таблицы двойных слов	%MDi	%MDi:L,%KDi:L
Таблицы слов с ПТ	%MFi	%MFi:L,%KFi:L

Количество вхождений числа в таблицу

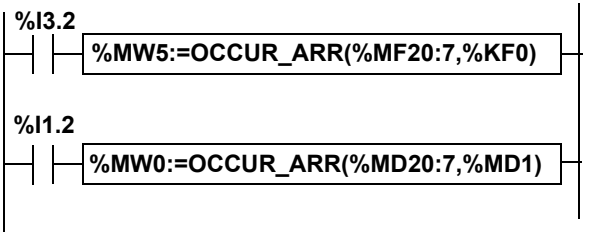
Общие положения

Функция поиска:

- **OCCUR_ARR**: ищет в таблице двойных слов или слов с ПТ число элементов, равных данному значению

Структура

Язык лестничной логики



Язык списка инструкций

LD %I3.2
[%MW5:=OCCUR_ARR(%MF20:7,%KF0)]
LD %I1.2
[**%MW0:=OCCUR_ARR(%MD20:7,%MD1)**

Синтаксис

Синтаксис инструкции поиска количества вхождений числа в таблицу:

Функция	**Синтаксис**
OCCUR_ARR	Res:=Function(Tab,Val)

Параметры инструкции поиска количества вхождений числа в таблицу:

Тип	**Результат (Res)**	**Таблица (Tab)**	**Значение (Val)**
Таблицы двойных слов	%MWi	%MDi:L,%KDi:L	%MDi,%KDi
Таблицы слов с ПТ	%MFi	%MFi:L,%KFi:L	%MFi,%KFi

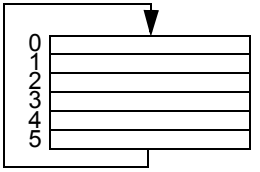
Функции циклического сдвига таблицы

Общие положения

Существует две функции сдвига:

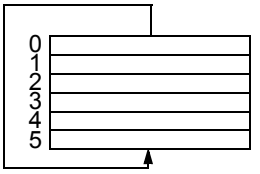
- **ROL_ARR**: выполняет циклический сдвиг элементов на n позиций сверху вниз в таблице слов с ПТ

Иллюстрация функции ROL_ARR



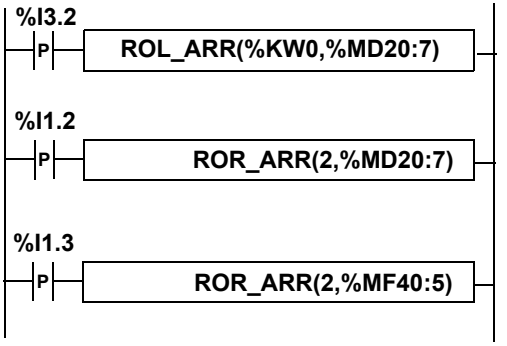
- **ROR_ARR**: выполняет циклический сдвиг элементов на n позиций снизу вверх в таблице слов с ПТ

Иллюстрация функции ROR_ARR



Структура

Язык лестничной логики



Язык списка инструкций

LDR %I3.2
[ROL_ARR(%KW0,%MD20:7)]
LDR %I1.2
[**ROR_ARR(2,%MD20:7)**]
LDR %I1.3
[**ROR_ARR(2,%MF40:5)]**

Синтаксис

Синтаксис инструкций циклического сдвига в таблицах двойных слов или слов с ПТ **ROL_ARR** и **ROR_ARR**

Функция	**Синтаксис**
ROL_ARR	Function(n,Tab)
ROR_ARR	

Параметры инструкций циклического сдвига в таблицах двойных слов или слов с ПТ: **ROL_ARR** и **ROR_ARR**:

Тип	**Число позиций (n)**	**Таблица (Tab)**
Таблицы слов с ПТ	%MWi, прямое значение	%MFi:L
Таблицы двойных слов	%MWi, прямое значение	%MDi:L

Примечание: если значение n отрицательно или =0, сдвиг не выполняется.

Функция сортировки таблиц

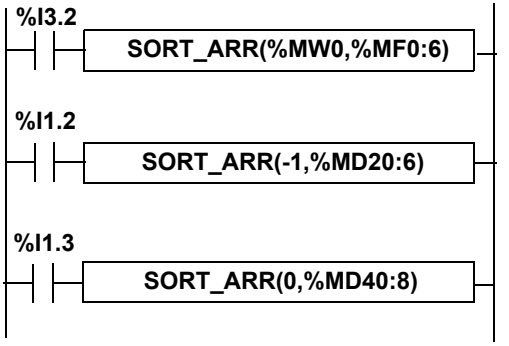
Общие положения

Функция сортировки доступна в следующем виде:

- **SORT_ARR**: выполняет сортировку элементов по возрастанию или по убыванию в таблице двойных слов или слов с ПТ и сохраняет результат в той же таблице.

Структура

Язык лестничной логики



Язык списка инструкций

LD %I3.2
[SORT_ARR(%MW20,%MF0:6)]
LD %I1.2
[**SORT_ARR(-1,%MD20:6)**]
LD %I1.3
[**SORT_ARR(0,%MF40:8)**

Синтаксис

Синтаксис функций сортировки таблицы:

Функция	**Синтаксис**
SORT_ARR	Function(direction,Tab)

- Параметр "direction" задает порядок сортировки: direction > 0, сортировка по возрастанию; direction < 0, сортировка по убыванию, direction = 0, сортировка не выполняется.
- результат (отсортированная таблица) возвращается в параметр Tab (таблица для сортировки).

параметры функций сортировки таблицы:

Тип	**Направление сортировки**	**таблица (Tab)**
Таблицы двойных слов	%MWi, прямое значение	%MDi:L
Таблицы слов с ПТ	%MWi, прямое значение	%MFi:L

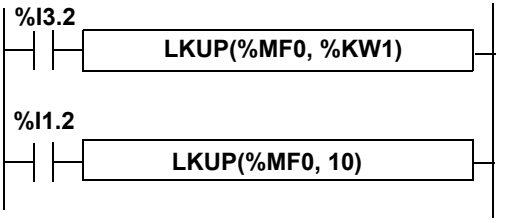
Функция интерполяции таблицы значений с плавающей точкой

Общие положения

Инструкция **LKUP** используется для вызова графического представления значений с ПТ в таблице путем линейной интерполяции точек. Это функция с переменной шириной точек с ПТ в таблице.
Пользователь должен задать число точек в таблице для интерполяции.

Структура

Язык лестничной логики



Язык списка инструкций
LD %I3.2
[LKUP(%MF0, %KW1)]
LD %I1.2
[**LKUP(%MF0, 10)**

Интерполяция

Алгоритм интерполяции, применяемый к числам с ПТ в таблице, следующий:

$Y = Y_i + ((Y_{i+1} - Y_i)/(X_{i+1} - X_i))*(X - X_i)$

для X_i J X J X_{i+1} где i = 1 ... (m-1)

Синтаксис

Синтаксис функции:

Функция	**Синтаксис**
LKUP	Result=LKUP(Tab, Nb)

Операнды функции:

Таблица (Tab)	**Результат (Res)**	**Nb**
%MFi	%MWi	прямое значение, %MWi, %KWi

Операнд "Nb" представляет число значений с ПТ в таблице, которые необходимо принять в расчет. Он должен быть четным числом.

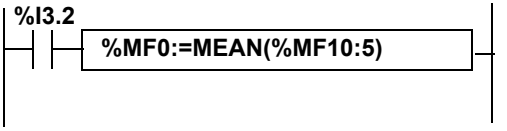
Функция нахождения среднего значения таблицы значений с плавающей точкой

Общие положения

Функция **MEAN** используется для расчета среднего значения от заданного числа значений в таблице чисел с ПТ.

Структура

Язык лестничной логики



Язык списка инструкций

LD %I3.2
[%MF0:=MEAN(%MF10:5)]

Синтаксис

Синтаксис инструкции нахождения среднего значения таблицы значений с плавающей точкой:

Функция	**Синтаксис**
MEAN	Result=Function(Op1)

Параметры инструкции нахождения среднего значения таблицы значений с плавающей точкой:

Операнд (Op1)	**Результат (Res)**
%MFi:L, %KFi:L	%MFi

Системные биты и системные слова

16

Обзор

Тема

В этой главе представлен обзор системных битов и системных слов, которые можно использовать для создания управляющих программ для контроллеров Twido.

Содержание главы

Эта глава содержит следующие темы:

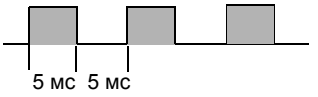
Системные биты (%S)

Введение

В следующем разделе представлена подробная информация о функциях системных битов и о том, как они контролируются.

Подробное описание

В следующей таблице представлен обзор системных битов и их контроль:

Системный бит	Функция	Описание	Начальное состояние	Контроль
%S0	“Холодный” пуск	Обычно установлен в 0, устанавливается в 1: ● Возвратом питания с потерей данных (сбой батареи), ● Программой пользователя или Редактором Анимационных Таблиц, ● Дисплеем оператора. Этот бит устанавливается в 1 во время первого полного сканирования. Он сбрасывается в 0 системой перед следующим сканированием.	0	S или U->S
%S1	“Теплый” пуск	Обычно установлен в 0, устанавливается в 1: ● Возвратом питания без потери данных, ● Программой пользователя или Редактором Анимационных Таблиц, ● Дисплеем оператора. Он сбрасывается в 0 системой в конце полного сканирования.	0	S или U->S
%S4 %S5 %S6 %S7	Масштаб по оси времени: 10 мс Масштаб: 100 мс Масштаб: 1 с Масштаб: 1 мин	Скорость изменений состояния измеряется внутренними часами. Они не синхронизированы со сканированием контроллера. Пример: %S4  	-	S
%S8	Тест соединения	Изначально установлен 1, этот бит используется для тестирования соединения, когда контроллер находится в “не сконфигурированном” состоянии. Чтобы измененить значение этого бита и статус выхода, используйте кнопки дисплея оператора. ● Установлен 1, сброс входа, ● Установлен в 0, тест соединения авторизирован.	1	U

Систем-ный бит	Функция	Описание	Нача-льное состо яние	Кон-троль
%S9	Сброс выходов	Обычно установлен в 0. Может быть установлен в 1 программой или терминалом (в Редакторе Анимационных Таблиц): ● В состоянии 1, выходы устанавливаются в 0, когда контроллер в режиме РАБОТА, ● В состоянии 0, выходы обновляются нормально.	0	U
%S10	Ошибка вх/вых	Обычно установлен в 1. Этот бит может быть установлен в 0 системой, когда обнаружена ошибка вх/вых.	1	S
%S11	Переполнение сторожевого таймера	Обычно установлен в 0. Этот бит может быть установлен в 1 системой, когда время выполнения программы (время сканирования) превышает максимальное время сканирования (программный сторожевой таймер). Переполнение сторожевого таймера вызывает изменение состояния контроллера в ОСТАНОВ.	0	S
%S12	ПЛК в режиме РАБОТА	Этот бит отображает состояние контроллера. Система устанавливает бит в 1, когда контроллер работает. Или в 0, когда он остановлен, инициализируется или в другом состоянии.	0	S
%S13	Первый цикл при режиме РАБОТА	Обычно установлен в 0, этот бит устанавливается в 1 системой во время первого сканирования, после того, как контроллер перешел в режим РАБОТА.	1	S
%S17	Превышение емкости	Обычно установлен в 0, устанавливается в 1 системой: ● Во время операции сдвига или циклического сдвига. Система переключает бит в 1. Он должен тестироваться программой после каждой операции, где есть риск переполнения, затем сбрасываться в 0 пользователем, если переполнение произошло.	0	S->U

Системный бит	Функция	Описание	Начальное состояние	Контроль
%S18	Арифметическое переполнение или ошибка	Обычно устанавливается в 0. Устанавливается в 1 в случае переполнения, когда выполняется 16 битная операция, для которой: ● Результат больше + 32 767 или меньше - 32 768, при одинарной длине, ● Результат больше + 2 147 483 647 или меньше - 2 147 483 648, при двойной длине, ● Результат больше + 3.402824E+38 или меньше - 3.402824E+38, при ПТ, ● Деление на 0, ● Квадратный корень отрицательного числа, ● Преобразование BTI или ITB не имеет значения: BCD значение выходит за пределы. Он должен тестироваться программой пользователя после каждой операции, где есть риск переполнения, затем сбрасываться в 0 пользователем, если переполнение произошло.	0	S->U
%S19	Выход за границы периода сканирования (периодическое выполнение)	Обычно установлен в 0, этот бит устанавливается в 1 системой в случае выхода за границы периода сканирования (время сканирования больше, чем период, определенный пользователем при конфигурации, или запрограммированный в %SW0). Этот бит сбрасывается в 0 пользователем.	0	S->U
%S20	Переполнение индекса	Обычно установлен в 0, устанавливается в 1, когда адрес индексированного объекта становится меньше 0 или больше максимального размера объекта. Он должен тестироваться программой после каждой операции, где есть риск переполнения, затем сбрасываться в 0, если переполнение произошло.	0	S->U
%S21	Инициализация GRAFCET	Обычно установлен в 0, устанавливается в 1: ● “Холодным” пуском, %S0=1, ● Программой пользователя только в части предварительной обработки, используя инструкцию Set (S %S21) или катушку установки -(S)- %S21, ● Терминалом. В состоянии 1 он вызывает инициализацию GRAFCET. Активные шаги дезактивируются, начальные шаги активируются. Он сбрасывается в 0 системой после инициализации GRAFCET.	0	U->S

Систем-ный бит	Функция	Описание	Нача-льное состо яние	Кон-троль
%S22	Сброс GRAFCET	Обычно установлен в 0, может быть установлен в 1 только программой при предобработке. В состоянии 1 он вызывает дезактивацию активных шагов всего GRAFCET. Он сбрасывается в 0 системой в начале выполнения последовательной обработки.	0	U->S
%S23	Предустановка и замораживание GRAFCET	Обычно установлен в 0, может быть установлен в 1 только программой в модуле предобработки. В состоянии 1 он подтверждает препозицию GRAFCET. Поддерживание этого бита в состоянии 1 замораживает GRAFCET (замораживает диаграмму). Он сбрасывается в 0 системой в начале выполнения последовательной обработки, чтобы гарантировать, что диаграмма GRAFCET выходит из замороженного состояния.	0	U->S
%S24	Функционирование дисплея	Обычно установлен в 0, этот бит может быть установлен в 1 пользователем. ● В состоянии 0 дисплей оператора функционирует нормально, ● В состоянии 1 дисплей оператора заморожен, остается в текущем экране, мигание запрещено, нажатие на кнопки не обрабатывается.	0	U->S
%S31	Маска событий	Обычно установлен в 1, этот бит может быть установлен в 0 пользователем. ● Установлен в 0, события не выполняются, помещаются в очередь. ● Установлен в, события могут быть выполнены, Этот бит может быть установлен в 0 пользователем и системой (при “холодном” перезапуске).	0	U->S
%S38	Разрешение помещать события в очередь событий	Обычно установлен в 1, этот бит может быть установлен в 0 пользователем. ● Установлен в 0, события не могут быть помещены в очередь событий. ● Установлен в 1, события помещаются в очередь событий сразу после обнаружения, Этот бит может быть установлен в 0 пользователем и системой (при “холодном” перезапуске).	0	U->S

Систем-ный бит	Функция	Описание	Нача-льное состо яние	Кон-троль
%S39	Насыщение очереди событий	Обычно установлен в 1, этот бит может быть установлен в 0 пользователем. ● Установлен в 0, все события выполнены, ● Установлен в 1, как минимум одно событие утеряно. Этот бит может быть установлен в 0 пользователем и системой (при “холодном” перезапуске).	0	U->S
%S50	Обновление даты и времени, используя слова с %SW49 по %SW53	Обычно установлен в 0, этот бит может быть установлен в 1 или 0 программой или через дисплей. ● Установлен в 0, дату и время можно читать, ● Установлен в 1, дату и время можно обновлять. Внутренние RTC контроллера обновляются по спаду %S50.	0	U->S
%S51	Статус часов	Обычно установлен в 0, этот бит может быть установлен в 1 или 0 программой или через дисплей. ● Установлен в 0, дата и время согласованы, ● Установлен в 1, дата и время должны быть инициализированы пользователем. Когда этот бит установлен в 1, данные часов не действительны. Дата и время могли никогда не конфигурироваться, батарея могла сесть, или корректирующая константа может быть неверна (никогда не конфигурировалась, различия между скорректированным значением часов и сохраненным значением или значение выходит за пределы). Изменение состояния из 1 в 0 приводит к записи корректирующей константы в RTC.	0	U->S
%S52	Ошибка RTC	Этот бит, устанавливающийся системой, указывает, что коррекция RTC не была проведена, и дата и время не правильные. ● Установлен в 0, дата и время согласованы, ● В состоянии 1, дата и время должны быть инициализированы.	0	S
%S59	Обновление даты и времени, используя слово %SW59	Обычно в 0, этот бит может быть установлен в 1 или 0 программой или через дисплей. ● Установлен в 0, системное слово %SW59 не регулируется, ● Установлен в 1, дата и время уменьшаются или увеличиваются в соответствии с фронтами управляющих битов, установленных в %SW59.	0	U
%S69	Отображение STAT LED	Установлен в 0, STAT LED выключен. Установлен в 1, STAT LED включен.	0	U

Систем-ный бит	Функция	Описание	Нача-льное состо яние	Кон-троль
%S95	Восстановление слов памяти	Этот бит может быть установлен, когда слова памяти были предварительно сохранены во внутреннюю EEPROM. После завершения система устанавливает этот бит обратно в 0, и число восстановленных слов памяти записывается в %SW97	0	U
%S96	Резервная программа OK	Этот бит может быть прочитан в любое время (либо программой, либо при настройке), особенно после “холодного” пуска или “теплого” перезапуска. ● Установлен в 0, резервная программа недействительна. ● Установлен в 1, резервная программа действительна.	0	S
%S97	Сохранение %MW OK	Этот бит может быть прочитан в любое время (либо программой, либо при настройке), особенно после “холодного” пуска или “теплого” перезапуска. ● Установлен в 0, сохраненные %MW не OK. ● Установлен в 1, сохраненные %MW OK.	0	S
%S100	Соединение коммуникаци-онного кабеля TwidoSoft	Показывает, соединен ли коммуникационный кабель TwidoSoft. ● Установлен в 1, либо коммуникационный кабель TwidoSoft не присоединен, либо подсоединен TwidoSoft. ● Установлен в 0, присоединен кабель дистанционной связи TwidoSoft.	-	S
%S101	Изменение адреса порта (протокол Modbus)	Используется для изменения адреса порта при помощи системных слов %SW101 (порт 1) и %SW102 (порт 2). Для этого %S101 должен быть установлен в 1. ● Установлен в 0, адрес не может быть изменен. Значения %SW101 и %SW102 соответствуют текущим адресам портов, ● Установлен в 1, адрес можно изменить, изменяя значения %SW101 (порт 1) и %SW102 (порт 2). После изменения значений системных слов, %S101 должен быть установлен обратно в 0.	0	U

Систем-ный бит	Функция	Описание	Нача-льное состо яние	Кон-троль
%S103 %S104	Использование протокола ASCII	Разрешает использование протокола ASCII на Comm 1 (%S103) или Comm 2 (%S104). Протокол ASCII конфигурируется, используя системные слова %SW103 и %SW105 для Comm 1, и %SW104 и %SW105 для Comm 2. ● Установлен в 0, используется протокол, сконфигурированный в Twido Soft, ● Установлен в 1, протокол ASCII используется на Comm 1 (%S103) или Comm 2 (%S104).В этом случае, системные слова %SW103 и %SW105 должны быть предварительно сконфигурированы для Comm 1, и %SW104 и %SW105 для Comm 2.	0	U
%S110	Обмены по дистанционной связи	Этот бит сбрасывается в 0 программой или через терминал. ● Установлен в 1 для master, все обмены по дистанционной связи (только удаленные вх/вых) завершены. ● Установлен в 1 для slave, обмен с master завершен.	0	S->U
%S111	Одиночный обмен по дистанционной связи	● Установлен в 0 для master, одиночный обмен по дистанционной связи завершен. ● Установлен в 1 для master, одиночный обмен по дистанционной связи активен.	0	S
%S112	Соединение дистанционной связи	● Установлен в 0 для master, дистанционная связь активирована. ● Установлен в 1 для master, дистанционная связь дезактивирована.	0	U
%S113	Конфигурация/ функционирование дистанционной связи	● Установлен в 0 для master или slave, конфигурация/ функционирование дистанционной связи OK. ● Установлен в 1 для master, конфигурация/ функционирование дистанционной связи имеет ошибку. ● Установлен в 1 для slave, конфигурация/функционирование дистанционной связи имеет ошибку.	0	S->U
%S118	Ошибка удаленных вх/ вых	Обычно установлен в 1. Этот бит может быть установлен в 0, когда обнаружена ошибка вх/вых дистанционной связи.	1	S
%S119	Ошибка локальных вх/ вых	Обычно установлен в 1. Этот бит может быть установлен в 0, когда обнаружена ошибка вх/вых дистанционной связи. %SW118 определяет природу ошибки. Устанавливается в 1, когда ошибка исчезает.	1	S

Описание аббревиатур

Расшифровка аббревиатур:

Аббревиатура	**Описание**
S	Контролируется системой
U	Контролируется пользователем
U->S	Устанавливается в 1 пользователем, сбрасывается в 0 системой
S->U	Устанавливается в 1 системой, сбрасывается в 0 пользователем

Системные слова (%SW)

Введение

В следующем разделе представлена подробная информация о функциях системных слов и о том, как они контролируются.

Подробное описание В следующей таблице представлена подробная информация о функциях системных слов и о том, как они контролируются:

Системн. слова	Функция	Описание	Контроль
%SW0	Период сканирования контроллера (периодическая задача)	Изменяет период сканирования контроллера, определенный при конфигурации, через программу пользователя в Редакторе анимационных таблиц.	U
%SW6	Статус контроллера	Статус контроллера: 0 = NO CONFIG (не сконфигурирован) 2 = STOP (ОСТАНОВКА) 3 = RUN (РАБОТА) 4 = HALT (ОСТАНОВ)	S
%SW7	Состояние контроллера	● Бит [0]: Происходит резервное сохранение/восстановление: ● Установлен в 1, если происходит резервное сохранение/ восстановление, ● Установлен в 0, если резервное сохранение/восстановление завершено или не разрешено. ● Бит [1]: Конфигурация контроллера OK: ● Установлен в 1, если конфигурация OK. ● Бит [3..2]: биты статуса EEPROM: ● 00 = нет картриджа ● 01 = 32 Kb картридж EEPROM ● 10 = 64 Kb картридж EEPROM ● 11 = Зарезервировано для будущего использования ● Бит [4]: Приложение в RAM отличается от прилож. в EEPROM: ● Установлен в 1, если приложение в RAM отличается от пр. в EEPROM. ● Бит [5]: Приложение в RAM отличается от приложения в картридже: ● Установлен в 1, приложение в RAM отличается от пр. в картридже. ● Бит [6] не используется (статус 0) ● Бит [7]: Контроллер зарезервирован: ● Установлен в1, если зарезервирован. ● Бит [8]: Приложение в режиме Запись: ● Установлен в 1, если приложение защищено. ● Бит [9] не используется (статус 0) ● Бит [10]: Установлен второй последовательный порт: ● Установлен в 1, если установлен. ● Бит [11]: Тип второго последовательного порта: (0 = EIA RS-232, 1 = EIA RS-485): ● Установлен в = EIA RS-232 ● Установлен в 1 = EIA RS-485 ● Бит [12]: Действительное приложение во внутренней памяти: ● Установлен в 1, если приложение действительно. ● Бит [13] Действительное приложение в картридже: ● Установлен в 1, если приложение действительно. ● Бит [14] Действительное приложение в RAM: ● Установлен в 1, если приложение действительно. ● Бит [15]: Готов к выполнению: ● Установлен в 1, если готов к выполнению.	S

Системн. слова	Функция	Описание	Кон-троль
%SW11	Значение программного сторожевого таймера	Содержит максимальное значение сторожевого таймера. Значение (от 10 до 500 мс) определяется конфигурацией.	U
%SW17	Статус по умолчанию для операции с ПТ	При обнаружении ошибки в арифметической операции с ПТ, бит %S18 устанавливается в 1 и статус по умолчанию в %SW17 обновляется в соответствии со следующим кодом: ● Бит 0: Недействительная операция, результат не является числом (1.#NAN или -1.#NAN), ● Бит 1: Зарезервирован, ● Бит 2: Деление на 0, результат бесконечность (-1.#INF или 1.#INF), ● Бит 3: Результат по модулю больше +3.402824e+38, результат бесконечность (-1.#INF или 1.#INF).	S и U
%SW18-%SW19	100 мс абсолютный счетчик	Счетчик работает, используя два слова: ● **%SW18** отражает младшее значащее слово, ● **%SW19** отражает старшее значащее слово.	S и U
%SW30	Последнее время сканирования	Показывает время выполнения последнего цикла сканирования контроллера (в мс). **Примечание:** Это время соответствует времени между началом (опрос входов) и концом (обновление выходов) цикла сканирования.	S
%SW31	Максимальное время сканирования	Показывает время выполнения самого длинного цикла сканирования контроллера с момента последнего “холодного” пуска (в мс). **Примечание:** Это время соответствует времени между началом (опрос входов) и концом (обновление выходов) цикла сканирования.	S
%SW32	Минимальное время сканирования	Показывает время выполнения самого короткого цикла сканирования контроллера с момента последнего “холодного” пуска (в мс). **Примечание:** Это время соответствует времени между началом (опрос входов) и концом (обновление выходов) цикла сканирования.	S
%SW48	Число событий	Показывает, сколько событий произошло с момента последнего “холодного” пуска. **Примечание:** Устанавливается в 0 (после загрузки приложения и “холодного” пуска), увеличивается при каждом событии.	S

Системн. слова	Функция	Описание		Кон-троль
%SW49 %SW50 %SW51 %SW52 %SW53	Часы реального времени (RTC)	Функции RTC: слова содержат значения текущей даты и времени (в BCD):		S и U
		%SW49	xN День недели (N=1 для понедельника)	
		%SW50	00SS Секунды	
		%SW51	HHMM Час и минута	
		%SW52	MMDD Месяц и день	
		%SW53	CCYY Век и год	
		Эти слова контролируются системой, когда бит **%S50** в 0. Эти слова могут быть записаны программой пользователя или через терминал, когда бит установлен в 1. По спаду **%S50** внутренние RTC контроллера обновляются значениями, записанными в эти слова.		
%SW54 %SW55 %SW56 %SW57	Дата и время последней остановки	Системные слова, содержащие дату и время последнего сбоя питания или остановки контроллера (в BCD):		S
		%SW54	SS Секунды	
		%SW55	HHMM Час и минута	
		%SW56	MMDD Месяц и день	
		%SW57	CCYY Век и год	
%SW58	Код последней остановки	Отображает код причины последней остановки:		S
		1 =	Run/Stop входной фронт	
		2 =	Программная ошибка (выход за пределы периода сканирования)	
		3 =	Команда Stop	
		4 =	Отсутствие питания	
		5 =	Аппаратная ошибка	

Системн. слова	Функция	Описание			Кон-троль
%SW59	Корректирова-ние текущей даты	Корректирует текущую дату. Содержит два набора по 8 битов для корректирования текущей даты. Операция всегда выполняется по фронту бита. Это слово разрешается битом **%S59**.			U
		Увеличение	**Уменьшение**	**Параметр**	
		бит 0	бит 8	День недели	
		бит 1	бит 9	Секунды	
		бит 2	бит 10	Минуты	
		бит 3	бит 11	Часы	
		бит 4	бит 12	Дни	
		бит 5	бит 13	Месяцы	
		бит 6	бит 14	Годы	
		бит 7	бит 15	Века	
%SW60	Коррекция RTC	Значение коррекции RTC			U
%SW63	Код ошибки блока EXCH1	Код ошибки блока EXCH1: 0 - операция была успешной 1 - число байтов для пересылки слишком велико (> 128) 2 - таблица пересылки слишком маленькая 3 - таблица слов слишком маленькая 4 - таблица приема переполнена 5 - произошел тайм-аут 6 - пересылка 7 - неправильная команда в таблице 8 - выбранный порт не сконфигурирован/не доступен 9 - ошибка получения 10 - нельзя использовать%KW при получении 11 - смещение передачи больше, чем таблица передачи 12 - смещение приема больше, чем таблица приема 13 - контроллер остановил обработку EXCH			S
%SW64	Код ошибки блока EXCH2	Код ошибки EXCH2: См. %SW63.			S

Системн. слова	Функция	Описание	Кон-троль
%SW67	Функция и тип контроллера	Содержит следующую информацию: ● Биты типа контроллера [0 -11] ● 8B0 = TWDLCAA10DRF ● 8B1 = TWDLCAA16DRF ● 8B2 = TWDLMDA20DUK/DTK ● 8B3 = TWDLCAA24DRF ● 8B4 = TWDLMDA40DUK/DTK ● 8B6 = TWDLMDA20DRT ● Биты 12,13,14,15 не используются = 0	S

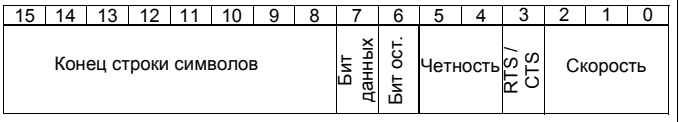
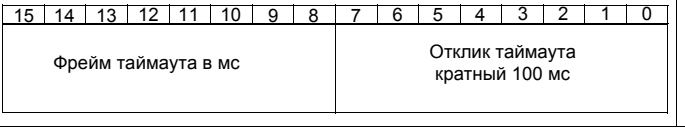
Системн. слова	Функция	Описание	Контроль
%SW73 и %SW74	AS-Interface состояние системы	● Бит [0]: Установлен в 1, если конфигурация OK. ● Бит [1]: Установлен в 1, если разрешен обмен данными. ● Бит [2]: Установлен в 1, если модуль в режиме Offline. ● Бит [3]: Установлен в 1, если прервана инструкция ASI_CMD. ● Бит [4]: Установлен в 1, ошибка в инструкции ASI_CMD.	S и U
%SW76 до%SW79	Счетчики вниз 1-4	Эти 4 слова служат 1 мс таймерами. они индивидуально уменьшаются системой каждую мс, если они имеют положительное значение. Это дает 4 счетчика вниз по мс, чо эквивалентно рабочему диапазону от 1 мс до 32767 мс. Установка бита 15 в 1 может остановить уменьшение.	S и U
%SW80	Статус базовых вх/вых	Бит [0] Каналы нормально функционируют (для всех каналов) Бит [1] Модуль инициализируется (или инициализируется информация всех каналов) Бит [2] Аппаратный сбой (сбой внешнего источника питания, общего для всех каналов) Бит [3] Ошибка конфигурации модуля Бит [4] Происходит преобразование данных входного канала 0 Бит [5] Происходит преобразование данных входного канала 1 Бит [6] Входная термопара канала 0 не сконфигурирована Бит [7] Входная термопара канала 1 не сконфигурирована Бит [8] Не используется Бит [9] Не используется Бит [10] Превышение диапазона канала 0 аналоговых вх. данных Бит [11] Превышение диапазона канала 1 аналоговых вх. данных Бит [12] Неправильный монтаж (канал 0 аналоговых вх. данных ниже диапазона, открыта токовая петля) Бит [13] Неправильный монтаж (канал 1 аналоговых вх. данных ниже диапазона, открыта токовая петля) Бит [14] Не используется Бит [15] Выходной канал не доступен	

Системн. слова	Функция	Описание	Контроль
%SW81	Статус модуля расширения вх/вых 1: такое же определение, как у %SW80		
%SW82	Статус модуля расширения вх/вых 2: такое же определение, как у %SW80		
%SW83	Статус модуля расширения вх/вых 3: такое же определение, как у %SW80		
%SW84	Статус модуля расширения вх/вых 4: такое же определение, как у %SW80		
%SW85	Статус модуля расширения вх/вых 5: такое же определение, как у %SW80		
%SW86	Статус модуля расширения вх/вых 6: такое же определение, как у %SW80		
%SW87	Статус модуля расширения вх/вых 7: такое же определение, как у %SW80		
%SW81 до%SW87	Статус модуля расширения		

%SW96	Команда и/или диагностика для функции сохранения/ восстановления прикладной программы и %MW.	• Бит [0]: Указывает, что слова памяти %MW должны быть сохранены в EEPROM:
		  • Установлен в 1, если сохранение требуется,
		  • Установлен в 0, если текущее сохранение не завершено.
		• Бит [1]: этот бит устанавливается программно-аппаратными средствами, чтобы указать, когда сохранение выполнено:
		  • Установлен в 1, если сохранение завершено,
		  • Установлен в 0, если есть запрос на новое сохранение.
		• Бит [2]: Ошибка сохранения, обратитесь к битам 8, 9, 10 и 14 для дальнейшей информации:
		  • Установлен в 1, если произошла ошибка,
		  • Установлен в 0, если есть запрос на новое сохранение.
		• Бит [6]: Установлен в 1, если контроллер содержит пустое приложение.
		• Бит [8]: Указывает, что число слов %MWs, указанное в %SW97больше, чем число %MWs, сконфигурированных в приложении:
		  • Установлен в 1, если обнаружена ошибка,
		• Бит [9]: Указывает, что число слов %MWs, указанное в %SW97больше, чем максимальное число %MWs, которое можно определит в любом приложении в TwidoSoft.
		  • Установлен в 1, если обнаружена ошибка,
		• Бит [10]:Различие между внутренней RAM и внутренней EEPROM (1 = да).
		  • Установлен в 1, если есть различия.
		• Бит [14]: Указывает, что произошла ошибка записи EEPROM:
		  • Установлен в 1, если обнаружена ошибка,
			S и U

Системн. слова	Функция	Описание	Контроль
%SW97	Команда или диагностика для функции сохранения/ восстановления	При сохранении слов памяти это значение отражает физическое количество %MW, сохраняемых в EEPROM. При восстановлении слов памяти это значение обновляется количеством слов памяти, восстанавливаемых в RAM. Для операции сохранения, когда это число установлено в 0, слова памяти не будут сохранены. Пользователь должен определить логику программы. Иначе, программа устанавливается в 0 в приложении контроллера, за исключением следующего случая: При “холодном” пуске это слово устанавливается в -1, если во внутренней Flash EEPROM нет файла сохраненных слов памяти **%MW**. В случае “холодного” пуска, если внутренняя Flash EEPROM содержит список слов памяти **%MW**, значение числа сохраненных в файле слов памяти должно быть установлено в **%SW97**.	S и U

Системн. слова	Функция	Описание	Контроль
%SW101 %SW102	Значения адресов портов Modbus	Когда бит %S101 или %S102 установлен в 1, Вы можете изменить адрес Modbus порта 1 или порта 2 соответственно. Адрес порта 1 - %SW101, адрес порта 2 - %SW102.	S

Системн. слова	Функция	Описание	Контроль
%SW103 %SW104	Конфигурация для использования протокола ASCII	Когда бит%S103 (Comm 1) или %S104 (Comm 2) установлен в 1, используется протокол ASCII. Системное слово %SW103 (Comm 1) или %SW104 (Comm 2) должно быть установлено в соответствии со следующими элементами:  ● Скорость: ● 0: 1200 бод, ● 1: 2400 бод, ● 2: 4800 бод, ● 3: 9600 бод, ● 4: 19200 бод, ● 5: 38400 бод. ● RTS/CTS: ● 0: не разрешен, ● 1: разрешен. ● Четность: ● 00: нет, ● 10: нечетные, ● 11: четные. ● Бит остановки: ● 0: 1 бит остановки, ● 1: 2 бита остановки. ● Биты данных: ● 0: 7 битов данных, ● 1: 8 битов данных.	S
%SW105 %SW106	Конфигурация для использования протокола ASCII	Когда бит %S103 (Comm 1) или %S104 (Comm 2) установлен в 1, используется протокол ASCII. Системное слово %SW105 (Comm 1) или %SW106 (Comm 2) должно быть установлено в соответствии со следующими элементами:  	S

Системн. слова	Функция	Описание	Контроль
%SW111	Статус дистанционной связи	Индикация: Бит 0 соответствует удаленному контроллеру 1, бит 1 удаленному контроллеру 2 и т.д. Биты с [0] по [6]: ● Установлен в 0 = отсутствует удаленный контроллер 1-7 ● Установлен в 1 = присутствует удаленный контроллер 1-7 Биты с [8] по [14]: ● Установлен в 0 = удаленный вх/вых определен на удаленном контроллере 1-7 ● Установлен в 1 = контроллер расширения определен на удаленном контроллере 1-7	S
%SW112	Код ошибки конфигурации/ функционирования дистанционной связи	00: успешная работа 01: обнаружен таймаут (slave) 02: обнаружена ошибка контрольной суммы (slave) 03: несоответствие конфигураций (slave) Устанавливается в 1 системой, должен быть сброшен пользователем.	S
%SW113	Конфигурация дистанционной связи	Индикация: Бит 0 соответствует удаленному контроллеру 1, бит 1 удаленному контроллеру 2 и т.д. Биты с [0] по [6]: ● Установлен в 0 = удаленный контроллер 1-7 не сконфигурирован ● Установлен в 1 = удаленный контроллер 1-7 сконфигурирован Биты с [8] по [14]: ● Установлен в 0 = удаленный вх/вых сконфигурирован как удаленный контроллер 1-7 ● Установлен в 1 = равноправный контроллер сконфигурирован как удаленный 1-7	S
%SW114	Разрешение блоков планировщика	Разрешает или запрещает работу блоков планировщика через программу пользователя или дисплей. Бит 0: 1 = разрешает блок планировщика #0 ... Бит 15: 1 = разрешает блок планировщика #15 первоначально все блоки планировщика разрешены. Если блоки сконфигурированы, значение по умолчанию =FFFF Если ни один блок не сконфигурирован, значение по умолчанию=0.	S and U

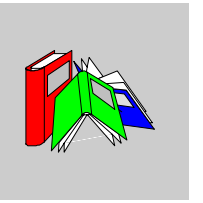
Системн. слова	Функция	Описание	Контроль
%SW118	Статусное слово базового контроллера	Показывает ошибки, определенные на master контроллере. Бит 9: 0 = Внешний сбой или сбой коммуникаций Бит 12: 0 = RTC не установлены Бит 13: 0 = Ошибка конфигурации (расширение вх/вых сконфигурировано, но отсутствует или ошибочно). Все остальные биты этого слова установлены 1 и зарезервированы. Для контроллера, у которого нет ошибок, значение этого слова FFFFh.	S
%SW120	Состояние модуля расширения вх/вых	Один бит на модуль. Адрес 0 = Бит 0 1 = Непорядок 0 = OK	S

Описание аббревиатур

Расшифровка аббревиатур:

Аббревиатура	Описание
S	Контролируется системой
U	Контролируется пользователем

Глоссарий



!

% Префикс, который идентифицирует внутренние адреса памяти в контроллере, которые используются для хранения значений переменных программы, констант, вх/вых и т.д.

A

Addresses **Адреса.** Внутренние регистры в контроллере используются для хранения значений переменных программы, констант, вх/вых, и т.д. Адреса идентифицируются префиксом символом процента (%) . Например, %I0.1 определяет адрес в пределах оперативной памяти контроллера, содержащей значение для входного канала 1.

Analog potentiometer **Аналоговый потенциометр.** Прикладываемое напряжение, которое может быть откорректировано и преобразовано в цифровое значение для использования прикладной программой.

Analyze program **Анализировать программу.** Команда, которая компилирует программу и проверяет ошибки в программе: синтаксис и ошибки структуры, символы без соответствующих адресов, недоступные ресурсы, используемые в программе, и если программа не соответствует в доступной памяти контроллера. Ошибки отображаются в Просмотрщике Программных Ошибок.

Animation table	**Анимационная таблица.** Таблица, созданная в пределах редактора языка или экрана оператора. Когда PC связан с контроллером, обеспечивает обзор переменных контроллера и позволяет форсировать значения при отладке. Может быть сохранена как отдельный файл с расширением .tat.
Animation Tables Editor	**Редактор анимационных таблиц.** Специализированное окно в прикладной программе TwidoSoft для просмотра и создания Анимационных Таблиц.
Application	**Приложение.** Прикладная программа TwidoSoft состоит из программы, данных конфигурации, символов и документации.
Application browser	**Браузер приложения.** Специализированное окно в TwidoSoft, которое отображает графическое древовидное представление прикладной программы. Обеспечивает удобное конфигурирование и просмотр прикладной программы.
Application file	**Файл приложения.** Приложения Twido хранятся как файл с расширением .twd.
ASCII	**ASCII.** (Американский Стандартный Код для Информационного Обмена) протокол связи для представления алфавитно-цифровых символов, особенно символов, рисунков и некоторых графических символов и управляющих символов.
Auto line validate	**Автоматическая проверка строк.** При вставке или изменении команд Списка, эта дополнительная настройка позволяет проверять строки программы сразу после ввода на наличие ошибок и нерешенных символов. Каждый элемент должен быть исправлен прежде, чем Вы можете выходить из строки. Выбирается, используя диалоговое окно Preferences.
Auto load	**Автозагрузка.** Особенность, которая всегда допускается и обеспечивает автоматическую передачу прикладной программы с резервного картриджа в оперативную память контроллера в случае потери или разрушения прикладной программы. При включении контроллера сравнивает прикладную программу, которая присутствует в оперативной памяти контроллера, с прикладной программой в дополнительном резервном картридже памяти (если установлен). Если есть различие, то копия в резервном картридже копируется во внутреннюю EEPROM контроллера. Если резервный картридж не установлен, то прикладная программа во внутренней EEPROM копируется в контроллер.

B

Backup — **Резервное хранение.** Команда, которая копирует прикладную программу в оперативной памяти контроллера во внутреннюю EEPROM контроллера и дополнительный резервный картридж памяти (если установлен).

C

Coil — **Катушка.** Элемент лестничной диаграммы, отображающий выход контроллера.

Cold start or restart — **Холодный запуск или перезапуск.** Запуск контроллером со всеми данными, инициализированными к значениям по умолчанию, и программой, начатой с начала, со всеми очищенными переменными. Все программные и аппаратные параметры настройки инициализированы. Холодная перезагрузка может быть вызвана загрузкой новой прикладной программы в оперативную память контроллера. Любой контроллер без аварийного батарейного питания всегда запускается через Холодный Запуск.

Comment lines — **Строки комментариев.** В программах Списка, комментарии могут быть введены в отдельные стоки от команд. Строки комментариев не имеют номера строки, и должны быть вставлены в пределах круглых скобок и звездочек типа: (*КОММЕНТАРИИ НАХОДЯТСЯ ЗДЕСЬ *).

Comments — **Комментарии.** Комментарии - это тексты, которые Вы вводите, чтобы документировать цель программы. Для программ лестничной логики введите до трех линий текста в ступени заголовка, чтобы описать цель ступени. Каждая линия может состоять из 1 - 64 символов. Для программ Списка введите текст в n не пронумерованных строк программы. Комментарии должны быть вставлены в пределах круглых скобок и звездочек типа: (*КОММЕНТАРИИ НАХОДЯТСЯ ЗДЕСЬ *).

Compact controller — **Компактный контроллер.** Тип контроллера Twido, который обеспечивает простую конфигурацию “все в одном “ с ограниченным расширением. Модульный - другой тип контроллера Twido.

Configuration editor — **Редактор конфигураций.** Специализированное окно TwidoSoft, использующееся для управления аппаратной и программной конфигурацией.

Constants **Константы.** Сконфигурированные значения, которые не могут изменяться исполняющейся программой.

Contact **Контакт.** Элемент лестничной диаграммы, отображающий вход контроллера.

Counter **Счетчик.** Функциональный блок, использующийся для подсчета событий (прямого и обратного).

Cross references **Перекрестные ссылки.** Генерирование списка операндов, символов, номеров строк/ступеней, и операторов, используемых в прикладной программе, для упрощения создания и управления прикладных программ.

Cross References Viewer **Просмотрщик перекрестных ссылок.** Специализированное приложение TwidoSoft для просмотра перекрестных ссылок.

D

Data variable **Переменная данных.** См. Переменная.

Date/Clock functions **Функции Даты/Часов.** Позволяют управлять событиями по месяцу, дню месяца и времени дня. См. Блоки планировщика.

Drum controller **Барабанный контроллер.** Функциональный блок, который работает подобно электромеханическому барабанному контроллеру с изменениями шагов, связанными с внешними событиями.

E

EEPROM Электрически Стираемая Программируемая Постоянная память. Twido имеет внутреннюю EEPROM и дополнительный внешний картридж EEPROM памяти.

Erase **Стереть.** Эта команда удаляет приложение из контроллера и имеет две опции:

- Стереть содержимое ОП контроллера, внутренней EEPROM контроллера и установленного опционального картриджа резервного хранения.
- Стереть содержимое только установленного опционального картриджа резервного хранения.

Executive loader **Исполнительный загрузчик.** 32-битное приложение Windows, использующееся для загрузки новой “вшитой” исполнительной программы для контроллера Twido .

Expansion bus **Шина расширения.** Модули расширения вх/вых соединяются с базовым контроллером при помощи этой шины.

Expansion I/O modules **Модули расширения вх/вых.** Опциональные модули расширения вх/вых способны добавить вх/вых контроллеру Twido. (Не все модели контроллеров допускают расширение).

F

Fast counters **Быстрые счетчики.** Функциональный блок, который обеспечивает более быстрый счет up/down, чем доступен с функциональным блоком счетчика. Быстрые счетчики могут считать с частотой до 5 КГц.

FIFO Первый вошел, первый вышел. Функциональный блок, используемый для операций очереди.

Firmware executive **Исполнительные программно-аппаратные средства.** Исполнительные программные средства - операционная система, которая выполняет Ваши прикладные программы и управляет работой контроллера.

Forcing **Форсирование.** Преднамеренная установка входов и выходов контроллера в значение 0 или 1, даже если действительные значения отличаются. Используется для отладки при анимации программы.

Function block **Функциональный блок.** Программный модуль входов и переменных, организованных, чтобы вычислить значения для выходов, основанных на определенной функции типа таймера или счетчика.

G

Grafcet Grafcet используется, чтобы представить функционирование последовательной операции в структурированной и графической форме. Это - аналитический метод, который делит любую последовательную систему управления на ряд шагов, с которыми связаны действия, переходы, и условия.

I

Init state — **Начальное состояние.** Операционное состояние TwidoSoft, которое отображено в строке состояния, когда TwidoSoft запущен или не имеет открытой прикладной программы.

Initialize — **Инициализировать.** Команда, которая устанавливает все значения данных в начальные состояния. Контроллер должен быть в Режиме Остановки или Ошибки.

Instance — **Образец.** Уникальный объект в программе, который принадлежит определенному типу функционального блока. Например, в формате %TMi таймера, i - номер, представляющий образец.

Instruction List language — **Язык Списка команд.** Программа, написанная на языке Списка команд (IL), составляется из ряда команд, выполняемых последовательно контроллером. Каждая команда состоит из номера строки, кода команды, и операнда.

L

Ladder editor — **Лестничный Редактор.** Специализированное окно TwidoSoft, использующееся для редактирования лестничной программы.

Ladder language — **Язык Лестничной логики.** Программа, написанная на языке Лестничной логики составлена из графического представления команд программы контроллера с символами для контактов, катушек, и блоков в наборы ступеней, выполняемых последовательно контроллером.

Ladder list rung — **Ступень Лестничной программы.** Отображает части программы на языке Списка, которые не обратимы к языку лестничной логики.

Latching input — **Запирающийся вход.** Поступающий импульс фиксируется и записывается для более поздней проверки прикладной программой.

LIFO — Последний вошел, первый вышел. Функциональный блок, используемый для операций стека.

List editor **Редактор Списка.** Простой редактор текстов программ использующийся для создания и редактирования программы на языке Списка.

M

Master controller **Master контроллер.** Контроллер Twido, сконфигурированный, чтобы быть Master в сети удаленной связи.

Memory cartridge **Картридж памяти.** Дополнительный резервный картридж памяти, который может использоваться, чтобы резервировать и восстановить прикладную программу (программа и данные конфигурации). Есть два доступные размера: 32 и 64 КБ.

Memory usage indicator **Индикатор использования памяти.** Часть строки состояния в главном окне TwidoSoft, которая отображает процент от полной памяти контроллера, используемой прикладной программой. Обеспечивает предупреждение, когда памяти мало.

Modbus Master-slave протокол связи, который позволяет одному единственному master запрашивать ответы от slaves.

Modular controller **Модульный контроллер.** Тип контроллера Twido, который предлагает гибкую конфигурацию со способностями расширения. Компактный - другой тип контроллера Twido.

Monitor state **Состояние монитора.** Операционное состояние TwidoSoft, которое отображено на Строке состояния, когда PC связан с контроллером в не записываемом режиме.

O

Offline operation **Автономная работа.** Операционный режим TwidoSoft, когда ПК не связан с контроллером и приложение в памяти ПК отличается от приложения в памяти контроллера. Вы создаете и развиваете приложение при Автономной работе.

Offline state **Автономное состояние.** Операционное состояние TwidoSoft, которое отображается в Строке Состояния, когда ПК не связан с контроллером.

Online operation **Оперативная работа.** Операционный режим TwidoSoft, когда ПК связан с контроллером и приложение в памяти ПК не отличается от приложения в памяти контроллера. Оперативная работа может использоваться для отладки приложения.

Online state **Оперативное состояние.** Операционное состояние TwidoSoft, которое отображается в строке состояния, когда ПК связан с контроллером.

Operand **Операнд.** Число, адрес, или символ, представляющий значение, которым программа может управлять в инструкции.

Operating states **Операционные состояния.** Указывают состояние TwidoSoft. Показываются в строке состояния. Есть четыре операционных состояния: Начальное, Автономное, Оперативное, и Контрольное.

Operator **Оператор.** Символ или код, определяющий действие, которое будет выполнено инструкцией.

P

PC **ПК.** Персональный компьютер.

Peer controller **Равноправный контроллер.** Контроллер Twido, сконфигурированный как slave в сети удаленной связи. Приложение может иметь доступ к локальным данным и данным расширения вх/вых, но данные вх/вых не могут быть переданы в Master контроллер. Программа, выполняющаяся в равноправном контроллере, передает информацию к Master контроллеру, используя сетевые слова (%INW и %QNW).

PLC **ПЛК.** Программируемый контроллер Twido. Существует два типа контроллеров: компактный и модульный.

PLS Генерация импульсов. Функциональный блок, который генерирует квадратные волновые сигналы с 50% высоким уровнем и 50% низким уровнем.

Preferences **Предпочтения.** Диалоговое окно с выборными опциями для настройки редакторов программ Списка и лестничных диаграмм.

Program errors viewer **Просмотрщик программных ошибок.** Специализированное окно TwidoSoft, использующееся для просмотра ошибок программы и предупреждений.

Programmable controller — **Программируемый контроллер.** Контроллер Twido. Существует два типа контроллеров: компактный и модульный.

Protection — **Защита.** Относится к двум различным типам защиты приложения: защита паролем, которая обеспечивает управление доступом, и защита приложения контроллера, которая предотвращает чтение и запись прикладной программы.

PWM — Широтно-импульсная модуляция. Функциональный блок, который генерирует прямоугольные волновые сигналы с переменным рабочим циклом, который может быть установлен программой.

R

RAM — **ОЗУ.** Оперативное запоминающее устройство. Приложения Twido загружаются во внутреннюю временную оперативную память для выполнения.

Real-time clock — **Часы реального времени.** Опция, которая будет отсчитывать время даже, когда контроллер не включен в течение ограниченного промежутка времени.

Reflex output — **Рефлексный выход.** В режиме счета, текущее значение очень быстрого счетчика (%VFC.V) измеряется по отношению к пороговым значениям, чтобы определить состояние этих выделенных выходов.

Registers — **Регистры.** Специальные регистры внутри контроллера, выделенные для функциональных блоков LIFO/FIFO.

Remote controller — **Удаленный контроллер.** Контроллер Twido, сконфигурированный для связи с Master контроллером в сети удаленной связи.

Remote link — **Удаленная связь.** Высокоскоростная шина master/slave, созданная для передачи малых объемов данных между Master контроллером и до семи удаленных контроллеров (slave). Существует два типа удаленных контроллеров, которые могут быть сконфигурированы для передачи данных Master контроллеру: Равноправный контроллер, который может передавать прикладные данные и контроллер удаленных вх/вых, который может передавать данные вх/вых. Сеть удаленной связи может состоять из смешения обоих типов.

Resource manager — **Менеджер ресурсов.** Компонент TwidoSoft, который контролирует требования памяти приложением во время программирования и конфигурирования, отслеживая ссылки на объекты программного обеспечения, сделанные приложением. Объект считается упомянутым в приложении, если он используется как операнд в инструкции списка или ступени лестницы. Показывает статусную информацию о проценте от полной используемой памяти и обеспечивает предупреждение, если памяти становится мало. См. Индикатор использования памяти.

Reversible instructions — **Обратимые инструкции.** Метод программирования, который позволяет просматривать инструкции и в виде списка инструкций, и в виде ступеней лестницы.

RTC — См. Часы реального времени.

RTU — Удаленный терминал. Протокол, использующий восемь битов, который используется для связи контроллера и ПК.

Run — **Запустить.** Команда, которая вызывает запуск прикладной программы в контроллере.

Rung — **Ступень.** Ступень расположена между двумя шинами питания в сетке и составлена из группы графических элементов, соединенных друг с другом горизонтальными и вертикальными связями. Максимальные размеры ступени - семь рядов и одиннадцать колонок.

Rung header — **Заголовок ступени.** Панель, которая появляется непосредственно над ступенью Лестницы и может использоваться, чтобы документировать цель ступени.

S

Scan — **Сканирование.** Контроллер сканирует программу и по существу исполняет три основных функции. Сначала, он читает входы и размещает эти значения в память. Затем, он выполняет прикладную программу по одной инструкции одновременно и сохраняет результаты в память. Наконец, он использует результаты, чтобы обновить выходы.

Scan mode **Режим сканирования.** Определяет, как контроллер просматривает программу. Есть два типа способов сканирования: Нормальный (Циклический), контроллер сканирует непрерывно, или Периодический, контроллера сканирует в течение выбранной длительности (диапазон 2 - 150 мс) перед началом другого просмотра.

Schedule blocks **Блоки планировщика.** Функциональный блок, использующийся для программирования функций Даты и Времени для управления событиями. Требует опции часов реального времени.

Step **Шаг.** Шаг Grafcet определяет состояние последовательной операции автоматизации.

Stop **Остановить.** Команда, которая вызывает остановку выполнения прикладной программы контроллером.

Symbol **Символ.** Символ - это строка максимум из 32 алфавитно-цифровых символов, первый из которых алфавитный. он позволяет персонализировать объект контроллера, чтобы обеспечить удобство сопровождения.

Symbol table **Таблица символов.** Таблица символов, использованных в приложении. Отображается в Редакторе Символов.

T

Threshold outputs **Выходы порога.** Катушки, которые управляются непосредственно очень быстрым счетчиком (%VFC) согласно настройкам, установленным в течение конфигурации.

Timer **Таймер.** Функциональный блок, использующийся для выбора промежутка времени для управления событием.

Twido Ряд контроллеров Schneider Electric , состоящий из двух типов контроллеров (компактный и модульный), модулей расширения для добавления вх/вых, опций, таких как часы реального времени, коммуникации, дисплей оператора и картриджи памяти для резервного хранения.

TwidoSoft 32-битное Windows-совместимое графическое инструментальное программное средство для конфигурирования и программирования контроллеров Twido.

U

Unresolved symbol **Неразрешимый символ.** Символ без адреса переменной.

V

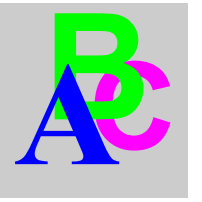
Variable **Переменная.** Элемент памяти, который может быть адресован и изменен программой.

Very fast counter **Очень быстрый счетчик.** Функциональный блок, который обеспечивает более быстрый счет, чем счетчики и быстрые счетчики. Очень быстрый счетчик может считать на частоте до 20 КГц.

W

Warm restart **Теплый перезапуск.** Запуск контроллера после отключения питания без изменения приложения. Контроллер возвращается в состояние, которое было до отключения питания и завершает цикл сканирования, который выполнялся. Все прикладные данные сохраняются. Эта возможность доступна только на модульных контроллерах.

Индекс



Символы

A

B

C

D

E

F

G

I

J

L

M

N

M

N

O

P

Q

R

Q

R

S

T

U

V

W

X