SIMATIC
S7-200 Примеры

№ №	Наименование примера	Тип CPU					Версия	Стр.
		210	212	214	215	216		
0	Содержание						3.2	
1	Прерывания по времени		x	x	x	x	3.3	6
2	Прерывания по событию		x	x	x	x	3.3	6
3	Аналоговые потенциометры			x	x	x	3.5	9
4	Быстрый счетчик			x	x	x	3.3	8
5	Режим свободного порта		x	x	x	x	3.3	5
6	Цикл For-Next		x	x	x	x	3.4	6
7	Широтно-импульсная модуляция			x	x	x	3.3	6
8	Часы реального времени			x	x	x	3.3	5
9	Обработка фронтов		x	x	x	x	3.3	4
10	Реверсивный переключатель (1)		x	x	x	x	3.3	6
11	Реверсивный переключатель (2)		x	x	x	x	3.3	9
12	Звезда-треугольник (1)		X	x	x	x	3.3	6
13	Звезда-треугольник (2)		x	x	x	x	3.3	7
14	Фазный ротор		x	x	x	x	3.3	8
15	КОПЛ-включение		x	x	x	x	3.3	5
16	Счетчик часов работы		x	x	x	x	3.4	5
17	Выключение по времени		x	x	x	x	3.3	4
18	Шаговая цепочка		x	x	x	x	3.3	7
19	Принтер		x	x	x	x	3.3	8
20	Считыватель штрих-кода		x	x	x	x	3.3	8
21	Регулирование лампы			x	x	x	3.3	4
22	Шаговый двигатель (1)			x	x	x	3.3	7
23	Шаговый двигатель (2)			x	x	x	3.4	12
24	Музыка			x	x	x	4.1	18
25	Периферийный интерфейсный модуль		x	x	x	x	3.4	13
26	Hayes модем		x	x	x	X	3.4	11
27	Удаленная периферия		x	x	x	x	3.4	25
29	Запись аналогового значения			x	x	x	3.3	6
30	Шаговый двигатель (3)			x	x	x	3.3	20
31	Таймеры		x	x	x	x	3.3	4
32	ПИД-регулятор			x	x	x	2.1	22
34	Аналоговый модуль		x	x	x	x	1.3	7

Copyright ©1996 by SIEMENS

35	Windows интерфейс		x	x	x	x	1.3	10
36	PT 100		x	x	x	x	1.3	10
37	Датчик RS 485		x	x	x	x	1.1	24
38	Масштабирование		x	x	x	x	1.1	11
39	PT 100 Линеаризация		x	x	x	x	1.1	13
40	Управление двигателем крана		x	x	x	x	1.1	8
41	Modbus RTU Slave			x	x	x	1.1	43
42	Замечания о сети PPI		x	x	x	x	1.1	12
44	HP 200LX Paimtop	x	x	x	x	x	1.1	4
45	Измерение температуры			x	x	x	1.1	8
46	Использование нескольк. термопар		x	x	x	x	1.1	3
47	RTD Замечания		x	x	x	x	1.1	4
48	Работа в сети			x	x	x	1.1	10
49	PPI модем		x	x	x	x	1.0	5
50	PTO Ramp			x	x	x	1.0	8
51	Использование операторного интерфейса TD 200		x	x	x	x	1.0	23
52	XMT и RCV				x	x	1.0	19
53	PID- команда				x	x	1.0	7
54	Демонстрация PID				x	x	1.0	20
55	Данные ASI - I\O		x	x	x	x	1.0	8
56	Profibus				x		1.0	16
57	Демонстрация Profibus				x		1.0	12

SIMATIC S7-200 Tips

1. General Applications

Tip No.	Topic	Ver	# Pgs
0	Master Listing of S7 Application Notes	3.2	2
16	How to track how long a device has been operating	3.3	5
17	Timing for Staircase Lighting	3.3	4
18	Step Sequence (Event Drum Timer)	3.3	7
21	Dimming a Light Bulb with the Integrated DC Pulse Outputs of the S7200	3.3	4
31	Creation of OFF-Delay, Pulse, and Extended Pulse Timers	3.3	4
33	Handling of the S7-200 Timers	1.3	6
43	Emulating an External Potentiometer	1.1	5
44	Using Micro/DOS with the HP Palmtop	1.1	4
46	Using Multiple Thermocouples with an S7-200 CPU	1.1	3
47	Using Multiple Resistance Temperature Detectors (RTDs) with an S7-200 CPU	1.1	4

2. Integrated Functions

Tip No.	Topic	Ver	# Pgs
1	Handling Timed Interrupts	3.3	6
2	Handling Event Interrupts	3.3	6
3	Analog-Potentiometer of the S7-200	3.5	9
4	How to use High Speed Counters of the S7-200	3.3	8
6	Different possibilities for setting bits or bytes	3.4	6
7	Handling Pulse Width Modulation with the S7-200	3.3	6
8	How to Read and Write to the Real Time Clock of the S7-200	3.3	5
9	Edge Detection of Input Signals	3.3	4
24	How to play music using the Integrated DC Pulse Outputs	4.1	18

Copyright ©1997 by SIEMENS

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

3. Communication

Tip No.	Topic	Ver	# Pgs
5	Simple Applications using Freeport Mode	3.3	5
19	Connection of a Parallel Printer with a S7-200 in Freeport Mode	3.3	8
20	Receiving of Information from a Bar Code Reader via Freeport Mode	3.3	8
25	Connection of a S7-200 to a TI505-System via Field Interface Module (FIM)	3.4	13
26	Application of Hayes Modem with a S7-200 via Freeport Mode	3.4	11
27	Connection of several S7-200 CPUs in a Remote I/O Network using Freeport Mode	3.4	25
35	Connection between S7-200 and PC: Reading from a Windows Application	1.3	10
37	Connecting an RS485 Encoder to the S7-200 PPI Interface	1.1	24
41	Modbus RTU Slave for S7-200	1.1	43
42	S7-200 PPI Network Notes	1.1	12
48	Using NETRs and NETWs with the S7-200	1.1	10
49	Using a Modem to Remotely Program and Monitor an S7-200 PLC through a PPI Interface	1.0	5
51	Using the TD200 Operator Interface with the S7-200 PLC	1.0	23
52	Using XMT and RCV instructions to communicate to an operator interface	1.0	19
55	Integration of ASI- I/O Data into the S7 200	1.0	8
56	General notes about Profibus and CPU 215	1.0	16
57	Interfacing a S7-300(Master) with a S7-200(Slave) by Profibus	1.0	12
58	Fault Alarm Recording with S7-200	1.0	15

4. Motor-Control

Tip No.	Topic	Ver	# Pgs
10	Reversible Motor Starter Circuits for Changing the Rotational Direction of Three-Phase A.C. Induction Motors	3.3	6
11	Reversible Motor Starter for Pole-Changeable Three-Phase Induction Motors with Rotational Direction Selection	3.3	9
12	Star-Delta Starter without Acknowledgment	3.3	6
13	Star-Delta Motor Starter with Starter Acknowledgment	3.3	7
14	Wound Rotor	3.3	8
15	Stator-Resistance Starting Circuit	3.3	5
28	Freeport Communication Interface to SIMOVERT Motor Drive	3.4	24
40	Using the S7-200 AC Inputs with 220VAC to Control the Three Phase Motor of a Crane	1.1	8

Copyright ©1997 by SIEMENS

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

5. Positioning

Tip No.	Topic	Ver	# Pgs
22	Triggering a Stepper Motor Drive, using Integrated DC Pulse Outputs with the S7-200 CPU	3.3	7
23	Controlled Positioning with a Stepper Motor Drive using Integrated Pulse Outputs	3.4	12
30	Control Positioning with Position Monitoring and Position Correction	3.3	20
50	Using a PTO Ramp with the S7-200 PLC	1.0	8

6. Regulatory Control

Tip No.	Topic	Ver	# Pgs
29	Using a HSC Input for Recording Analog Values by means of an Analog/Frequency (A/f) Converter	3.3	6
32	Handling PID on the S7-200	2.1	21
34	Handling Analog Inputs	1.3	7
36	Measuring and Monitoring Temperature with a PT100 Resistance Temperature Detector	1.3	10
38	Scaling Analog Values	1.1	11
39	Measuring and Monitoring Temperature with the S7-200 using Linearized Temperature Detector PT100	1.1	13
45	Temperature Measurement using Digital Inputs	1.1	8
53	Using the PID-Instruction	1.0	7
54	Demo using the PID-Instruction	1.0	20

Copyright ©1997 by SIEMENS

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

SIMATIC
S7-200 Примеры

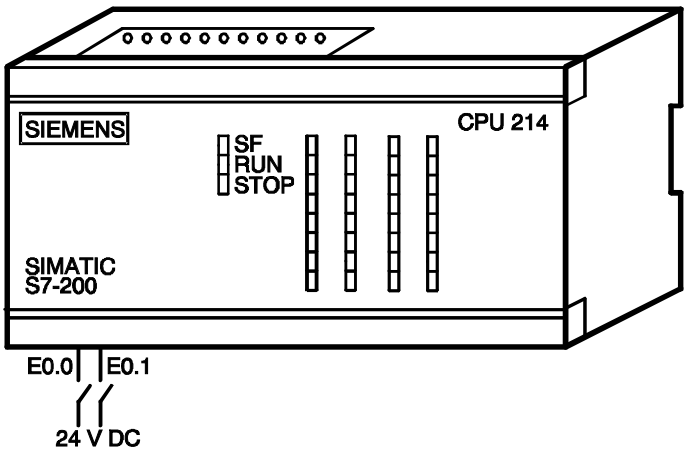
Группа	Пример к теме
2	Управление **прерываниями по времени**

Краткое описание

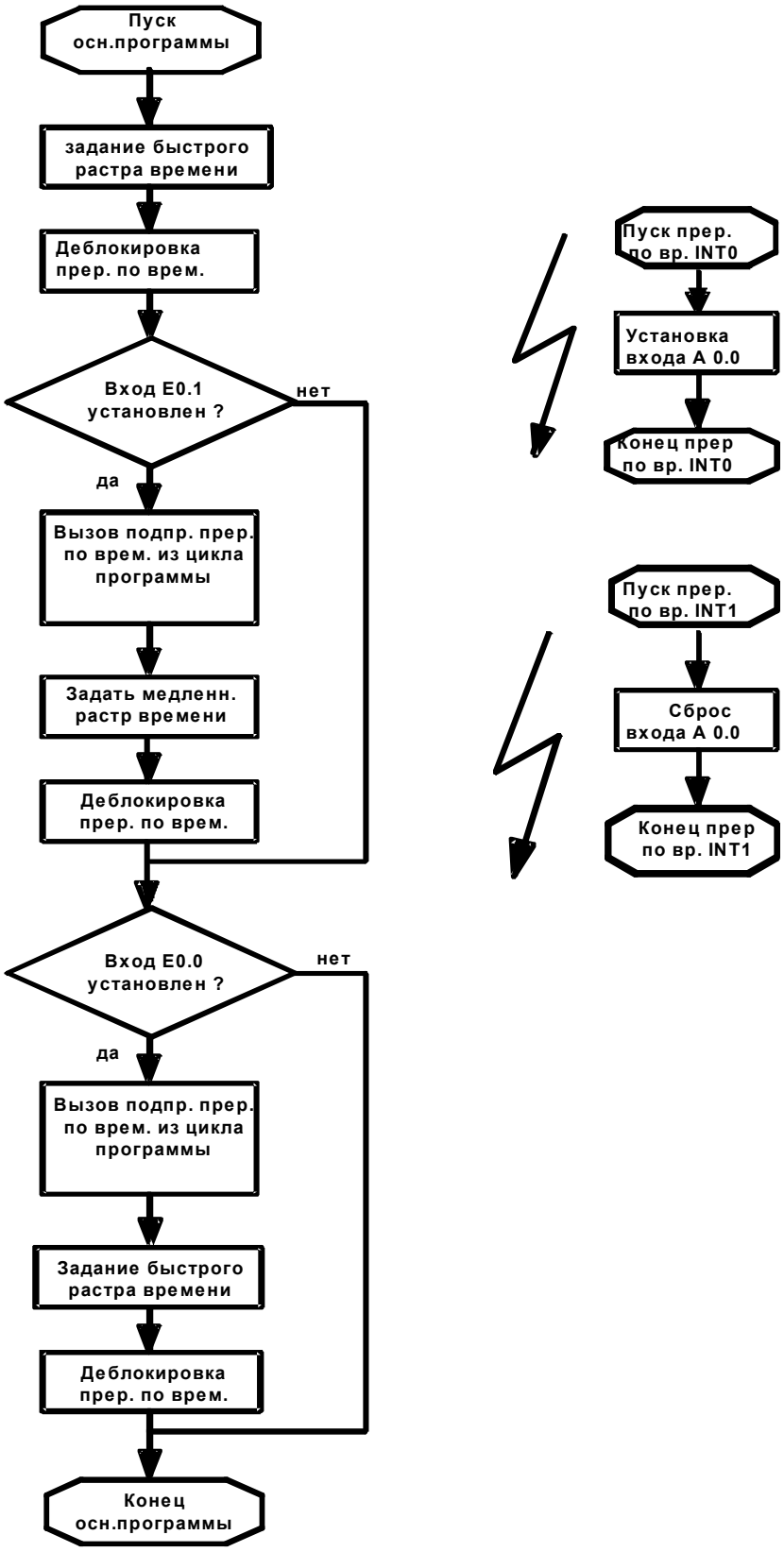
С помощью прерываний по времени создается тактовое мигание, причем частота мигания уполовинивается нажатием выключателя E0.1. Подача сигнала на вход 0.0 восстанавливает первоначальную частоту.

Этот пример призван пояснить общую работу с прерываниями по времени и изменением временного растра.

Схема включения



Структура программы



Описание программы вкл. листинг

В байт специальных меркеров SMB34 заносится растр времени, с которым вызывается прерывание по времени с номером 10 (первое прерывание по временем). В байт специальных меркеров SMB 35 растр времени, с которым вызывается прерывание по времени с номером 11 (второе прерывание по времени - поддерживается только CPU 214). Растр времени в обоих случаях будет определен с инкрементом 1мс. Минимальное допустимое значение растра времени составляет 5 мс, максимальное - 255 мс.
Программа состоит из следующих подпрограмм:

Main	Инициализация и задание времени
INT 0	Установка выхода A0.0
INT 1	Сброс выхода A0.0

Размер программы составляет 51 слово.

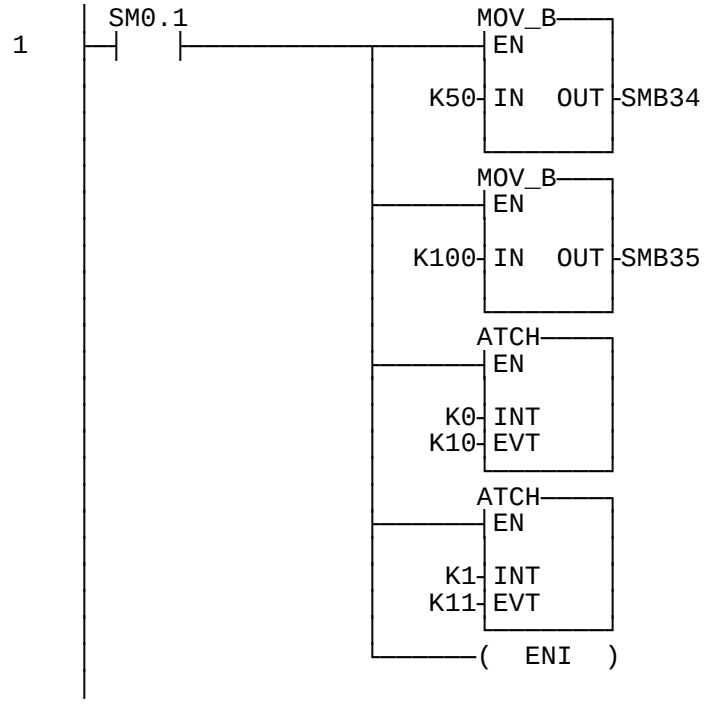
Подробнее о прерываниях управляемых по времени Вы найдете в главе 6.2 “Операции с прерываниями” в руководстве по программированию SIMATIC S7-200.

KOP (S7-MicroDOS)	AWL (TOOLITE2)

Основная программа

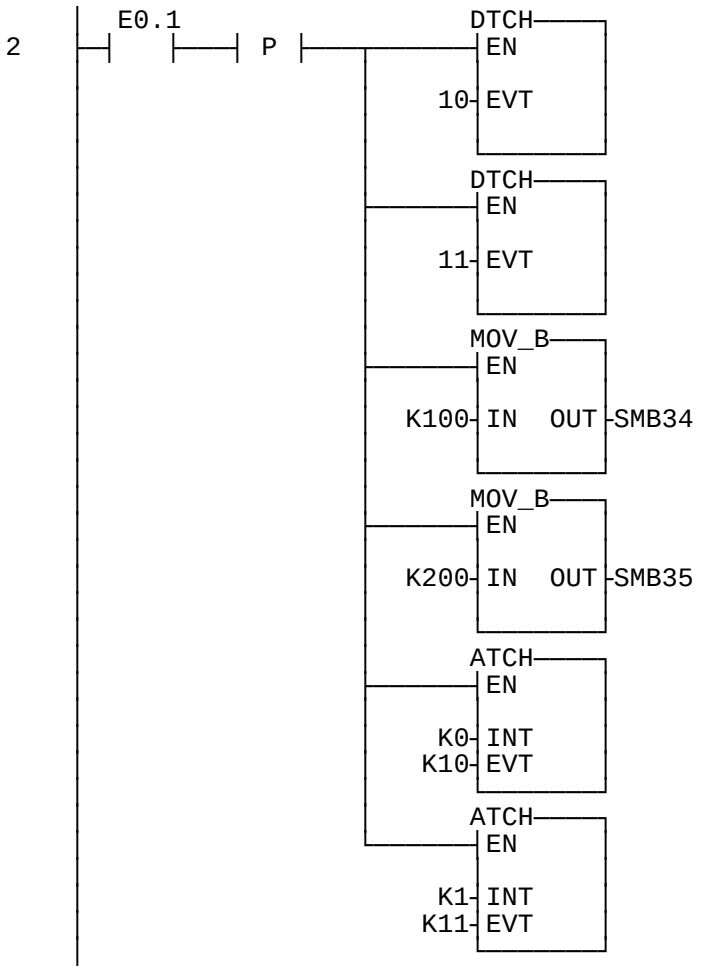
// TITEL = TIME-INT

// В этой первой части основной программы задается начальный растр времени
// и связываются два прерывания управляемых по времени.



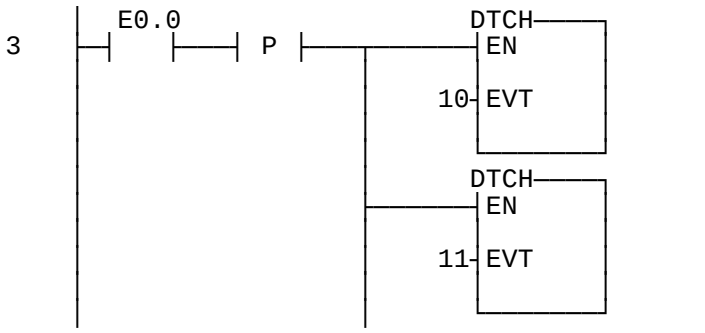

```
LD      SM0.1 // Обрабатывается только
              //  в первом цикле
MOVB    50,SMB34
              // Задание растра времени
              // для прерывания по
              // времени 0: 50 мс
MOVB    100,SMB35
              // Задание растра времени
              // для прерывания по
              // времени 1: 100 мс

ATCH    0,10  // Связь прерывания 10 с
              // подпрограммой
              // прерывания 0

ATCH    1,11  // Связь прерывания 10 с
              // подпрограммой
              // прерывания 1

ENI           // Деблокировка всех
              // прерываний
```

// По положительному фронту входа E0.1 растр времени прерывания по времени
// удваивается. Для выполнения этого нового задания необходимо
// разорвать связи между прерываниями и подпрограммами прерывания,
// так как в противном случае новые значения не будут восприняты. Разрыв
// связи производится командой DTCH.
// После задания нового растра времени, связи должны быть заново созданы
// командой ATCH.

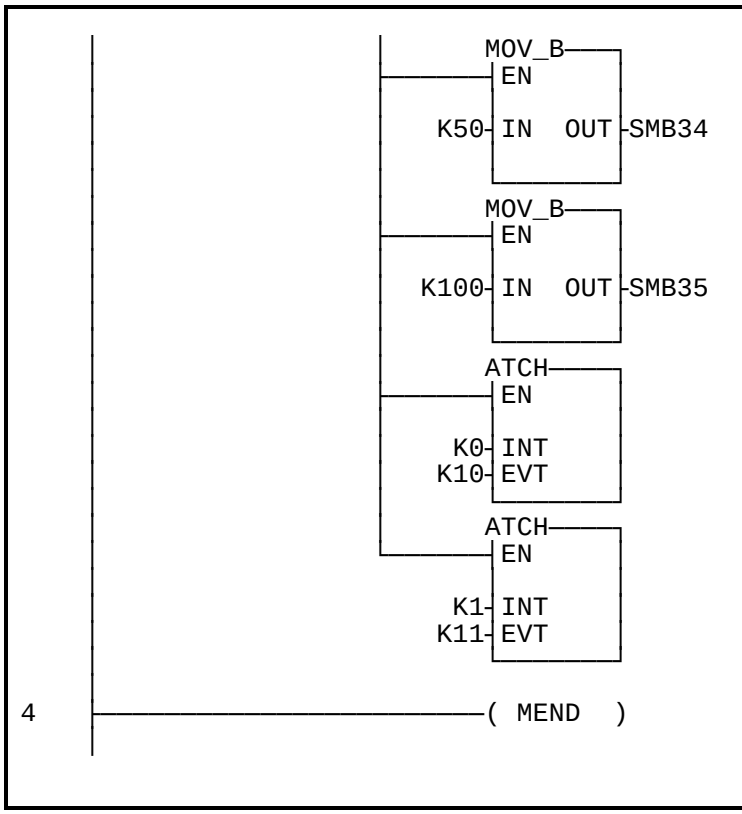



```
LD      E0.1    // Загрузка входа E0.1
EU              // и разыв связи с
DTCH    10      // прерыванием 0 по
                // положительному фронту
                // E0.1

DTCH    11      // Разрыв связи с
                // прерыванием 1

MOVB    100,SMB34
                // Задание нового растра
                // времени для
прерывания
                // по времени: 100 мс

MOVB    200,SMB35
                // Задание нового растра
                // времени для
прерывания
                // по времени: 200 мс

ATCH    0,10    // Восстановление
                // связи

ATCH    1,11    // Восстановление
                // связи
```

// По положительному фронту входа E0.0 будет восстановлена старая тактовая
// частота.



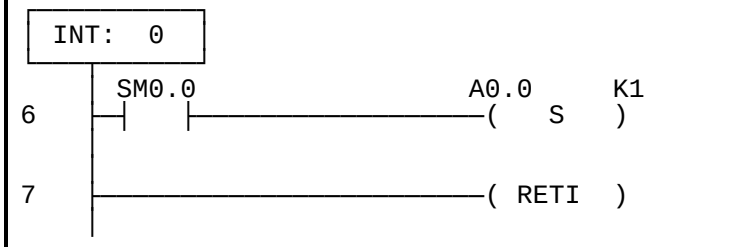

```
LD      E0.0    // Положительный фронт
EU              // ... фронта E0.0
DTCH    10      // разрывает связь с
                // прерыванием по
времени                         // 10

DTCH    11      // Прерывание связи с
                // прерыванием по
времени                         // 11
```




```
MOVB   50,SMB34
               // Время интервала
               // прерывания 0: 50 мс

MOVB   100,SMB35
               // Время интервала
               // прерывания 1: 100 мс

ATCH   0,10    // Деблокировка
               // прерывания по времени
0

ATCH   1,11    // Деблокировка
               // прерывания по времени
1

MEND           // Конец
               // основной программы
```

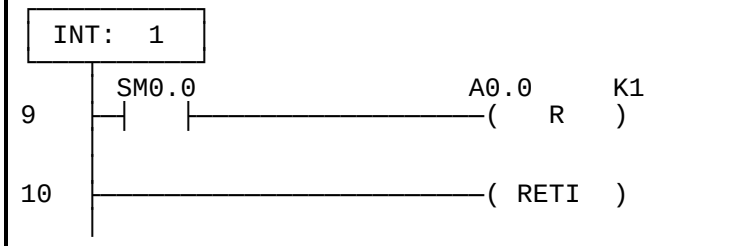
Подпрограммы прерываний

```
// ************************************************************
// При  вызове подпрограммы прерывания 0 устанавливается выход A0.0.
```




```
INT    0       // Подпрограмма
               // прерывания 0
LD     SM0.0 // устанавливает
S      A0.0,1  // выход A0.0 в 1

RETI           // Конец подпрограммы
               // прерывания 0
```

```
// ************************************************************
// При  вызове подпрограммы прерывания 1 выход A0.0 сбрасывается.
// Так как растр времени для вызова подпрограммы прерывания 1 в два раза длиннее, чем
// растр времени для вызова подпрограммы прерывания 0, возникает такт мигания.
```




```
INT    1       // Подпрограмма
               // прерывания 1
LD     SM0.0 // сбрасывает
R      A0.0,1  // выход A0.0

RETI           // Конец подпрограммы
               // прерывания 0
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL :

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 станет K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

Copyright ©1996 by SIEMENS

SIMATIC
S7-200 Примеры

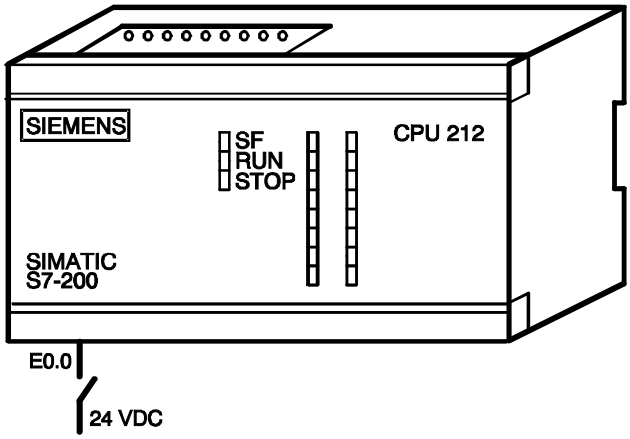
Группа	Пример к теме
2	Управление **подпрограммами прерываний ввода/вывода**

Краткое описание

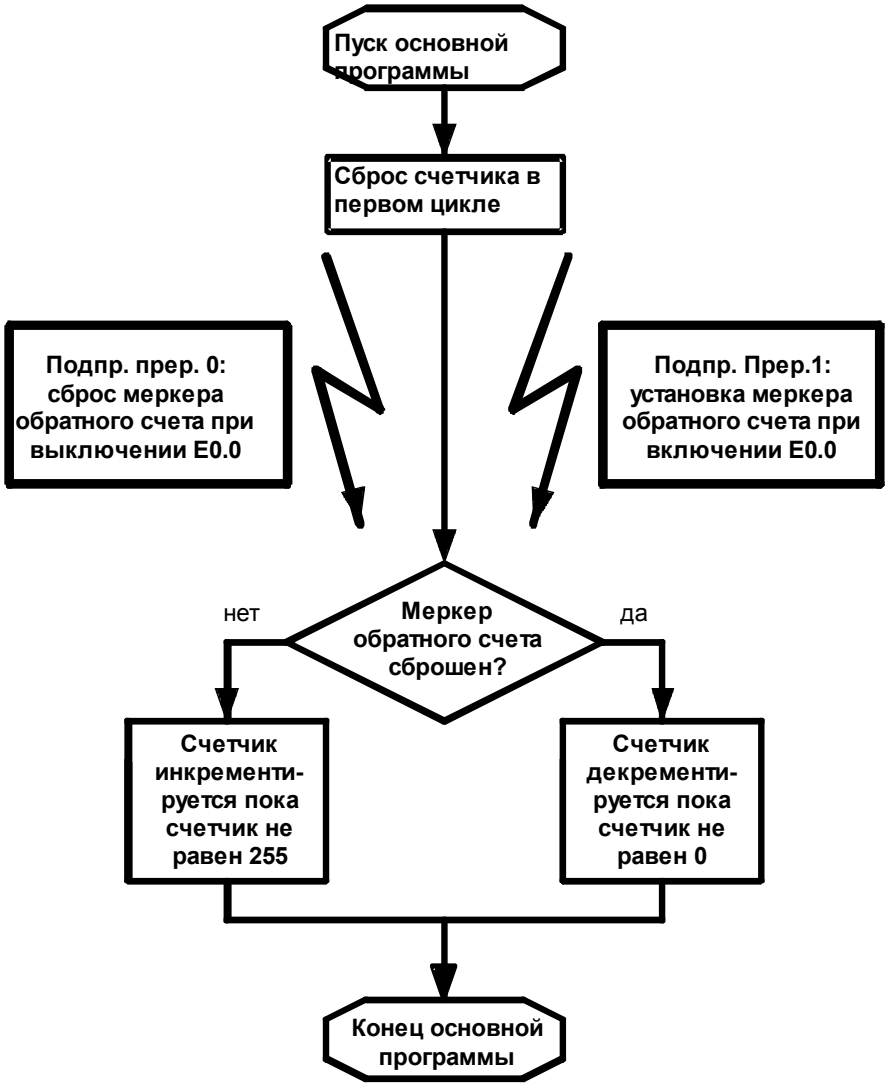
Эта программа для SIMATIC S7-212 и S7-214 считает от 0 до 255 в зависимости от входа E0.0: Если он установлен, то программа считает обратно, а если нет, то программа считает прямо.

При переключении этого входа выполняется подпрограмма прерывания ввода/вывода, которая устанавливает или сбрасывает меркерный бит обратного счета M0.0.

Схема включения



Структура программы



Описание программы вкл. листинг

Эта программа является примером подпрограммы прерывания ввода/вывода. Счетчик считает от 0 до 255. Если вход E0.0 не установлен, то идет прямой счет. Если вход E0.0 установлен, то идет обратный счет.

Стуктура программы:

Main	Инициализация и подпрограмма счета	
INT 0	прямой счет	- вход E0.0 не установлен
INT 1	обратный счет	- вход E0.0 установлен

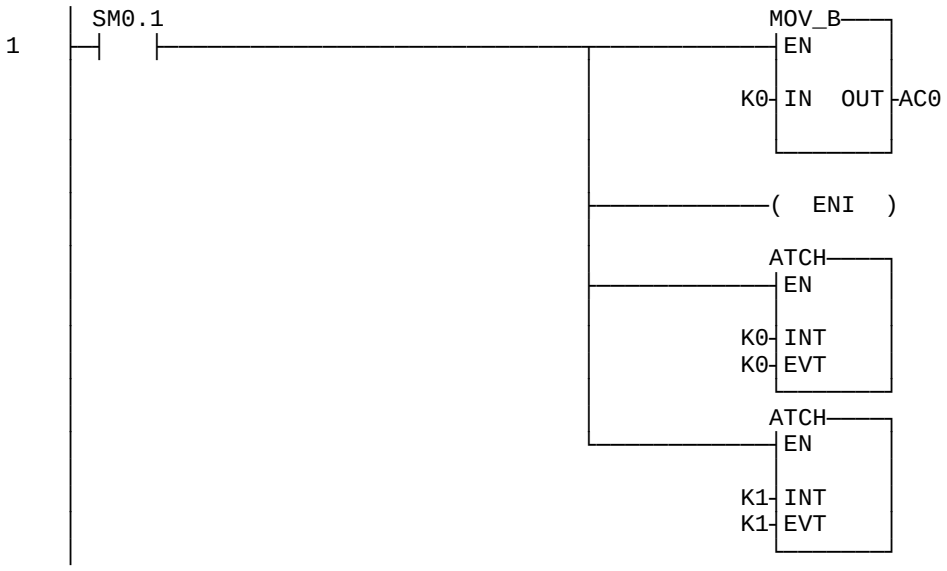
Размер программы составляет 32 слова.

Подробнее о прерываниях ввода/вывода Вы найдете в главе 6.2 “Операции с прерываниями” в руководстве по программированию SIMATIC S7-200.

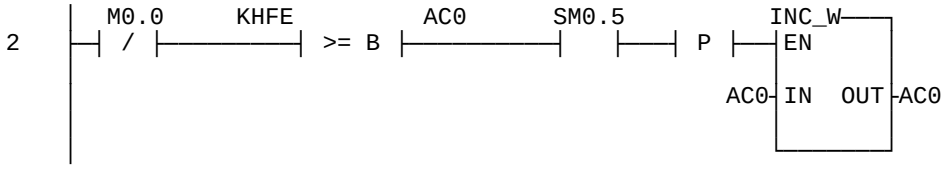
KOP (S7-MicroDOS)	AWL (TOOLITE2)

Основная программа

```
// TITLE=EVENT-INT
// Основная программа состоит из подпрограмм инициализации и счета.
// При 0 или 255 процесс счета завершается.
// Меркерный бит M0.0 (меркер обратного счета) отвечает за направление счета.
// Включение входа E0.0подпрограмма прерывания ввода/вывода INT 0 выключается.
// Вэтой подпрограмме устанавливается меркер обратного счета M0.0.
// При этом основная программа ведет обратный счет.
// Подпрограмма прерывания по событию INT 1 выполняется, если вход E0.0
// выключается. При этом меркер обратного счета M0.0 сбрасывается и
// программа вновь ведет прямой счет.
```




```
LD      SM0.1       // Бит инициализации - только в 1 в первом рабочем цикле
MOVB    0,AC0       // Установка аккумулятора счетчика в 0
ENI                 // Разрешение для подпрограмм прерываний
ATCH    0,0         // Подпрограма по событию для положительного фронта входа E0.0
ATCH    1,1         // Подпрограма по событию для отрицательного фронта входа E0.0
```




```
LDN     M0.0        // Если меркер обратного счета не установлен
UB>=    16#FE,AC0   // ... и текущее значение счетчика меньше или равно 254
U       SM0.5       // ... и импульс 0.5 секунды
EU                  // ... и положительный фронт импульса,
INCW    AC0         // то содержимое аккумулятора сетчика увеличивается на 1
```

```
3   |  M0.0      KH1          AC0      SM0.5          DEC_W
    |--|   |-----------| <= B |-----------|   |----| P |---| EN
    |                                                  |
    |                                              AC0-| IN  OUT |-AC0
    |                                                  |
    |                                                  |
```

```
LD      M0.0          // Если меркер обратного счета установлен
UB<=    16#1,AC0      // ... и текущее значение счетчика больше или равно 1
U       SM0.5         // ... и импульс 0.5 секунды
EU                    // ... и положительный фронт импульса,
DECW    AC0           // то содержимое аккумулятора сетчика уменьшается на 1
```

```
    | SM0.0                                          MOV_B
4   |--|   |-------------------------------------------| EN
    |                                                  |
    |                                              AC0-| IN  OUT |-AB0
    |                                                  |
    |                                                  |
5   |-------------------------------------------------( MEND  )
    |
```

```
LD      SM0.0         // Всегда в единице
MOVB    AC0,AB0       // Индикация текущего значения счетчика на выходах
MEND                  // Конец основной программы
```

Подпрограммы прерываний

```
// Подпрограмма прерывания по событию 0 устанавливает меркерный бит M0.0 в 0.
// В  этом случае программа ведет обратный счет.
```

```
 INT:  0
    | SM0.0                                          M0.0     K1
7   |--|   |-------------------------------------------(   S   )
    |
8   |-------------------------------------------------( RETI  )
    |
```

```
INT     0             // Подпрограмма по событию ведет обратный счет
LD      SM0.0
S       M0.0,1        // Установка меркера обратного счета
RETI                  // Конец подпрограммы прерываний
```

```
// Подпрограмма прерывания по событию 1 устанавливает меркерный бит M0.0 в 1.
// В  этом случае программа ведет прямой счет.
```

```
 INT:  1
    |
    | SM0.0                                          M0.0     K1
10  |-|   |-----------------------------------------(   R   )
    |
    |
11  |-----------------------------------------------( RETI  )
    |
```

INT	**1**	**// Подпрограмма по событию ведет прямой счет**
R	**M0.0,1**	**// Сброс меркера обратного счета**
RETI		**// Конец подпрограммы прерываний**

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ → ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

Copyright ©1996 by SIEMENS

SIMATIC
S7-200 Примеры

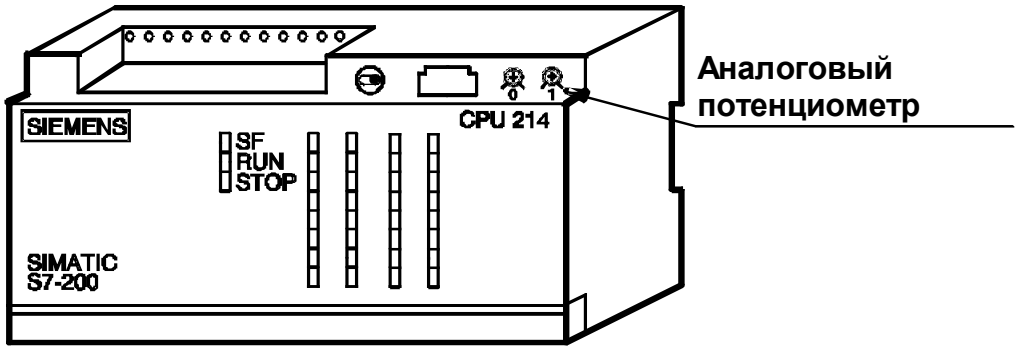
Группа	Пример к теме
3	Аналоговые потенциометры S7-200 CPU 214

Краткое описание

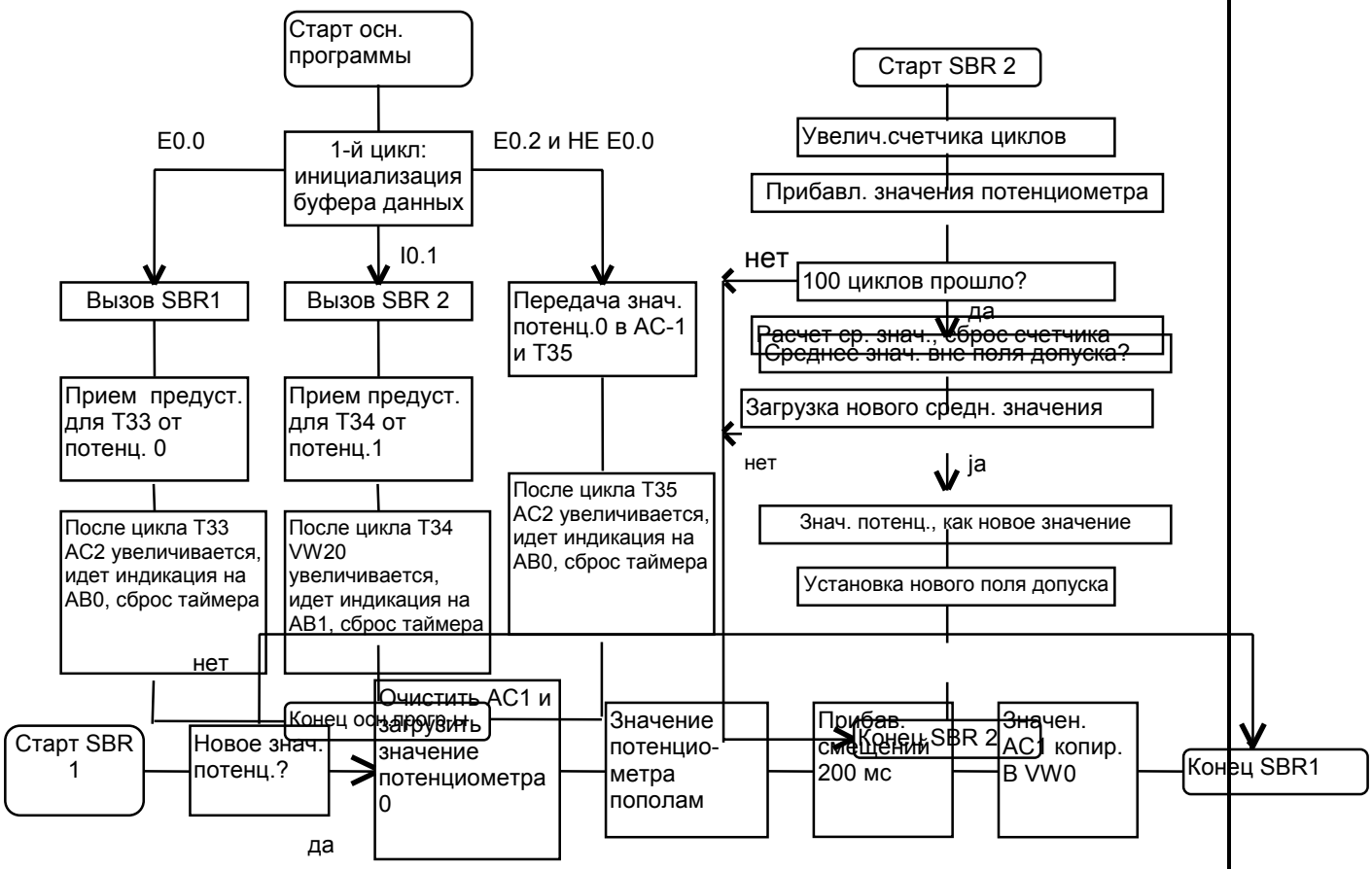
Этот пример поясняет применение аналоговых потенциометров SIMATIC CPU 214. Положение аналоговых потенциометров преобразуется в цифровое значение между 0 и 255 и заносится в два байта специальных, причем SMB28 содержит значения потенциометра 1, а SMB29 значение потенциометра 2.

Для установки потенциометров не требуется отвертки. Этот пример применения показывает три различных варианта установки таймера с помощью аналоговых потенциометров.

Схема включения



Структура программы



Описание программы включая листинг

Метод 1:
В качестве первого варианта рассмотрим применение аналогового потенциометра 0 для точной установки таймера. Грубая или предварительная установка (здесь 200мс) производится в программе. Потенциометр может быть при этом использован для дальнейшей установки нужного значения до ,примерно, 1,48 секунды. После каждого цикла таймера будут выполнятся операции подпрограммы 1: Значение потенциометра 0, содержащееся в SMB28, будет загружено в AC1, разделено пополам и прибавлено к 200мс-предустановке. В основной программе число циклов таймера в AC2 будет увеличено на 1 и индицировано на выходном байте AB0.

Метод 2:
Второй метод позволяет сложить и усреднить в AC3 значения потенциометра 1 в 100

следующих друг за другом циклах программы. В первом цикле запоминаются нижнее граничное в VW14 и верхнее граничное в VW16 значения. Если усредненное значение ниже нижней границы или выше верхней границы, то это значение заносится в VW12 и копируется в VW14, VW16 и VW18.

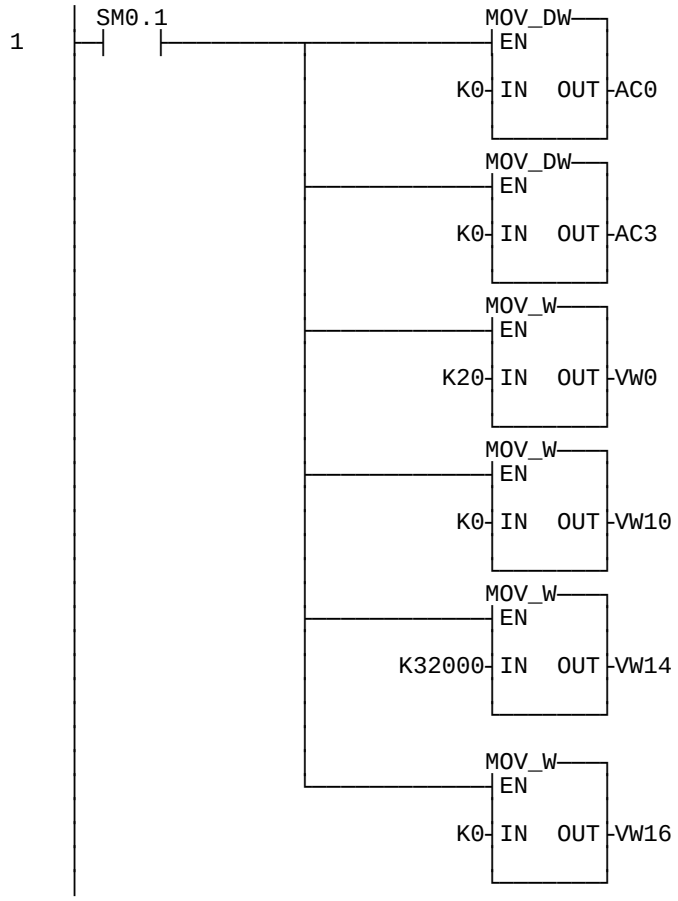
Новые верхняя и нижняя границы задаются прибавлением или вычитанием 3мс к усредненному значению. Среднее значение потенциометра, сохраненное в VW18, будет передано в основную программу в таймер T34. Здесь считаются циклы таймера и индицируются затем на выходном байте AB1. Этот метод создает путем фильтрации значений потенциометра относительно постоянную установку таймера от 0 до 2,55 секунд.

Метод 3:
Метод 3 это прямая загрузка значения потенциометра в качестве предустановки для таймера T35. Здесь число циклов таймера будет индицироваться на выходном байте AB0. Установка таймера может быть из менена в любом цикле.

Размер программы составляет 112 байт.

KOP (S7-MicroDOS)	AWL (TOOLITE2)

Основная программа




```
LD      SM0.1          // В первом цикле
MOVD    0,AC0          // сброс аккумулятора

MOVD    0,AC3

MOVW    20, VW0        // Занесение значения 20
                       // в слово данных VW0

MOVW    0, VW10        // Занесение значения 0
                       // в слово данных VW10

MOVW    32000, VW14    //Сброс нижней границы

MOVW    0, VW16        //Сброс верхней границы
```

// Метод 1

```
2  E0.0                          TON——T33
                                 IN
                          VW0-   PT
                                   1
                          ( CALL )

3  T33                           INC_W
                                 EN
                          AC2-   IN   OUT-AC2
                                 MOV_B
                                 EN
                          AC2-   IN   OUT-AB0
                          T33      K1
                          (   R   )
```

LD	E0.0	// Вход E0.0 для метода // Opt 1
TON	T33, VW0	// Установка для таймера // берется от потенц. 0
CALL	1	// Вызов // подпрограммы 1
LD	T33	// Счет циклов таймера
INCW	AC2	// в аккумуляторе AC2
MOVB	AC2, AB0	// Индикация байта // с наименьшим значением // на выходном байте 0
R	T33, 1	// Сброс битов таймера // T33

// Метод 2

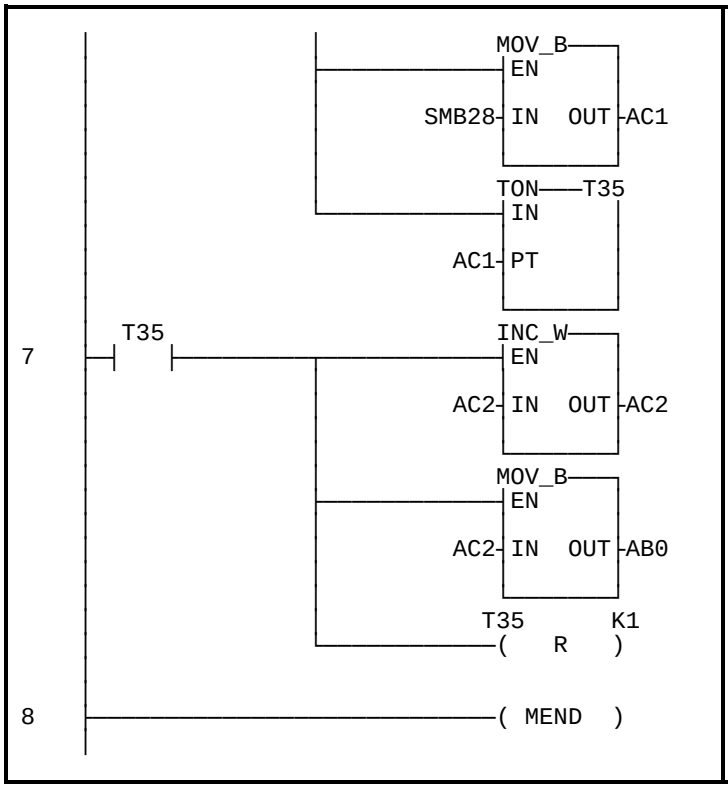
```
4  E0.1                            2
                          ( CALL )
                                 TON——T34
                                 IN
                          VW18-  PT

5  T34                           INC_W
                                 EN
                          VW20-  IN   OUT-VW20
                                 MOV_B
                                 EN
                          VB21-  IN   OUT-AB1
                          T34      K1
                          (   R   )
```

LD	E0.1	// Вход E0.1 //демонстрирует метод 2
CALL	2	// Вызов // подпрограммы 2
TON	T34, VW18	// Установка для таймера // берется от потенц. 1
LD	T34	// Счет циклов таймера
INCW	VW20	// в VW20
MOVB	VB21, AB1	// Индикация байта с // наименьшим значением // на выходном байте 1
R	T34, 1	// Сброс битов таймера // T34

// Метод 3

```
6  E0.2    E0.0                  MOV_W
                                 EN
                          K0-    IN   OUT-AC1
```

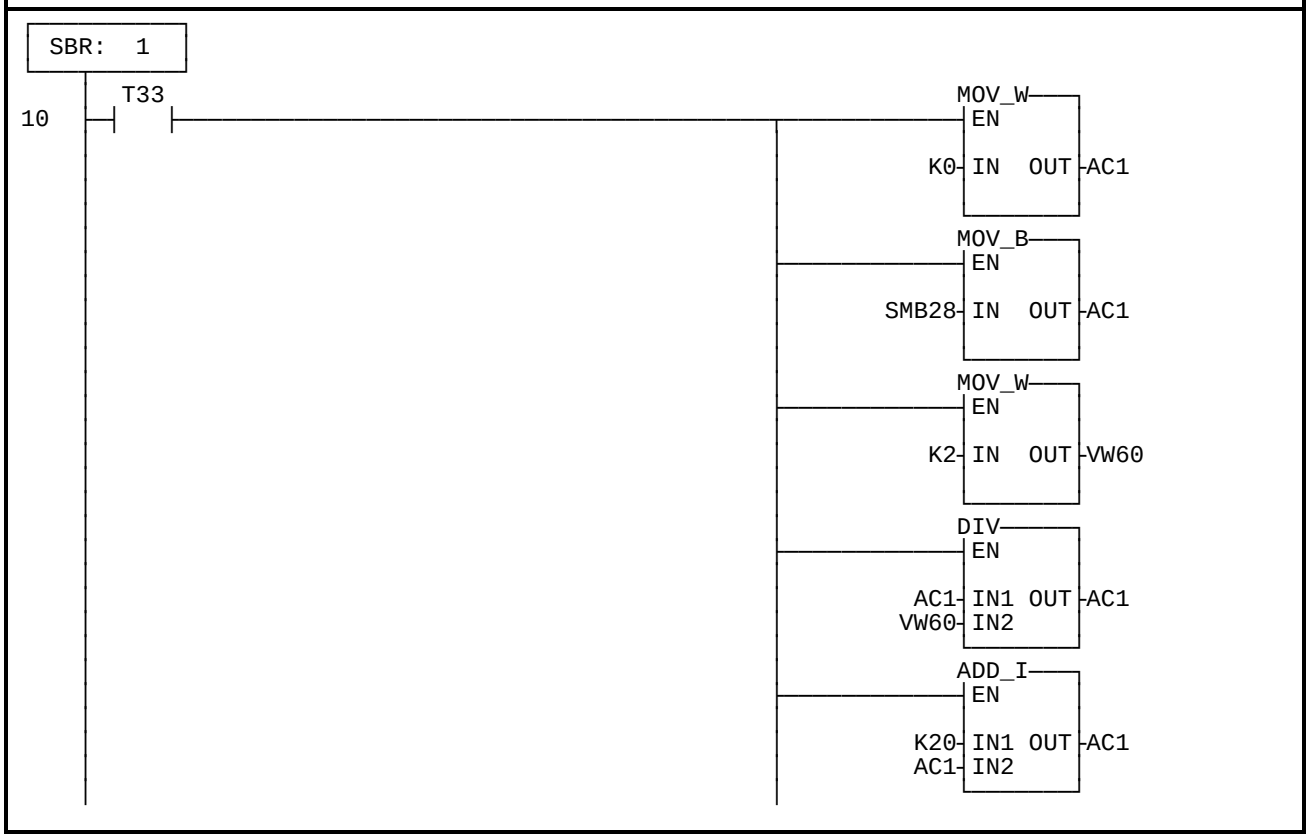
LD	E0.2	// Вход E0.2 запускает // метод 3
UN	E0.0	// если метод 1 не // используется
MOVW	0, AC1	// очищается // аккумулятор AC1



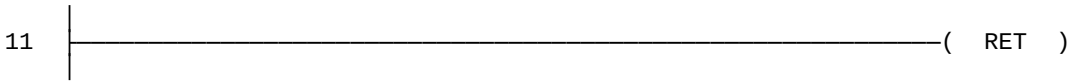
MOVB	**SMB28, AC1**	**// Прием значения // от потенциометра 0 в // аккумулятор AC1**
TON	**T35, AC1**	**// и загрузка значения //аккумулятора в // таймер T35**
LD	**T35**	**// Счет циклов таймера // в аккумуляторе**
INCW	**AC2**	**// AC2**
MOVB	**AC2, AB0**	**// Индикация байта с //наименьшим значением // на выходном байте 0**
R	**T35, 1**	**// Сброс битов // таймера T35**
MEND		**// Конец // основной программы**

Подпрограммы

// Подпрограмма SBR 1:
// Уменьшение значения потенциометра 0 включая сложение с
// предустановленным значением.

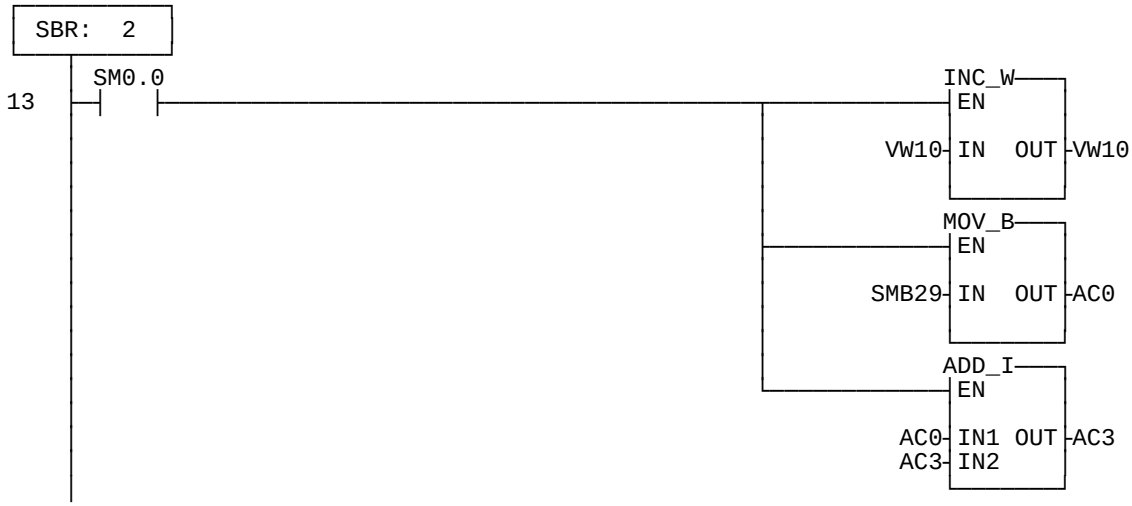


```
SBR     1
LD      T33             // После выполнения цикла таймера
MOVW    0, AC1          // аккумулятор AC1 сбрасывается
MOVB    SMB28, AC1      // и значение потенциометра 0 копируется в AC1
MOVW    2, VW60
DIV     VW60, AC1       // Принятое значение потенциометра делится пополам
+I      20, AC1         // и прибавляется смещение 200 мс
MOVW    AC1, VW0        // Значение AC1 копируется в VW0
```

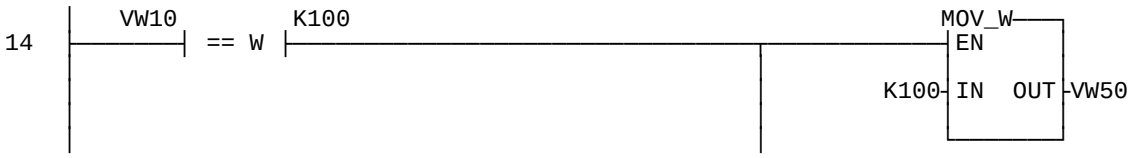


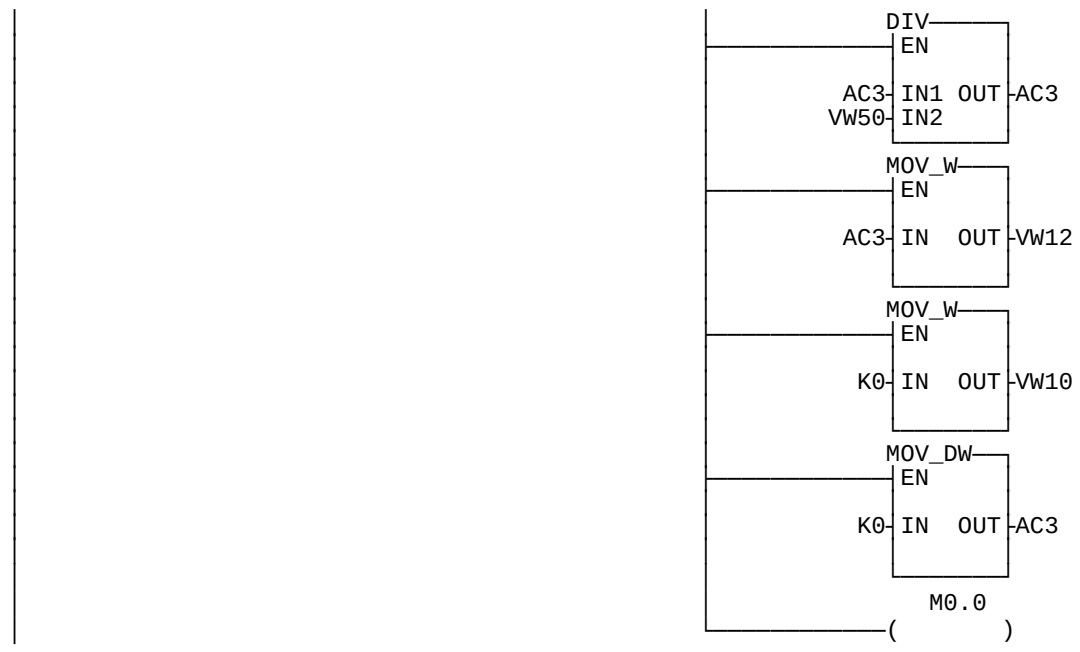

```
RET     // Возврат в основную программу
```

```
// Подпрограмма SBR 2:
// Определение среднего значения потенциометра 1 по 100 циклам программы
```

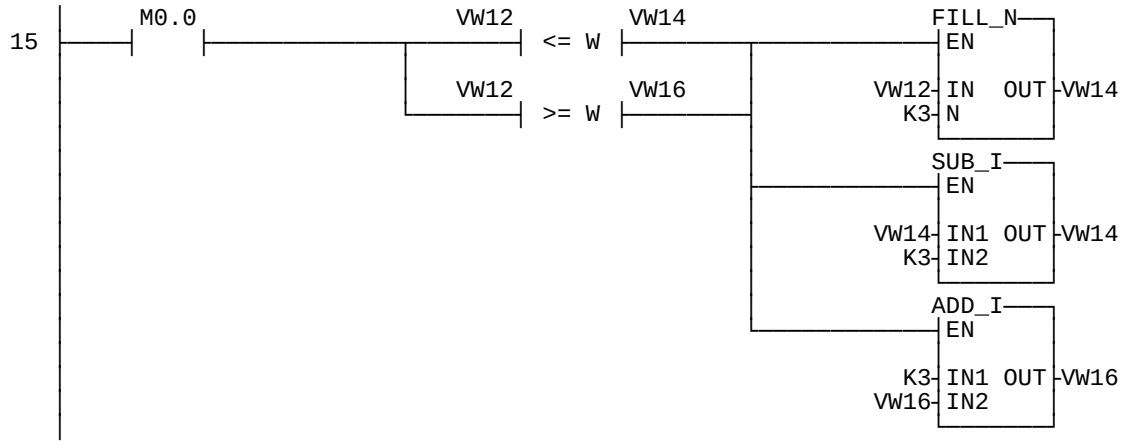



```
SBR     2
LD      SM0.0           // В каждом цикле
INCW    VW10            // счетчик циклов увеличивается на 1
MOVB    SMB29, AC0      // Значение потенциометра 1 заносится в AC0
+I      AC0, AC3        // Прибавляется к общему значению в AC3
```






```
LDW=    VW10,100        // После 100 циклов
MOVW    100, VW50
DIV     VW50, AC3       // Значение для образования среднего значения делится на 100
MOVW    AC3, VW12       // Среднее значение заносится в VW12
MOVW    0, VW10         // Сброс счетчика циклов
MOVD    0, AC3          // Сброс памяти общего значения
=       M0.0
```




```
LD      M0.0
LDW<=   VW12, VW14      // и новое среднее значение вне поля допуска
OW>=    VW12, VW16
ALD
FILL    VW12, VW14, 3   // Новое среднее значение копируется в VW14, VW16 и VW18
-I      3, VW14         // Установка новой нижней границы (Среднее значение - 3мс)
+I      3, VW16         // Установка новой верхней границы (Среднее значение + 3мс)
```

```
16 |------------------------------------------------------------( RET )
```

RET	**// Возврат в основную программу**

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ → ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

Copyright ©1996 by SIEMENS

SIMATIC
S7-200 Примеры

Группа	Пример к теме
2	Управление **Быстрым счетчиком**

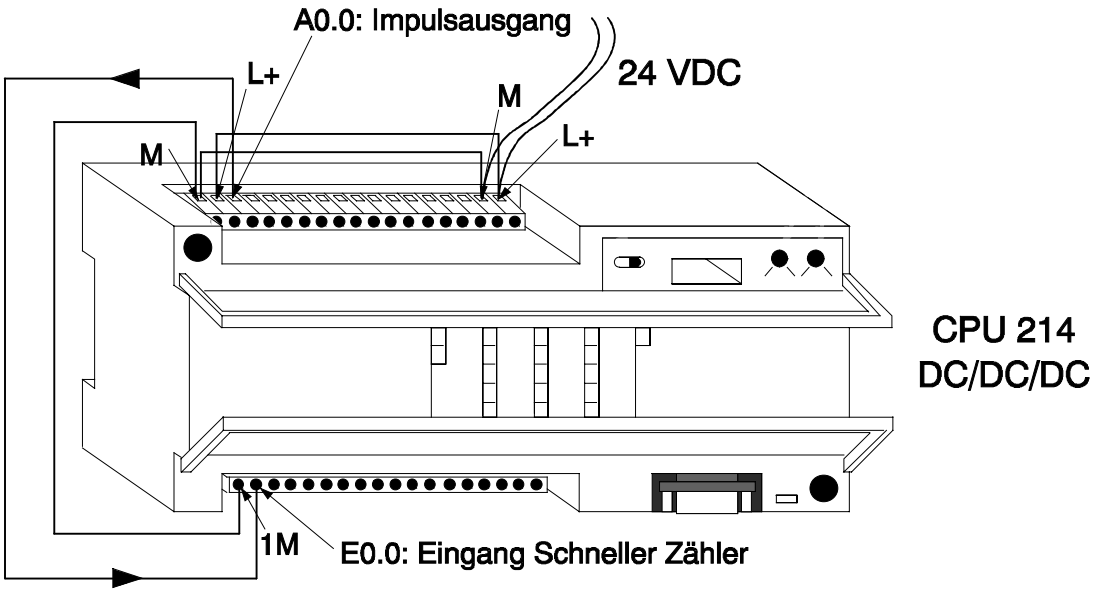
Краткое описание

В данном примере применения поясняется функция быстрого счетчика в SIMATIC S7-200. Быстрый счетчик может использоваться в различных конфигурациях, для того чтобы обрабатывать входные сигналы от датчиков, как напр. датчик перемещения.

Импульсные выходы будут использоваться в данном, для того чтобы создать быстрые сигналы. При этом появляется возможность генерировать последовательность импульсов (PTO), а также модулировать ширину импульсов (ШИМ), для того чтобы напр., управлять серводвигателем. Так как в данном применении используется последовательность импульсов, то можно использовать только **CPU 214 DC/DC/DC**.

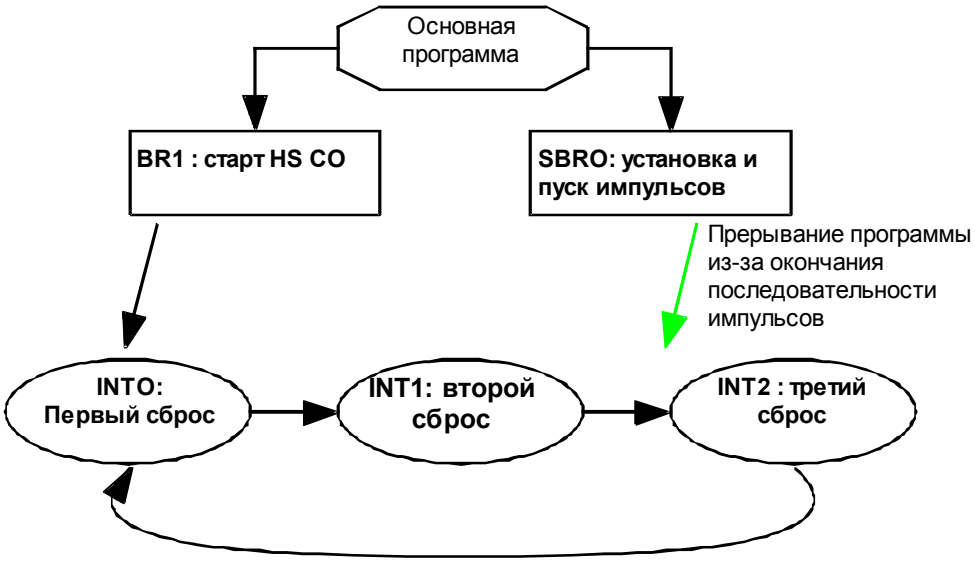
Данное применение призвано показать, как должна быть структурирована программа, для того чтобы использовать быстрый счетчик и функцию последовательности импульсов в простейших конфигурациях.

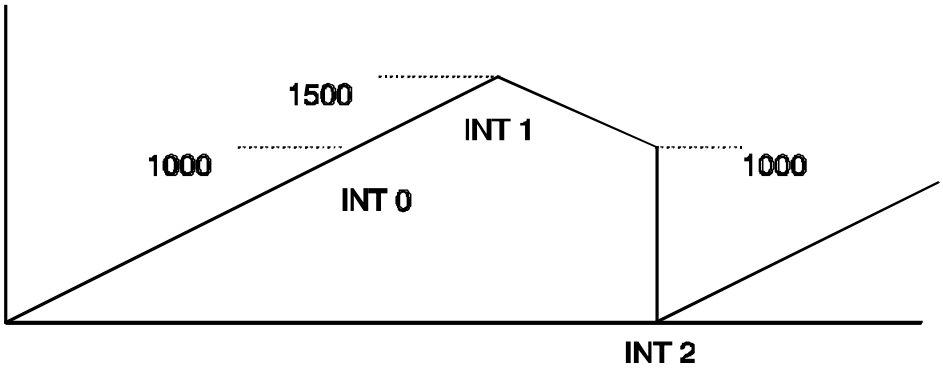
Схема включения



Copyright ©1996 by SIEMENS

Структура программы





Описание программы включая листинг

В этом примере описывается функция быстрого счетчика S7-200 DC/DC/DC. Он считает быстрее чем ПЛК может обработать цикл. Используемый счетчик это 2 килогерцовый программный счетчик, который встроен в S7-212. S7-214 имеет дополнительно два аппаратных счетчика по 7 КГц. Все счетчики вместе требуют 10 байт в памяти данных, для управления, хранения значений, счета и чтения текущего состояния.

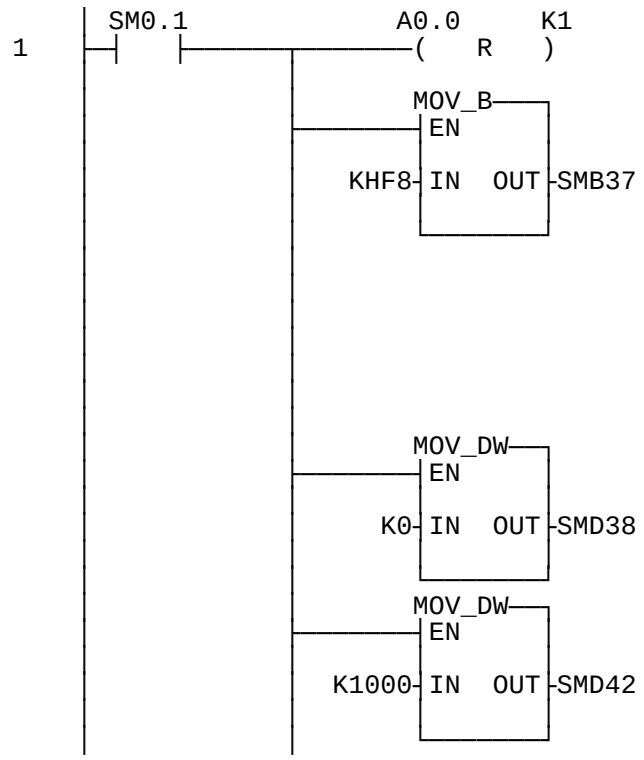
Размер программы составляет 96 слов.

Боле подробную информацию о быстром счетчике Вы найдете в Главе 6.1 Руководства по программированию SIMATIC S7-200, дальнейшую информацию о последовательности импульсов Вы найдете в Главе 6.3.

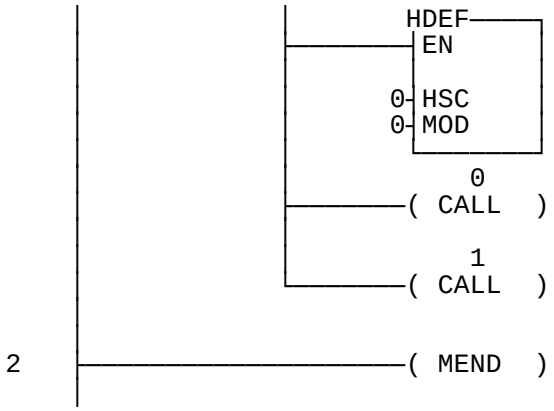
KOP (S7-MicroDOS)	AWL (TOOLITE2)

Основная программа

// Основная программа сбрасывает выход A0.0, так ка он необходим для
// функции импульсов. Кроме того инициализируется быстрый счетчик HSC0
// и вызываются подпрограммы 0 и 1. При этом HSC0 запускается
// со следующими свойствами: возможна актуализация значения, прямой
// счет. Программа завершается, если выполняется число циклов указанное
// в SBR 0 (SMD72).



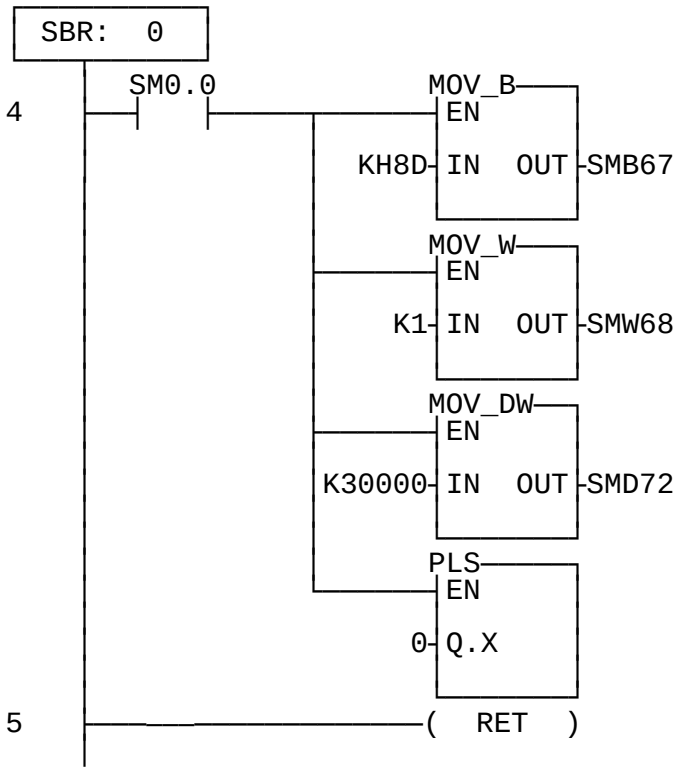

```
LD      SM0.1 // Первый цикл
R       A0.0,1 // Сброс выхода
                // A0.0 для импульсов
MOVB    16#F8,SMB37
                // Загрузка контрольных
                // битов для быстрого
                // счетчика HSC0
                // (деблокировка HSC0,
                // обновление текущих
                // значений, обновление
                // сбросов, обновление
                // направления счета и
                // числа циклов. Команда
                // HSC использует эти
                // контрольные биты,
                // для конфигурирования
                // быстрого счетчика.
MOVD    0,SMD38
                // Начальное значение
                // быстрого счетчика
                // HSC0

MOVD    1000,SMD42
                // Первое значение
                // останова от HSC0
```




```
HDEF    0,0     // Определение быстрого
                // счетчика 0 в режиме 0

CALL    0       // Вызов
                // подпрограммы 0

CALL    1       // Вызов
                // подпрограммы 1

MEND            // Конец основной
               // программы
```

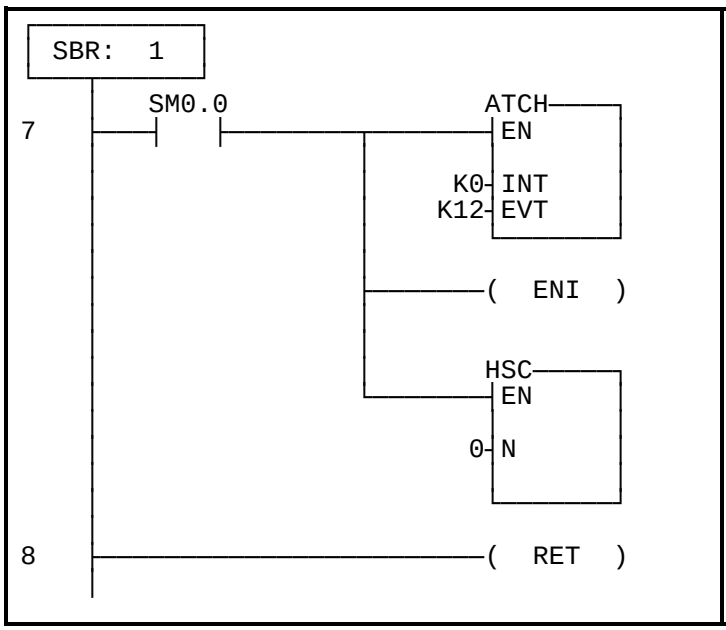
Подпрограммы

// Подпрограмма 0 служит для инициализации и деблокировки выдачи импульсов.
// В байте специальных меркеров SMB67 определяется вывод: последовательность импульсов,
// временой базис, обновление значений и деблокировка. Слово SMW68 содержит
// время цикла как мультипликатор временного базиса. В двойном слове специальных меркеров
// задается число генерируемых циклов.




```
SBR     0       // Подпрграмма 0

LD      SM0.0   // Всегда в единице
MOVB    16#8D,SMB67
                // Выдача импульсов:1мс,
                // PTO, деблокировка,
                // обновление

MOVW    1,SMW68
                // Длина цикла в мс

MOVD    30000, SMD72
                // Число гененируемых
                // циклов

PLS     0       // Разрешение
                // выдачи импульсов на
                // выходA0.0

RET             // Конец
                // подпрограммы 0
```

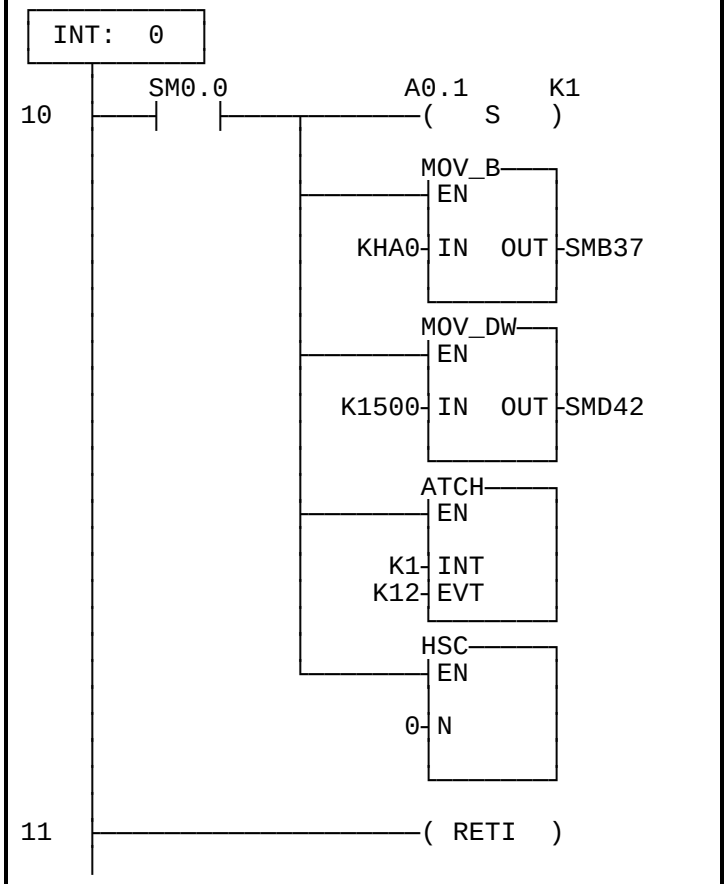
// Подпрограмма 1 запускает быстрый счетчик HSC0 и назначает
// подпрограмму 0 событию прерывания 12 (HSC0 текущее значение = сброс).
// Это событие возникает, если число сосчитанных импульсов (текущее значение)
// достигает текущего значения останова (сброс).
// Происходит деблокировка прерывания.




```
SBR    1       // Подпрограмма 1

LD     SM0.0   // Всегда в единице
ATCH   0,12    // Назначение INT 0
               // событию прерывания 12
               // (HSC0 текущее
               // значение = сброс)

ENI            // Деблокировка
               // прерывания

HSC    0       // Пуск быстрого
               // счетчика 0

RET            // Конец
               // подпрограммы 1
```

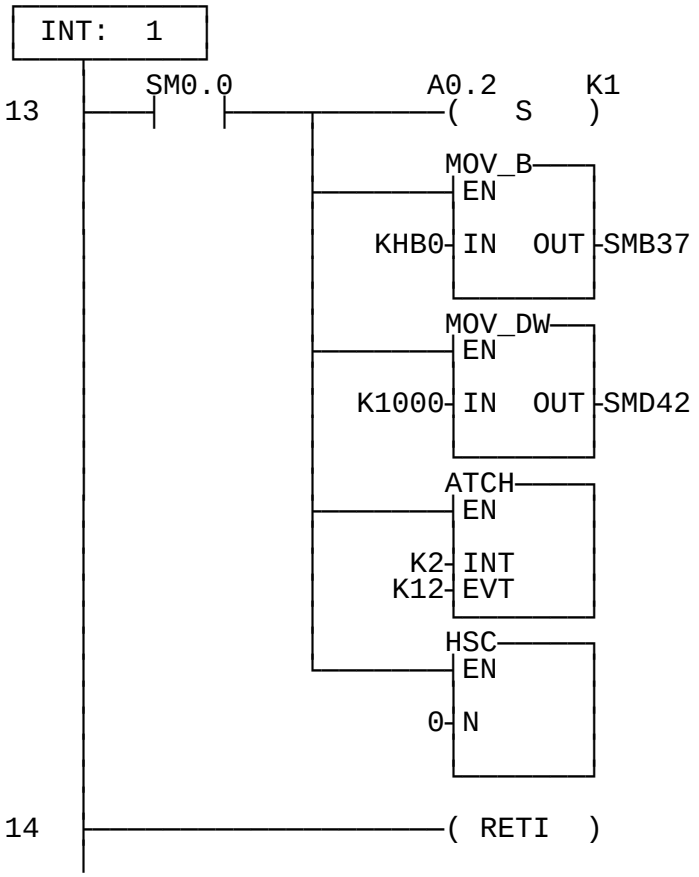
Подпрограммы прерываний

```
//Подпрограмма прерывания 0 будет вызвана, если достигнуто первое значение сброса (1000)
// быстрого счетчика 0. Выход A0.1 устанавливается и задается новое
// значение сброса (1500) для быстрого счетчика.
// Подпрограмма прерывания 1 назначается событию прерывания 12
// -вместо подпрограммы 0.
```



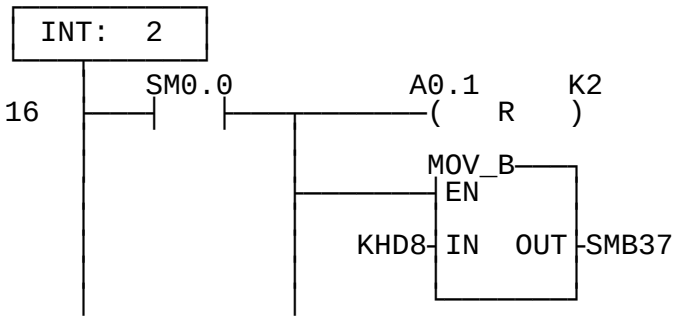

```
INT    0       // Подпрограмма
               // прерывания 0
LD     SM0.0   // Всегда в единице
S      A0.1,1  //Установка выхода A0.1

MOVD   16#A0,SMB37
               //Установка контрольных
               // битов: только загрузка
               // нового сброса

MOVD   1500,SMD42
               // Следующее значение
               // сброса HSC0

ATCH   1,12    // Назначение INT 1
               //событию прерывания 12
               // вместо INT 0

HSC    0       // Загрузка нового
               // сброса в HSC0

RETI           // Конец подпрограммы
               // прерывания 0
```

// Подпрограмма прерывания 1 вызывается, если достигнуто второе значение сброса
// (1500) быстрого счетчика 0. Выход A0.2 устанавливается, направление счета
// изменяется на обратное и определяется новое значение сброса (1000).
// Подпрограмма прерывания 2 назначается событию прерывания 12

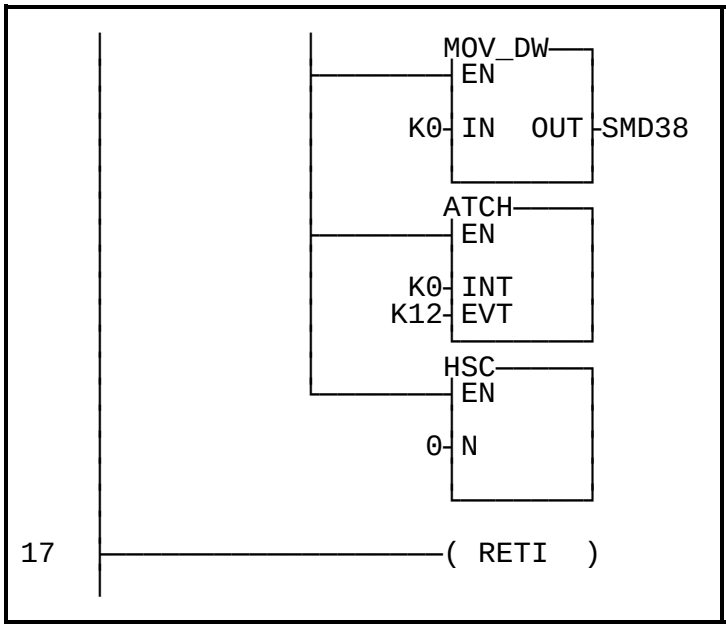


INT	**1**	**// Подпрограмма // прерывания 1**
LD	**SM0.0**	**// Всегда в единице**
S	**A0.2,1**	**//Установка выхода A0.2**
MOVB	**16#B0,SMB37**	**//Установка контрольных // битов для загрузки // нового сброса и // обратного счета**
MOVD	**1000,SMD42**	**// Следующее значение // сброса**
ATCH	**2,12**	**// Назначение INT 2 //событию прерывания 12 // вместо INT 1**
HSC	**0**	**// Загрузка нового сброса // и нового направления // в HSC0**
RETI		**// Конец // подпрограммы // прерывания 1**

// Подпрограмма прерывания 2 вызывается, если достигнуто третье значение сброса (1000)
// быстрого счетчика 0. Выходы A0.1 и A0.2 сбрасываются, направление счета
// изменяется на прямое и текущее значение счетчика сбрасывается в ноль.
// Значение сброса остается неизменным. При этом подпрограмма прерывания 0
// назначается событию прерывания 12, выполнение программы начинается заново.
// Программа завершается, если заданное в SBR 0 (SMD72) число циклов отработано.



INT	**2**	**// Подпрограмма // прерывания 2**
LD	**SM0.0**	**// Всегда в единице**
R	**A0.1,2**	**// Сброс выходов // A0.1 и A0.2**
MOVB	**16#D8,SMB37**	**//Установка контрольных // битов, для загрузки // нового текущего // значения прямого счета**



MOVD	**0,SMD38**	**// Сброс текущего //значения счетчика HSC0**
ATCH	**0,12**	**// Назначение INT 0 //событию прерывания 12**
HSC	**0**	**// Новый пуск // быстрого счетчика 0**
RETI		**// Конец подпрограммы // прерывания 2**

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ → ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии к строкам начинающиеся с "**//**" в S7-Micro/DOS не возможны, зато возможны комментарии к сегментам.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

Copyright ©1996 by SIEMENS

SIMATIC
S7-200 Примеры

Группа	Пример к теме
3	Простые применения **свободнопрограммируемого интерфейса**

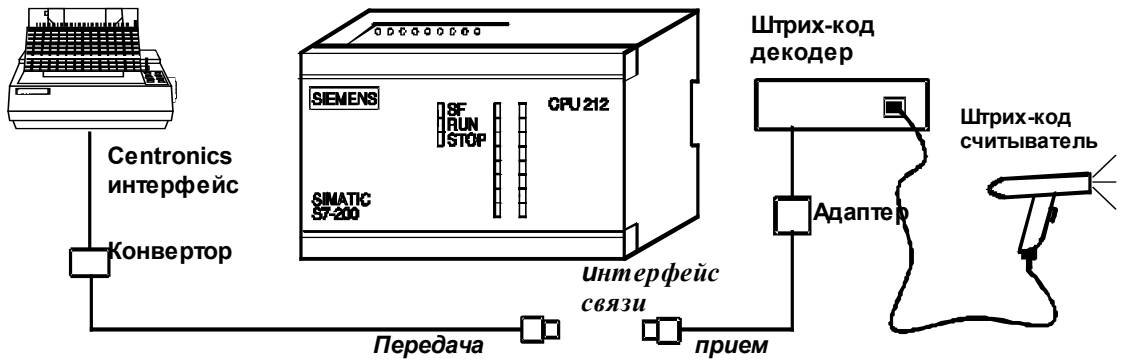
Краткое описание

В данном примере применения описывается использование свободнопрограммируемого интерфейса.
Свободнопрограммируемый означает в данном случае, что протокол свободно определяем. Информация необходимая для связи заносится при этом в байт специальных меркеров SMB30. Пользователь должен помнить о следующих соглашениях:

- Четность
- Число битов на символ
- Скорость

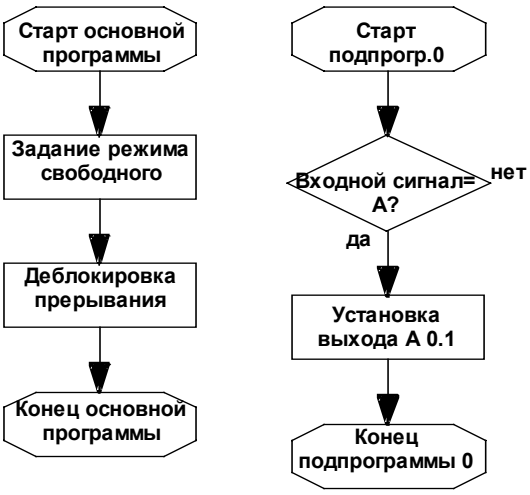
В режиме передачи именуемом **режим свободного порта** данные могут как передаваться так и приниматься. В данном примере описана передача данных с имитацией программы печати. Для того чтобы пояснить прием данных, добавлена программа считывателя штрих-кода.

Схема включения



Структура программы: Принтер

Структура программы: Штрих-код считыватель



Описание программы включая листинг: принтер

В данной программе описана передача данных на принтер. Для того чтобы упростить реализацию этого примера, вместо принтера в качестве приемника можно подключить программу-терминал под Windows.

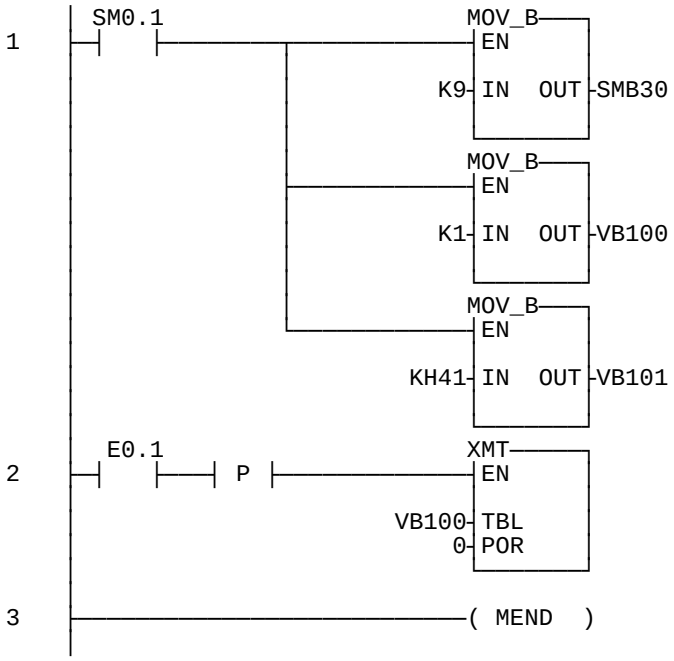
Программа составляет 13 слов.

Более подробную информацию о свободнопрограммируемом интерфейсе Вы найдете в Главе 2.6 Руководства по программированию.

KOP (S7-MicroDOS)	AWL (TOOLITE2)

Основная программа

// TITEL = Режим свободного порта
// Для этого приложения важно, корректно установить режим свободного порта.
// В байт специальных меркеров SMB30 заносится необходимая информация.
// Введенные значения можно определить с помощью руководства.
// Команда передачи XMT содержит начальный адрес передаваемой информации.
// Начальный адрес в результате содержит информацию о длине сообщения,
// указанной в байтах.




```
LD      SM0.1          //Меркер первого цикла
MOVB    +9,SMB30       // Свободный порт,
                       //9600 Бод,без паритета
                       // 8 бит/символ
MOVB    +1,VB100       // Длина сообщения: 1
                       // ASCII символ
MOVB    16#41,VB101    // Длина символа “A”
                       // 1 байт (A => 41 hex)
LD      E0.1           // Начало передачи
                       // вход E0.1
EU                     // Определение
                       //положит. фронта
XMT     VB100,0        // Передача на
                       // интерфейс связи
MEND                   // Конец
                       // основной программы
```

Описание программы вкл.листинг: Считыватель штрих-кода

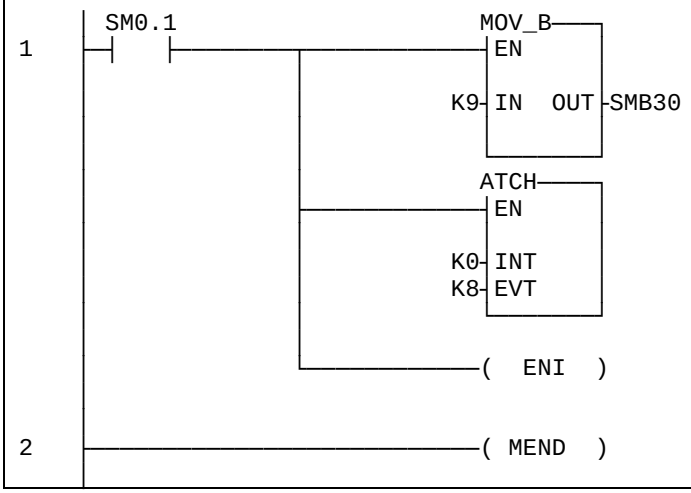
В данной программе описан прием данных. Штрих-код считыватель посылает при этом считанные данные через свободнопрограммируемый интерфейс на SIMATIC S7-200. Чтобы упростить реализацию данного примера, в качестве передатчика вместо считывателя штрих-кода можно использовать программу-терминал под Windows.

Размер программы составляет 15 слов.

Подробнее о свободнопрограммируемом интерфейсе Вы найдете в Главе 2.6 руководства по программированию SIMATIC S7-200.

Основная программа

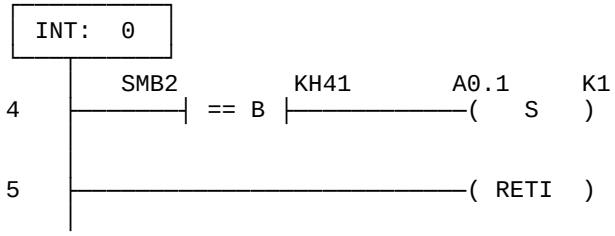
// TITEL = Режим свободного порта
// Для этого приложения важно, корректно установить режим свободного порта.
// В байт специальных меркеров SMB30 заносится необходимая информация.
// Введенные значения можно определить с помощью руководства.
// Принятые данные реализуются через прерывание. Если данные поступят через
// свободнопрограммируемый интерффейс, то выполняется т.н. прерывание для приема,
// которое в данном приложении носит обозначение INT 0.



LD	**SM0.1**	**// Меркер первого цикла**
MOVB	**+9,SMB30**	**// Свободный порт, //9600 Бод,без паритета // 8 бит/символ**
ATCH	**0, 8**	**//Назначен. прерывания // для приема 0**
ENI		**// Деблокировка // подпрограммы // прерываний**
MEND		**// Конец осн.программы**

Подпрограмма прерывания

// В подпрограмме прерывания 0 принятые символы, сохраняемые в
// байте специальных меркеров SMB2, сравниваются с большой буквой “A”.
//в случае совпадения устанавливается выходной бит A0.1.



INT	**0**	**// Подпрограмма прерывания // для приема**
LDB=	**SMB2,16#41**	**// Сравнение принятого // символа в SMB2 с “A”**
S	**A0.1,1**	**// Если символ “A” // получен, то // устанавливается A0.1**
RETI		**// Возврат в основную // программу**

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 →K4)
- Замените ‘16#’ → ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии к строкам, начинающиеся с "**//**" в S7-Micro/DOS не возможны, зато возможны комментарии к сегментам.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

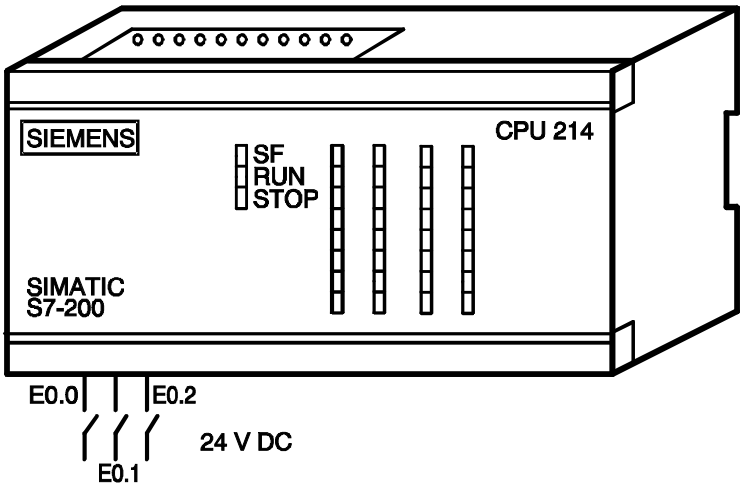
Группа	Пример к теме
2	Различные возможности, **установки битов или байтов**

Краткое описание

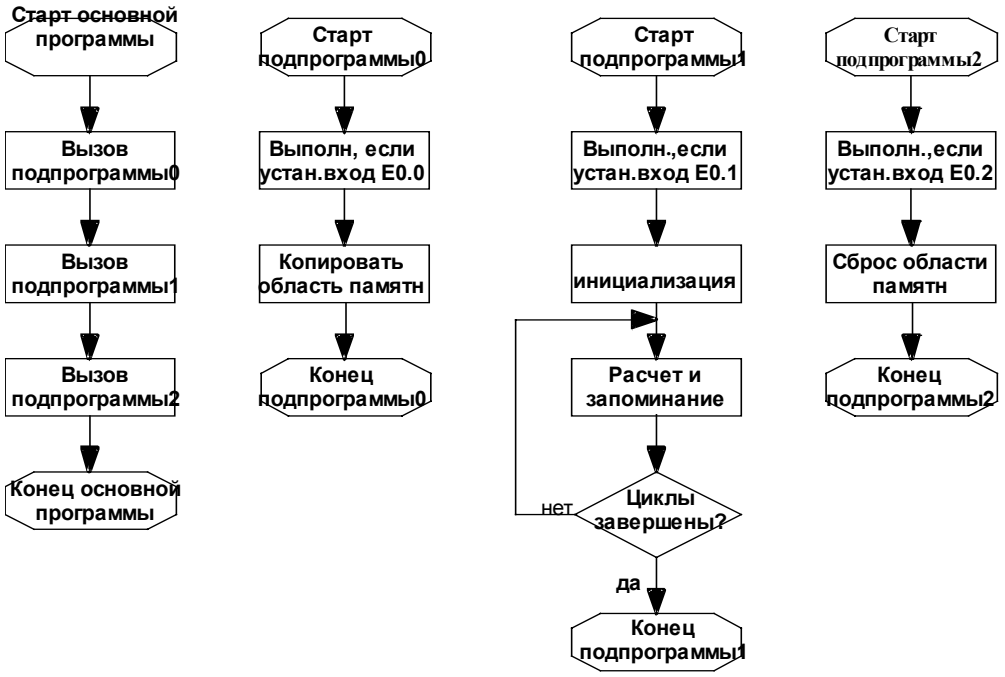
Данная программа описывает различные возможности занесения в области памяти определенных значений или очистки определенных областей памяти.

В этом примере применения рассмотрены:
- Команда FILL,
- цикл FOR-NEXT с применением указателя и
- команда RESET.

Схема включения



Структура программы



Описание программы вкл. листинг

Этот пример программы описывает различные возможности занесения определенных значений в заданные биты и байты или очистки определенных областей памяти.
Используются команды:

FILL	Устанавливает один или несколько битов
FOR... NEXT	Цикл FOR...NEXT
R	Сбрасывает один или несколько битов

Размер программы составляет 55 слов.

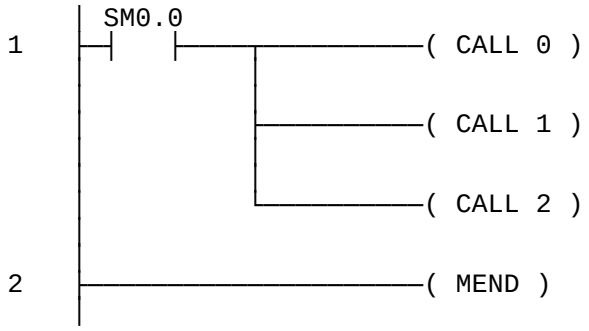
Более подробную информацию об установке и сбросе битов и байтов Вы найдете в Руководстве по программированию SIMATIC S7-200 в Главе 4.7 “Операции с областями данных” (FILL), 5.6 “Операции FOR и NEXT” и 3.5 “Логические операции с выходами” (RESET).

KOP (S7-MicroDOS)	AWL (TOOLITE2)

Основная программа

// TITEL=FORNEXT

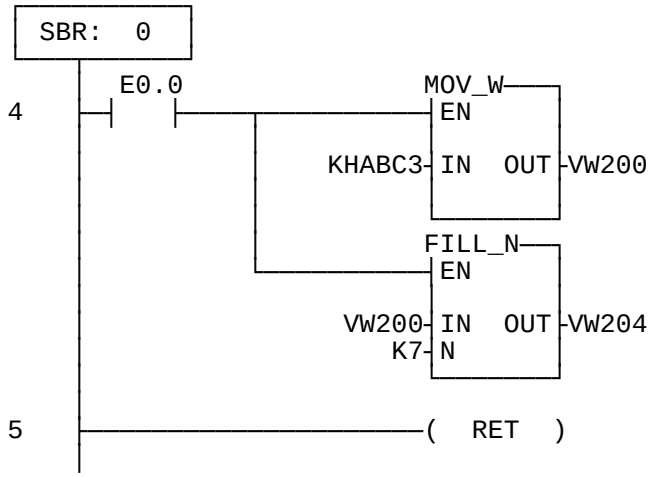
// Основная программа содержит вызовы подпрограмм 0, 1 и 2.




```
LD      SM0.0 // Всегда в единице
CALL    0     // Вызов подпрограммы 0
              // (FILL)
CALL    1     // Вызов подпрограммы 1
              // (FOR...NEXT)

CALL    2     // Вызов подпрограммы 2
              // (Reset)

MEND          // Конец основной
              // программы
```

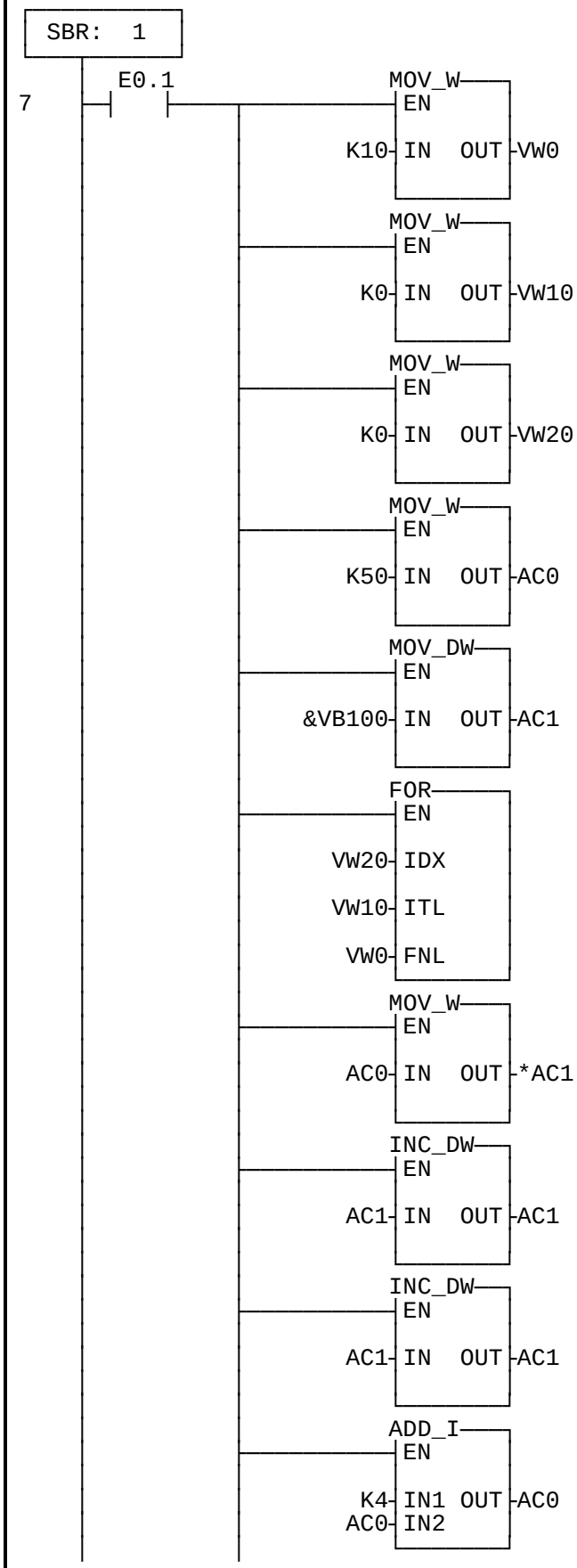
Подпрограммы

// Подпрограмма 0 копирует значение слова VW200 в слово VW204
// и в 6 последующих слов памяти (до VW216), если установлен
// вход E0.0.




```
SBR     0     // Подпрограмма 0

LD      E0.0  // Чтение входа E0.0
MOVW    16#ABC3,VW200
              // Запись 16-рич.значения
              // ABC3 в VW200

FILL    VW200,VW204,7
              // Копирование значения
              // слова VW200 в
              // слово VW204 и шесть
              // последующих
              // слов памяти (до
              // VW216)
RET           // Конец подпрограммы 0
```

// Подпрограмма 1 копирует следующие друг за другом числа в переменную
// область памяти, если установлен вход E0.1. Число выполняемых
// циклов определяется номерами первого (в VW10) и последнего
// цикла (в VW0). Текущее значение выполненных циклов находится в
// слове памяти VW20. Первое число, с которого надо начинать счет,
// загружается в аккумулятор AC0. Первый адрес, под которым надо
// запоминать значение,заносится в аккумулятор AC1, действующий
// в качестве указателя. Затем начинается выполнение циклов. К
// начальному значению AC0 за один цикл прибавляется 4, указатель
// AC1 увеличивается на следующее слово памяти, пока не будет
// обработан последний цикл.




```
SBR     1       // Подпрограмма 1
LD      E0.1    // Чтение входа E0.1
MOVW    10,VW0
                // Номер последнего
                // цикла в VW0

MOVW    0,VW10
                // Номер первого
                // цикла в VW10

MOVW    0,VW20
                // Текущий номер
                // цикла в
                // VW20 (счетчик)

MOVW    50,AC0
                // Начальное значение
                // счета в аккумулятор 0

MOVD    &VB100,AC1
                // аккумулятор 1
                // (указатель памяти)
                // указывает на байт
                // памяти VB100

FOR     VW20,VW10,VW0
                // Начало цикла

MOVW    AC0,*AC1
                // Сохранить текущее
                // значение счетчика по
                //текущ. адресам памяти

INCD    AC1     // Инкрементировать
                // указатель памяти
                // на 1 байт

INCD    AC1     // Инкрементировать
                // указатель памяти
                // на 1 байт

+I      4,AC0   // Прибавить 4 к текущ.
                // значению счетчика
```

Copyright ©1996 by SIEMENS

```
                    (  NEXT   )      NEXT              // Конец цикла

8                   (   RET   )      RET               // Конец подпрограммы 1
```

// Подпрограмма 2 сбрасывает биты памяти с V100.0 до V121.7 и
// с V204.0 до 217.7, если установлен вход E0.2.

```
 SBR:  2                             SBR        2      // Подпрограмма 2

     E0.2            VB100.0   K176  LD         E0.2   // Установка входа E0.2
10  ─┤   ├──────────(   R   )        R          V100.0,176
                                                       // Сброс битов с V100.0
                     V204.0   K112                     // до 121.7
                    (   R   )        R          V204.0,112
                                                       // Сброс битов с V204.0
                                                       // до 217.7
11                  (   RET   )      RET                // Конец подпрограммы 2
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ → ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии к строкам, начинающиеся с "**//**" в S7-Micro/DOS невозможны, зато возможны комментарии к сегментам.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

Группа	Пример к теме
2	Управление **широтно-импульсной модуляцией**

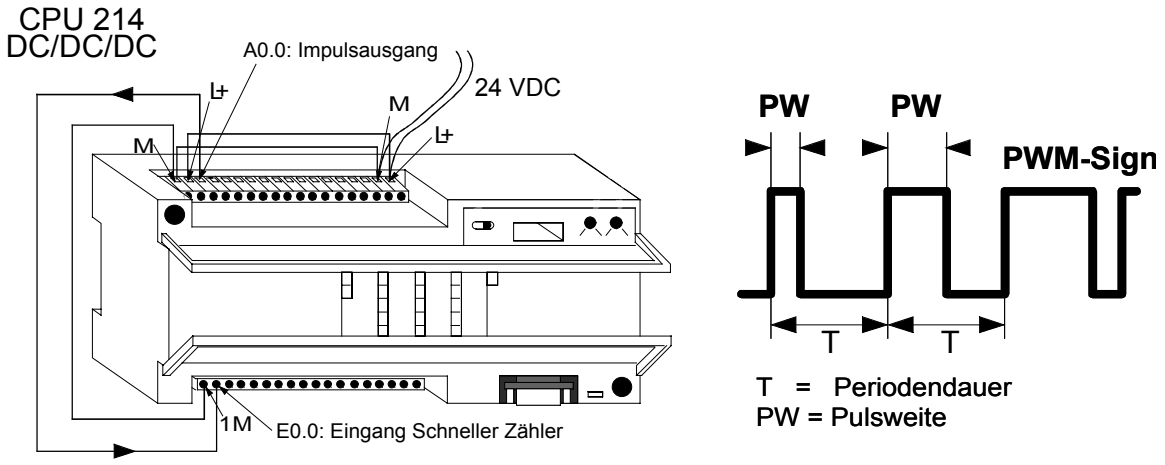
Краткое описание:

CPU-214 S7-200 содержит функцию выдачи прямоугольных сигналов на выходы A0.0 и A0.1. При этом длительность периода и ширина импульса устанавливаются независимо друг от друга. Ширина импульса соответствует времени, в течение которого выходной сигнал за период имеет состояние „1“.

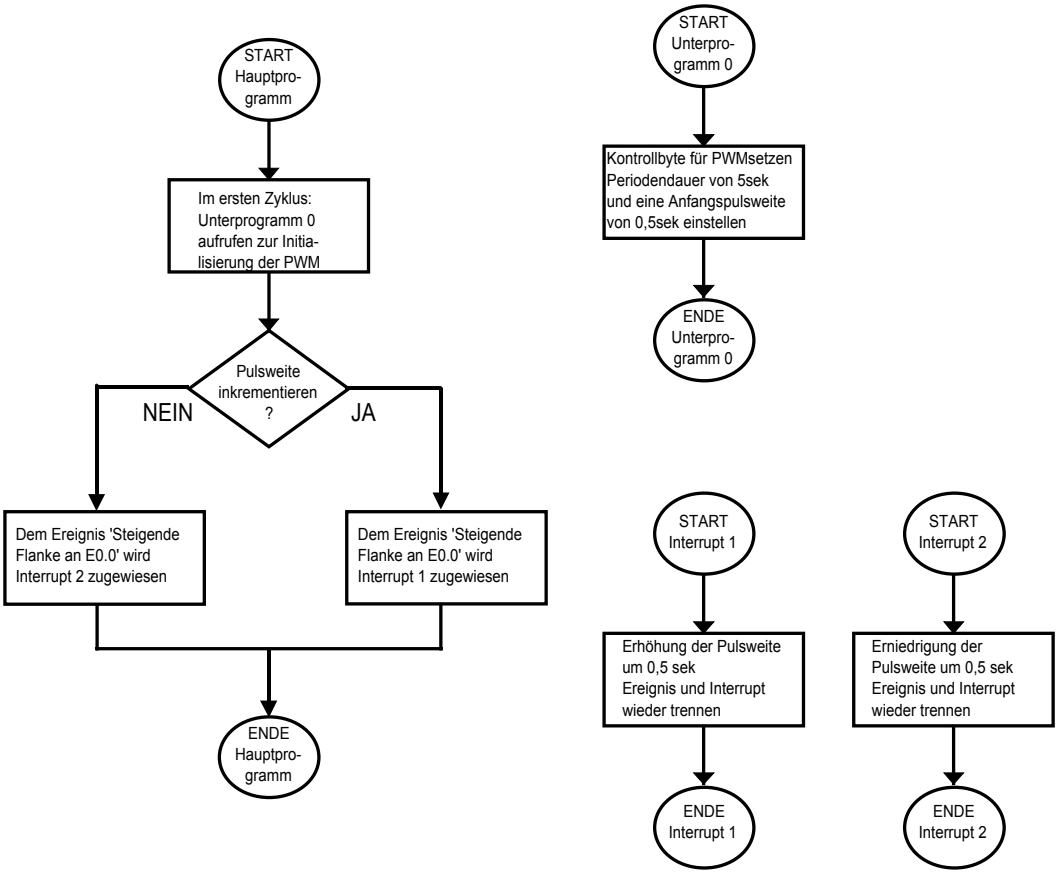
Данный пример описывает широтно-импульсную модуляцию (ШИМ), где на выход A0.0 выдается сигнал, ширина импульса которого увеличивается от периода к периоду на 0,5 с. Длительность периода составляет 5 с, а стартовая ширина импульса 0,5 с. Если будет достигнуто максимальное значение ширины импульса 4,5 с, то ширина импульса снова начинает уменьшаться на 0,5 с, пока значение не станет равно нулю. Этот процесс повторяется периодически.

Для проведения такой широтно-импульсной модуляции необходимо связать выход A0.0 со входом E0.0, также как массу напряжения питания с массой входов, для того чтобы имелась возможность управлять ШИМ из программы.

Схема включения:



Структура программы:



Описание программы вкл. листинг:

Байт специальных меркеров SMB67 служит для инициализации широтно-импульсной модуляции на выходе A0.0. Это инициализация содержит деблокировку ШИМ и установки, позволяющие изменять длительность периода и ширину импульса, а также выбрать временной базис (здесь в мс). В подпрограмме 0 будут установлены соответствующие контрольные байты. Команда ENI производит глобальную деблокировку всех прерываний. По команде PLS 0 операционная система запоминает установленные значения и инициализирует так называемый “PTO/ШИМ-генератор”, который будет производить ШИМ на выходе A0.0.

Длительность периода 5 с задается передачей значения 5000 в слово специальных меркеров SMW68. Начальная ширина импульса устанавливается на 0,5 с, путем записи значения 500 в слово специальных меркеров SMW70.

Данная инициализация производится в первом цикле, путем логического связывания вызова подпрограммы с битом специальных меркеров SM0.1 (First Scan Flag). Инициализация и установка ШИМ будет прооизводится повторно, после завершения процесса ШИМ, т.е. когда текущая ширина импульса станет равной 0.

Вспомогательный меркер M0.0 служит для установки, будет ли ширина импульса увеличиваться или уменьшаться. Во время инициализации он устанавливается, для того чтобы могло производиться инкрементирование. Выход A0.0 связан со входом E0.0, так что выходной сигнал появляется и на входе E0.0. После того как будет выдан первый импульс, событие ‘0’ (положительный фронт на E0.0) будет назначено подпрограмме прерывания 1 (INT1) с помощью команды ‘ATCH’.

INT1 повышает текущее значение импульса на 0,5 с а затем команда ‘DTCH’ разрывает связь этого прерывания с INT1, причем прерывание снова будет блокировано. Если ширина импульса при следующем увеличении станет равна длительности периода, то вспомогательный меркер M0.0 будет сброшен. При этом событие ‘0’ назначается подпрограмме прерывания 2, которая уменьшат ширину импульса после каждого импульса на 0,5 с. Когда ширина импульса станет равна 0, снова будет вызвана подпрограмма инициализации (подпрограмма SBR 0).

Размер программы составляет 66 слов.

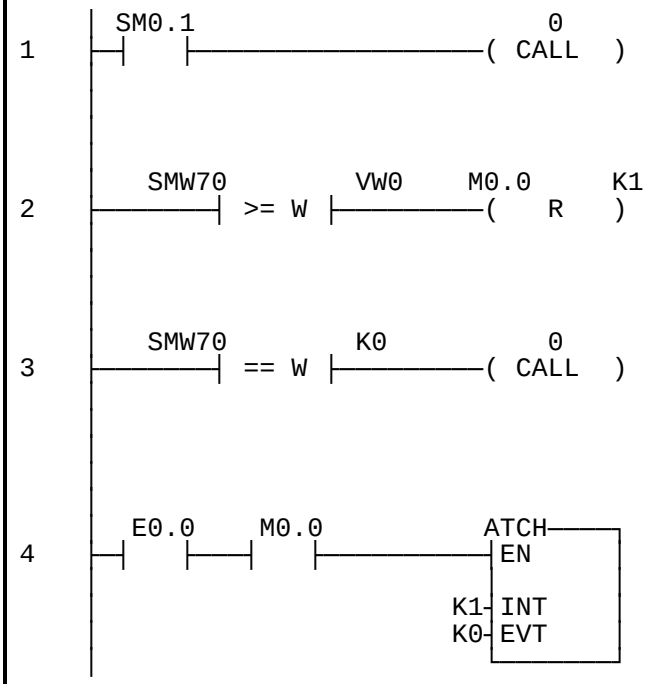
Более подробную информацию о широтно-импульсной модуляции Вы найдете в Главе 6.3 “Быстрые операции на выходах” Руководства по программированию SIMATIC S7-200.

KOP (S7-MicroDOS)	AWL (TOOLITE2)

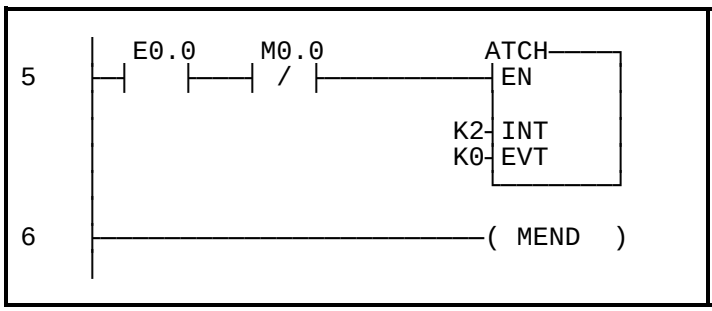
Основная программа

// TITEL = УПРАВЛЕНИЕ ШИРОТНО-ИМПУЛЬСНОЙ МОДУЛЯЈИЕЙ

// ************************** ОСНОВНАЯ ПРОГРАММА **************************************



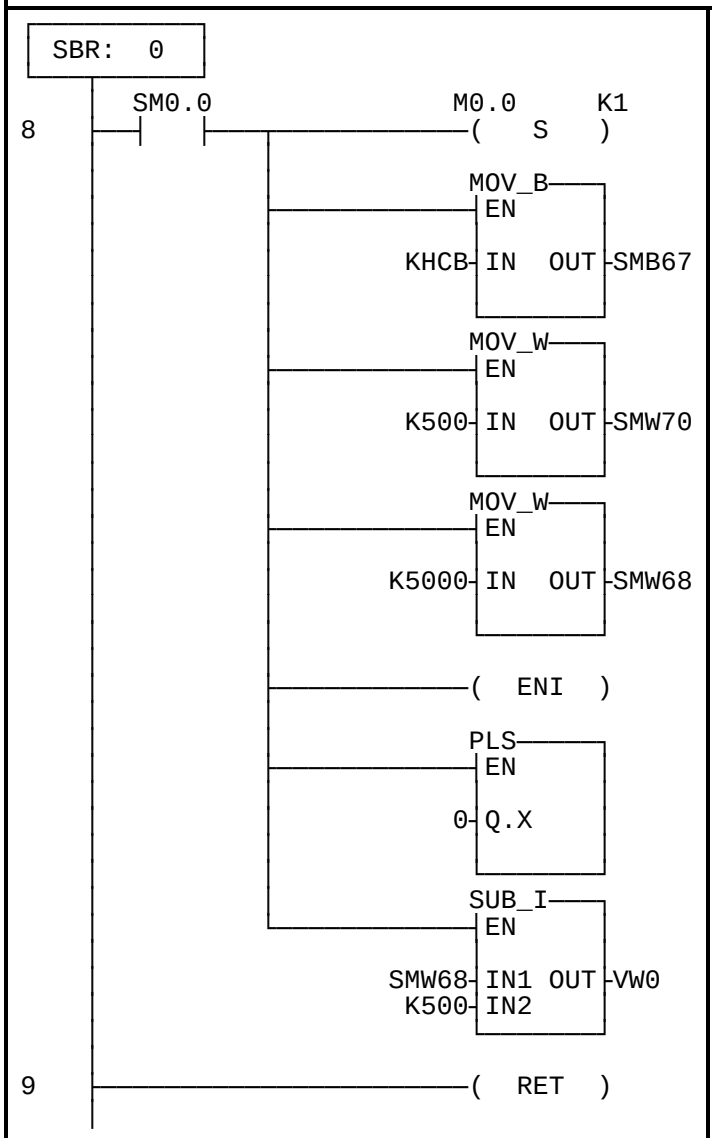

```
LD      SM0.1 // В первом цикле:
CALL    0     // вызов
              // подпрограммы 0 для
              // пуска ШИМ

LDW>=   SMW70, VW0
              // Если ширина импульса
              // >= (длина периода -
              // ширина импульса),
R       M0.0, 1// вспомогат. меркер
              // сбрасывается
LDW=    SMW70, 0
              // Если ширина импульса
              // равна нулю,
CALL    0     // Вызов
              // подпрограммы 0 для
              // нового пуска ШИМ
LD      E0.0  // ВходE0.0 установлен
U       M0.0  // и вспомогательный
              // меркер для увеличения
              // установлен,
ATCH    1, 0  // то INT 1 назначена
              // событию 0
              // (положительный фронт
              // на E0.0)
```




```
LD      E0.0    // Вход E0.0 установлен,
UN      M0.0    // а вспомогательный
                // меркер для увеличения
                // не установлен,
ATCH    2, 0    // то INT 2 назначена
                // событию 0 (положи-
                // телный фронт на E0.0)
MEND            // Конец основной
                // программы
```

Подпрограммы

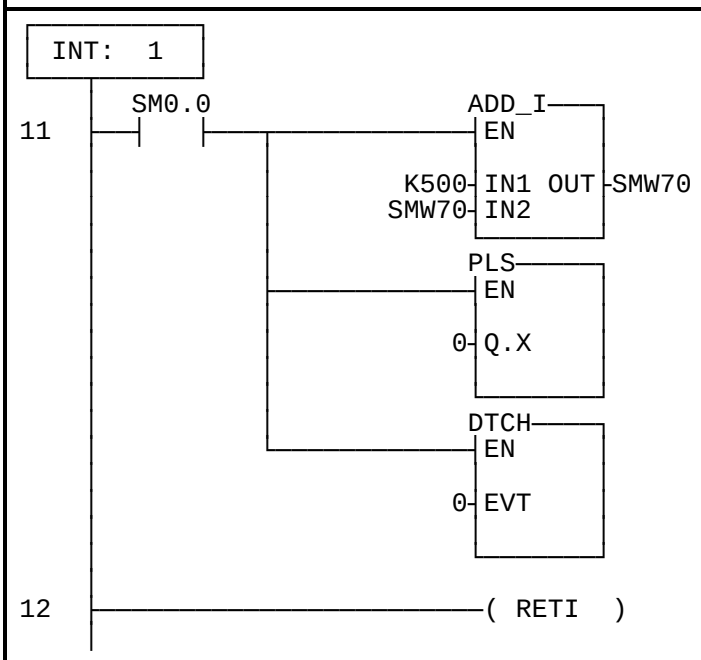
```
//*************************   ПОДПРОГРАММА 0   ******************************

// SM67.0 := 1        =>  сохранение новой длительности периода (деблокировка)
// SM67.1 := 1        =>  сохранение новой ширины импульса (деблокировка)
// SM67.3 := 1        =>  временной базис 1мс (если 0 => временной базис 1мкс)
// SM67.6 := 1        =>  выбор режима ШИМ (если 0 => режим PTO)
// SM67.7 := 1        =>общая деблокировка High-Speed-Output-функций
```




```
SBR     0       // Инициализация
                // ШИМ
LD      SM0.0   // Всегда в единице
S       M0.0, 1 // Установка вспом.
                // меркера “увеличение
                // ширины импульса”
MOVB    16#CB, SMB67
                //Установка контрольных
                //байтов PTO/ШИМ-
                // для выхода A0.0

MOVW    500, SMW70
                // Задание начальной
                // ширина импульса
                // (здесь 500 мс)

MOVW    5000, SMW68
                // длина периода
                // (здесь 5 с)

ENI             // Деблокировка всех
                // прерываний
PLS 0           // Команда на
                // программирование
                //PTO/ШИМ-генератора

MOVW    SMW68, VW0
                // Копировать длину
                // периода в слово
                // данных VW0
-I      500, VW0
                // Сохранить значение
                // 'длина периода-
                // ширина импульса'
                // в слове данных VW0
RET             // Конец подпрограммы 0
                // и  возврат в
                // основную программу
```

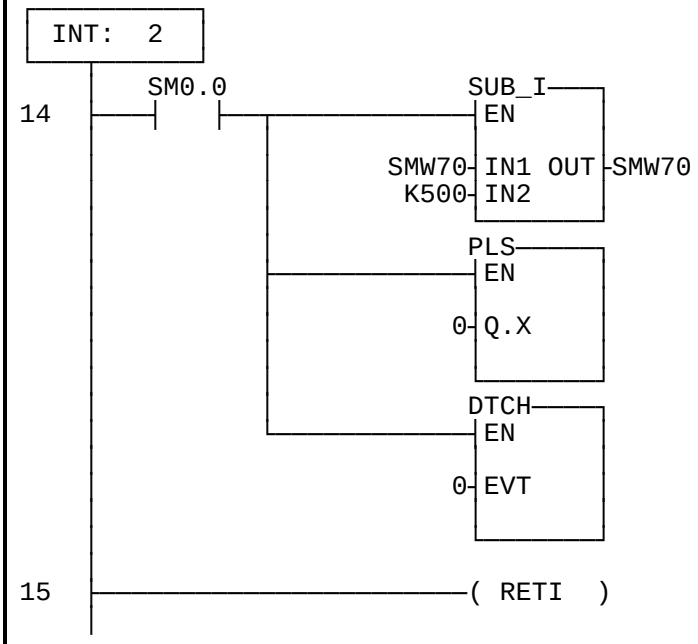
Подпрограммы прерываний

// ********** INTERRUPT SERVICE ROUTINE 1 **********




```
INT   1      // Увеличение ширины
             // импульса
LD    SM0.0  // Всегда в единице
+I    500, SMW70
             // Увеличение ширины
             // импульса на 500мс

PLS   0      // Команда на
             // программирование
             //РТО/ШИМ-генератора

DTCH  0      // Разделить прерывание
             // и назначенное ему
             // событие 0

RETI         // Конец ISR 1 и
             // возврат в
             // основную программу
```

// ********** INTERRUPT SERVICE ROUTINE 2 **********




```
INT   2      // Уменьшение ширины
             // импульса
LD    SM0.0  // Всегда в единице
-I    500, SMW70
             // Уменьшение ширины
             // импульса на 500мс

PLS   0      // Команда на
             // программирование
             // РТО/ШИМ-генератора

DTCH  0      // Разделить прерывание
             // и назначенное ему
             // событие 0

RETI         // Конец ISR 2 и
             // возврат в
             // основную программу
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии к строкам, начинающиеся с "**//**" в S7-Micro/DOS не возможны, зато возможны комментарии к сегментам.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

Copyright ©1996 by SIEMENS

SIMATIC
S7-200 Примеры

Группа	Пример к теме
2	Управление **часами реального времени** SIMATIC S7-214

Краткое описание

Данный пример программы содержит две специальных операции с часами реального времени: чтение и запись даты и времени. Для этих операций понадобится 8-байтовый буфер, со следующей структурой.

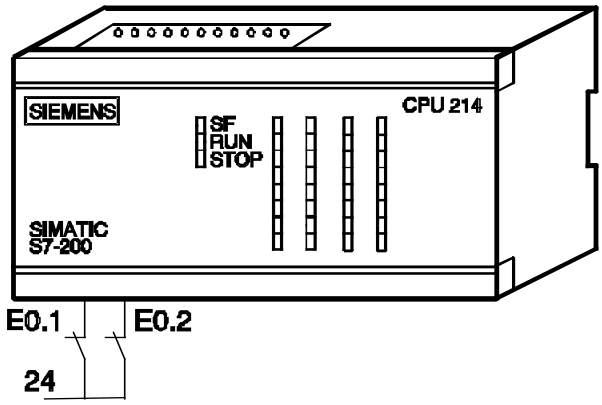
Байт 0: Столетие (00 - 99)
Байт 1: Месяц (1 - 12)
Байт 2: День (1 - 31)
Байт 3: Часы (00 - 24)
Байт 4: Минуты (00 - 59)
Byte 5: Секунды (00 - 59)
Byte 6: не занято
Byte 7: День недели (1-7 = Вс-Сб)

Данные хранятся в коде BCD (при чтении) или должны заноситься в коде BCD (при записи). При нажатии кнопки на E0.0 в часы реального времени будет занесена предустановленная дата. Кнопка на E0.1 служит для индикации текущего значения секунд, которое копируется на выходной байт AB0. Кодирование при этом производится на выбор в формате BCD (E0.1 = '1') или двоичном (E0.1 = '0').

Схема включения

EO.1=ON: Индикация секунд в формате BCD
EO.1=OFF: Индикация секунд в двоичном формате

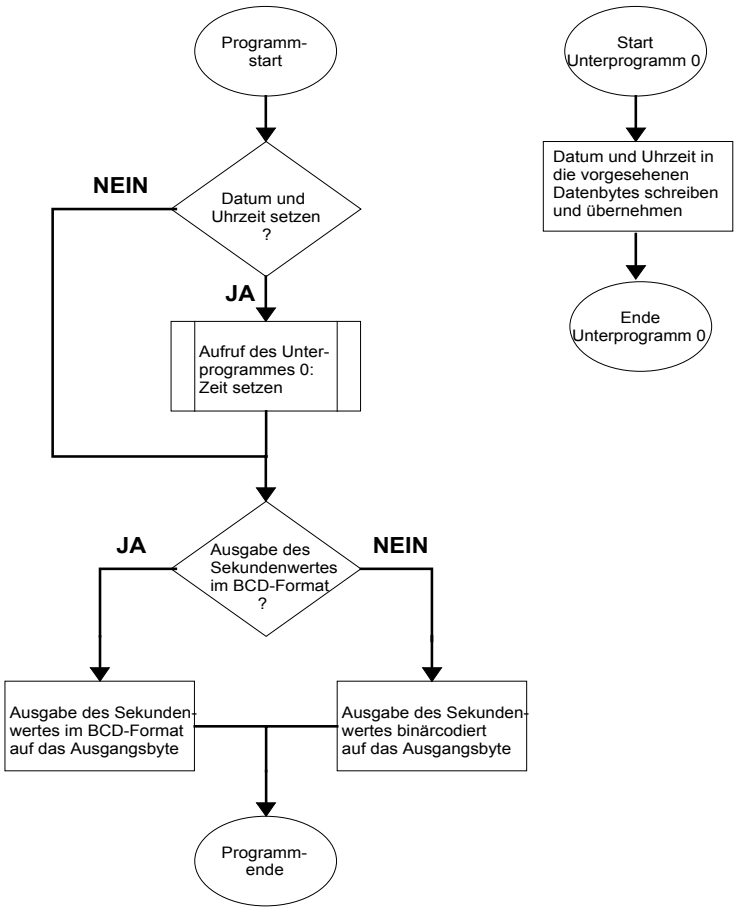
EO.0: Устанавливаются
Дата: 14.02.95
Время: 12:00:00



95	VB400	Год
02	VB401	Месяц
14	VB402	День
12	VB403	Часы
00	VB404	Минуты
00	VB405	Секунды
00	VB406	
03	VB407	День нед.

Copyright ©1996 by SIEMENS

Структура программы



Описание программы вкл. листинг

При нажатии кнопки на входе E0.0 будет вызвана подпрограмма 0. Эта подпрограмма заполняет 8 байтов VB100 до VB107 соответствующими значениями для даты и времени. Заключительная команда 'TODW' сохраняет установки для часов реального времени.

Часы реального времени считываются в каждом цикле. Эти данные заносятся в 8 байтов с VB400 по VB407 в формате BCD (4 бита представляют цифру от 0 до 9). Если установлен вход E0.1, то это значение копируется непосредственно на выходной байт.

Если вход E0.1 не установлен, то слово данных VW404 копируется в слово VW204, а затем старший байт VB204, который содержит значение минут, стирается. Это мероприятие необходимо, так как конвертирование значения секунд из формата BCD в двоичный может быть произведено только пословно. Теперь двоичнокодированное значение секунд передается на выходной байт AB0.

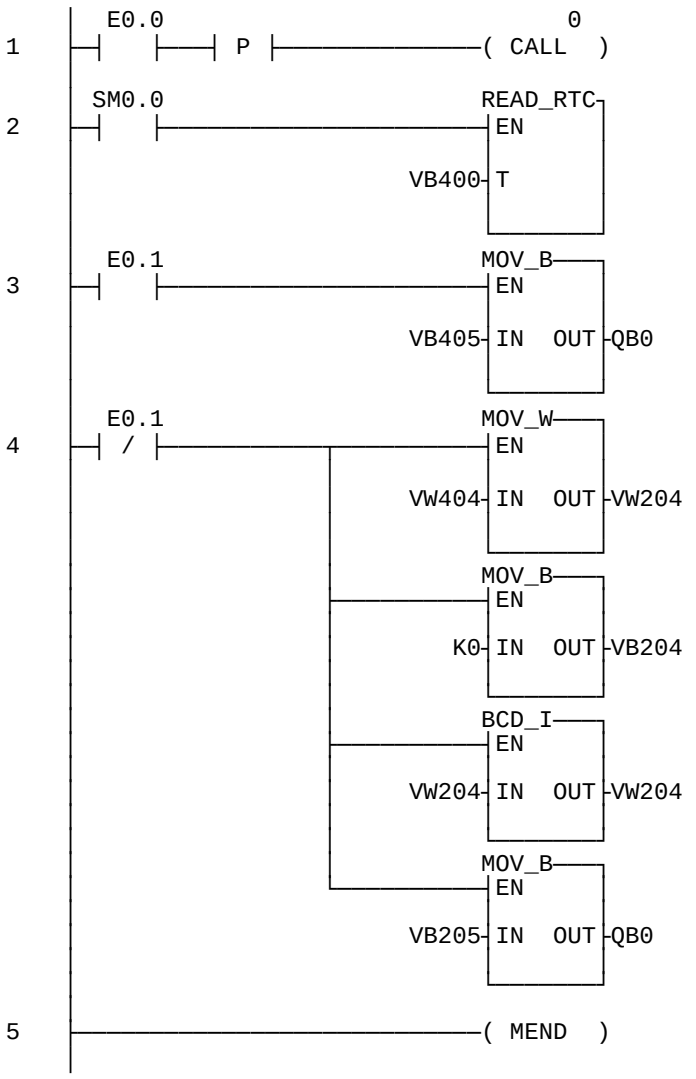
Размер программы составляет 46 слов.

Более подробную информацию о часах реального времени Вы найдете в Главе 5.7 “Операции с часами реального времени” Руководства по программированию SIMATIC S7-200. Детальная информация о конвертировании чисел содержится в Главе 5.5 “Операции преобразования”.

KOP (S7-MicroDOS)	AWL (TOOLITE2)

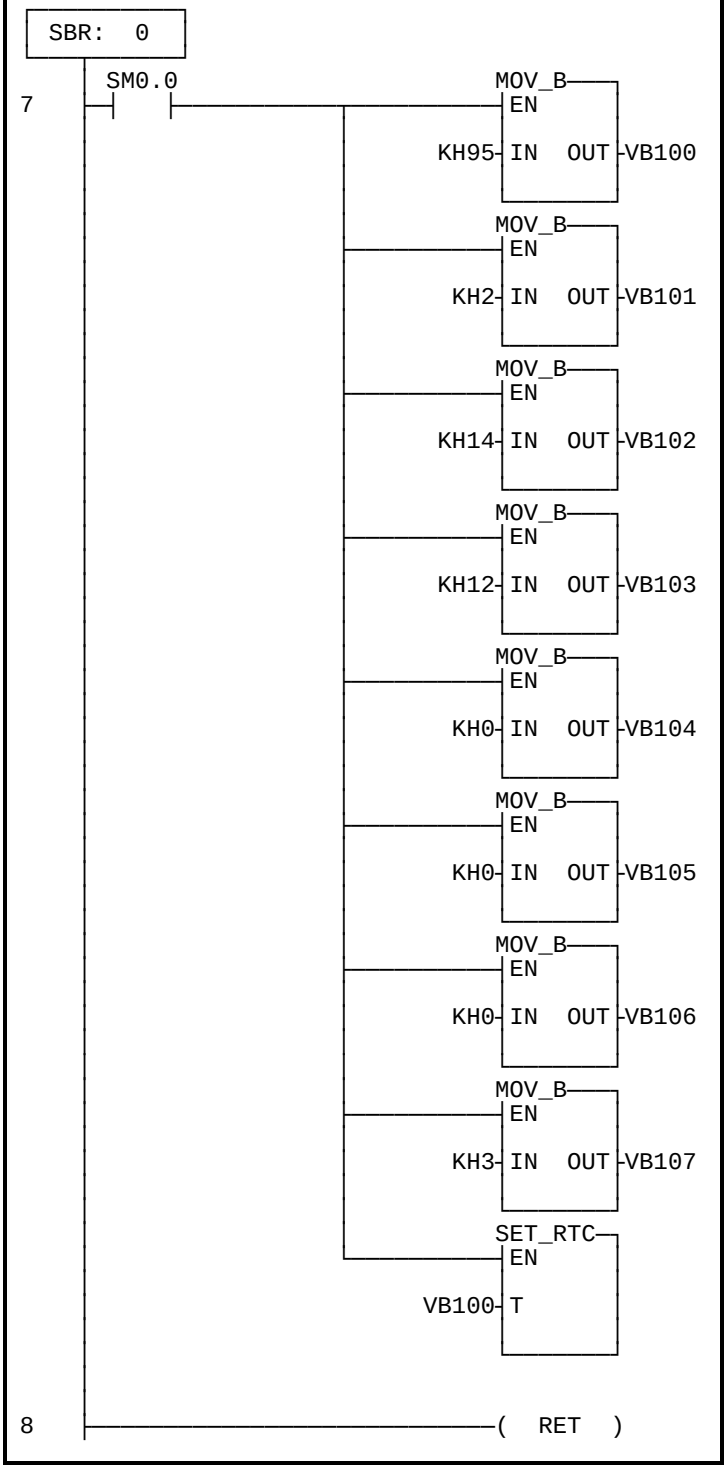
Основная программа

// TITEL = Часы реального времени




```
LD      E0.0        // Кнопка записи
                    // реального времени
EU
CALL    0           // Вызов
                    // подпрограммы 0
LD      SM0.0       // Начало стека
TODR    VB400       // Чтение данных
                    // реального времени
                    // и запись в 8-
                    // байтовый буфер
LD      E0.1        //Кнопка индикации
                    // секунд в
                    // формате BCD
MOVB    VB405, AB0  // Копирование
                    //текущего значения
                    // секунд в
                    // выходной байт
LDN     E0.1
MOVW    VW404, VW204 //Копирование
                    // слова

MOVB    0, VB204    //Стирание старш.
                    // байта (минуты)

BCDI    VW204       // Преобразование
                    // BCD => Двоич.

MOVB    VB205, AB0 // Копирование
                    //текущего значения
                    // секунд в
                    // выходной байт

MEND
```

Подпрограмма




```
SBR    0                  // Установка даты
                          // и времени
LD     SM0.0
MOVB   16#95, VB100       // год: 95

MOVB   16#02, VB101       // месяц: февраль

MOVB   16#14, VB102       // день: 14

MOVB   16#12, VB103       // часы: 12

MOVB   16#0, VB104        // минуты: 00

MOVB   16#0, VB105        // секунды: 00

MOVB   16#0, VB106        // не занято

MOVB   16#3, VB107        // день недели:
                          // вторник

TODW   VB100              // записать время

RET                       // Конец
                          // подпрограммы
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии к строкам, начинающиеся с "**//**" в S7-Micro/DOS не возможны, зато возможны комментарии к сегментам.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

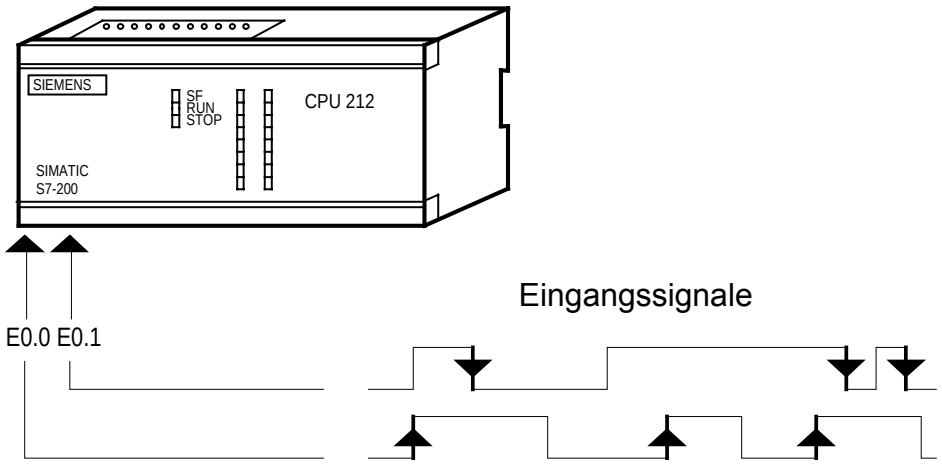
Группа	Пример к теме
2	**Обработка фронтов** входных сигналов

Краткое описание

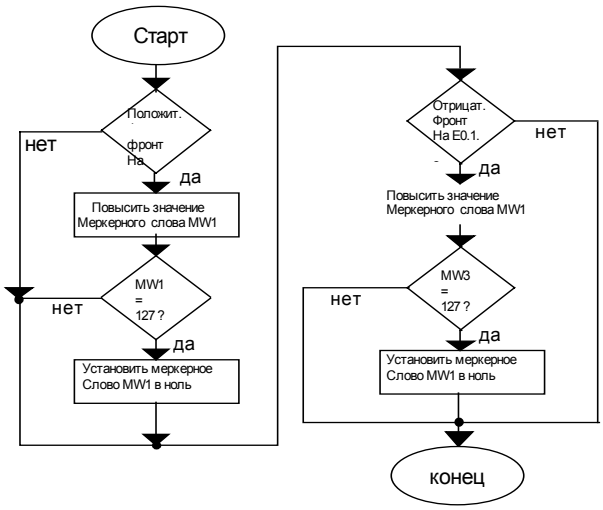
Этот пример показывает, как можно определить смену сигнала с помощью функции определения фронта S7-200. При этом различаются положительные и отрицательные фронты, причем под положительным фронтом понимается смена сигнала с ‘0’ на ‘1’, а под отрицательным - смена сигнала с ‘1’ на ‘0’. Логическая ‘1’ означает, что на вход подано напряжение, а под ‘0’ понимается отсутствие напряжения на входе.

Программа запоминает число положительных фронтов на входе E0.0 и отрицательных на входе E0.1 в слове меркеров. Можно дополнить программу опросом запомненных значений, проверяя при этом все ли фронты опознаны.

Схема включения



Структура программы



Описание программы вкл. листинг

Программа опрашивает вход E0.0 и и проверяет затем с помощью команды EU (Edge Up), поизошла ли положительная смена сигнала, то есть смена с ‘0’ на ‘1’. Если да, то значение меркерного слова MW1 повышается на 1. Отрицательный фронт входа E0.1 будет сосчитан посредством команды ED (Edge Down) iв меркерном слове MW3. Если число сосчитанных фронтов достигает 127, то соответствующий меркер сбрасывается в ноль. Не забывайте, что младший байт слова меркеров MW1 это байт M2, а старший байт это байт M1. В слове меркеров MW3 соответственн младший бит M4, а старший M3.

Размер программы составляет 27 слов.

Более подробную информацию об обработке фронтов Вы найдете В Главе 3.7 “Специальные операции с контактами” в Руководстве по программированию SIMATIC S7-200.

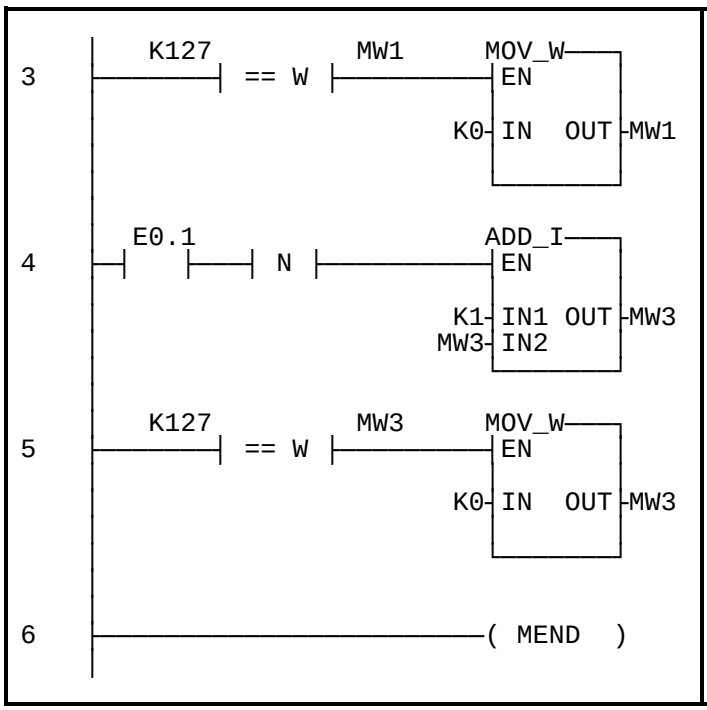
KOP (S7-MicroDOS)	AWL (TOOLITE2)

Основная программа

// TITEL: Обработка фронтов

```
     SM0.1                    MOV_DW
1   ─┤   ├──────────────────┤EN     ├
                             K0-IN   OUT-MD1

     E0.0                     ADD_I
2   ─┤   ├──┤ P ├──────────┤EN     ├
                             K1-IN1  OUT-MW1
                            MW1-IN2
```

```
LD      SM0.1  // Только в первом цикле
               // в единицу
MOVD    0, MD1// Установка двойного
               // слова MD1 в ноль

LD      E0.0   // Положительный фронт
EU
+I      1,MW1
               // При смене фронта:
               // слово меркеров MW1
               // увеличивается на 1
```




```
LDW=    127,MW1
                // После 127 смен фронта
                // (=свободно задаваемое
                //  значение)
MOVW    0,MW1
                // Слово меркеров MW1
                // сбрасывается в 0
LD      E0.1    // Отрицательный фронт
ED
+I      1,MW3
                // При смене фронта:
                // слово меркеров MW3
                // увеличивается на 1

LDW=    127,MW3
                // После 127 смен фронта
                // (=свободно задаваемое
                //  значение)
MOVW    0,MW3
                // Слово меркеров MW3
                // сбрасывается в 0

MEND            // Конец программы
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

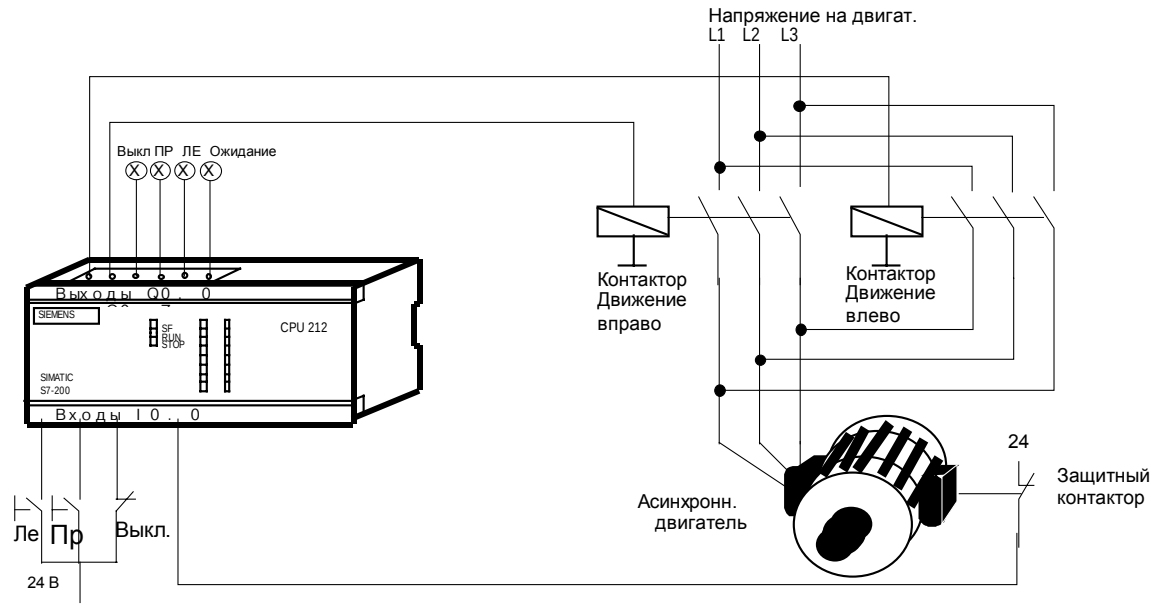
Группа	Пример к теме
4	**Реверсивный переключатель для переключения направления вращения асинхронного двигателя**

Краткое описание

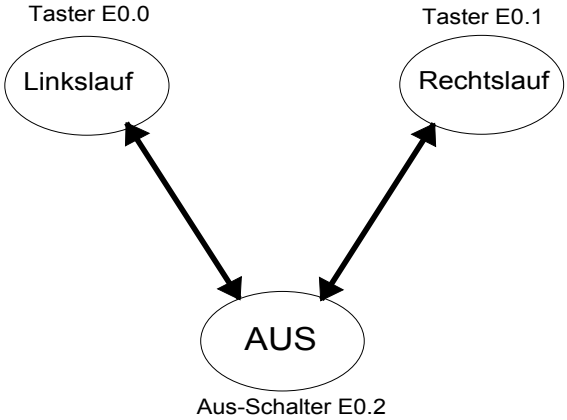
Данный пример программы служит для управления асинхронным двигателем с двума направлениями вращения.

Двигатель может быть запущен на движение влево нажатием кнопки на входе E0.0 или вправо нажатием кнопки на E0.1, при условии, что защитный выключатель на входе E0.3 и основной выключатель на E0.2 не нажаты. Переключить направление вращение можно только после нажатия основного выключателя и окончания времени ожидания 5 с, при этом может происходить торможение и запуск двигателя. Если обе кнопки на включение нажаты одновременно, то двигатель будет блокирован.

Схема включения



Структура программы



Описание программы включая листинг

Сначала программа проверяет, должна ли быть включена блокировка установкой вспомогательного меркера M2.0. Это как раз тот случай когда обе кнопки, на входах E0.0 и E0.1 нажаты одновременно или если время ожидания еще не вышло. Только если обе кнопки снова находятся в отжатом состоянии и кончилось время ожидания, т.е вспомогательный меркер M2.3 сброшен, будет сброшен и блокирующий меркер M2.0.

Разрешение на вращение влево осуществляется установкой соответствующего бита деблокировки M2.1, если ни защитный выключатель двигателя на входе E0.3 или кнопка ВЫКЛ на E0.2 (оба нормально закрытые) не нажаты, ни бит состояния для вращения вправо M1.1 не установлен. Бит состояния для вращения влево это M1.0. Деблокировка вращения вправо происходит аналогично.

Двигатель включается, если нажата одна кнопка включения и ни бит блокировки ни бит состояния другого направления не установлены. Пуск двигателя происходит установкой соответствующего выхода и относящегося к нему бита состояния, обеспечивающего самоподхват. На выходе A0.0 срабатывает контактор левого вращения, а на выходе A0.1 правого. Кроме того загорается сигнальная лампа режима. Сигнальная лампа движения влево подключена на A0.4, движения вправо на A0.3, а лампа состояния ВЫКЛ на A0.2.

Если двигатель выключается, то по положительному фронту ‘ED’ устанавливается вспомогательный меркер M2.3 (процесс выключения). Если последний установлен, то запускается таймер ожидания для повторного пуска со временем работы 5 с (500*10 мс). По завершении этого времени меркер M2.3 процесса выключения сбрасывается. Во время работы таймера ожидания на выходе A0.5 мигает сигнальная лампа. В случае, если бит состояния не установлен, то на выходе A0.2 горит сигнальная лампа состояния ‘ВЫКЛ’.

Размер программы составляет 61 слово.

S7-200 Примеры Реверсивный переключатель (1) Пример-No. 10

KOP (S7-MicroDOS)	AWL (TOOLITE2)

Основная программа

// TITEL = Контакторное переключение

// Блокировка

```
    │  E0.1    E0.0                                  M2.0     K1
1   ├─┤   ├───┤   ├──┬──────────────────────────────────(   S   )
    │                │
    │  M2.3          │
    ├─┤   ├──────────┘
```

LD	**E0.1**	**// Команда вправо**
U	**E0.0**	**// и команда влево**
O	**M2.3**	**// или время ожидания не кончилось**
S	**M2.0,1**	**// устанавливается блокировка**

```
    │  E0.0    E0.1    M2.3                          M2.0     K1
2   ├─┤ / ├───┤ / ├───┤ / ├──────────────────────────(   R   )
    │
```

LDN	**E0.0**	**// Нет команды влево**
UN	**E0.1**	**// нет команды вправо**
UN	**M2.3**	**// время ожидания кончилось**
R	**M2.0, 1**	**// блокировка сбрасывается**

// Деблокировка движения влево

```
    │  E0.2    E0.3    M1.1                              M2.1
3   ├─┤   ├───┤   ├───┤ / ├──────────────────────────(       )
    │
```

LD	**E0.2**	**// Нет команды выключения**
U	**E0.3**	**// не сработал защитный выключатель**
UN	**M1.1**	**// бит состояния движения вправо не установлен**
=	**M2.1**	

// Деблокировка движения вправо

```
    │  E0.2    E0.3    M1.0                              M2.2
4   ├─┤   ├───┤   ├───┤ / ├──────────────────────────(       )
    │
```

LD	**E0.2**	**// Нет команды выключения**
U	**E0.3**	**// не сработал защитный выключатель**
UN	**M1.0**	**// бит состояния движения влево не установлен**
=	**M2.2**	

// Движение влево

```
5   |  E0.0    M2.0    M2.1                                  M1.0
    |-|   |-+-| / |---|   |-------------------+-------------(      )
    |     |                                   |
    |  M1.0 |                                 |              A0.0
    |-|   |-+                                 |-------------(      )
    |                                         |
    |                                         |              A0.4
    |                                         +-------------(      )
    |
```

LD	**E0.0**	**// Команда влево**
O	**M1.0**	**// или состояние левого движения**
UN	**M2.0**	**// и нет блокировки**
U	**M2.1**	**// разрешение на движение влево**
=	**M1.0**	**// бит состояния движения влево**
=	**A0.0**	**// выход контактора**
=	**A0.4**	**// выход сигнальной лампы движения влево**

// Движение вправо

```
6   |  E0.1    M2.0    M2.2                                  M1.1
    |-|   |-+-| / |---|   |-------------------+-------------(      )
    |     |                                   |
    |  M1.1 |                                 |              A0.1
    |-|   |-+                                 |-------------(      )
    |                                         |
    |                                         |              A0.3
    |                                         +-------------(      )
    |
```

LD	**E0.1**	**// Команда вправо**
O	**M1.1**	**// или состояние правого движения**
UN	**M2.0**	**// и нет блокировки**
U	**M2.2**	**// разрешение на движение вправо**
=	**M1.1**	**// бит состояния движения вправо**
=	**A0.1**	**// выход контактора**
=	**A0.3**	**// выход сигнальной лампы движения вправо**

// Определение фронта процесса выключения

```
7   |  M1.0    M1.1                                          A0.2
    |-| / |---| / |-------------------------------------(      )
    |
```

LDN	**M1.0**	**// бит состояния движения влево**
UN	**M1.1**	**// бит состояния движения вправо**
=	**A0.2**	**// выход сигнальной лампы состояния ВЫКЛ**

```
8   |  A0.2                                          M2.3     K1
    |-|   |---| N |-------------------------------------(   S   )
    |
```

```
LD      A0.2                    // При выключении
ED
S  M2.3,1                       // Устанавливается вспомогательный меркер для
                                // процесса выключения
```

```
9   |  M2.3                                          MOV_W
    |--|   |-------------------------------------------|EN
    |      |                                   |
    |      |                                   |   K500-|IN   OUT|-VW20
    |      |                                   |
    |      |                                   |
    |      |                                   |       TON----T33
    |      |                                   |-------------|IN
    |      |                                   |
    |      |                                   |   VW20-|PT
    |      |                                   |
    |      |                                   |
    |      |  T33                                  M2.3      K1
    |      |--|   |-------------------------------------(   R   )
```

```
LD              M2.3
MOVW            500,VW20        // Загрузка времени ожидания для перезапуска
TON             T33,VW20        // Пуск таймера ожидания для перезапуска
U               T33
R               M2.3, 1         // Сброс вспомогательного меркера после времени ожидания
```

```
// Индикация состояния ВЫКЛ, Время ожидания
```

```
10  |  M2.3   SM0.5                                   A0.5
    |--|   |---|   |-------------------------------------(       )
    |
```

```
LD      M2.3                    // Вспомогательный меркер времени ожидания
                                // процесса выключения
U       SM0.5           // Мигание 1 с
=       A0.5                    // Выход сигнальная лампа время ожидания
```

```
11  |-------------------------------------------------------( MEND  )
    |
```

```
MEND                            // Конец основной программы
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

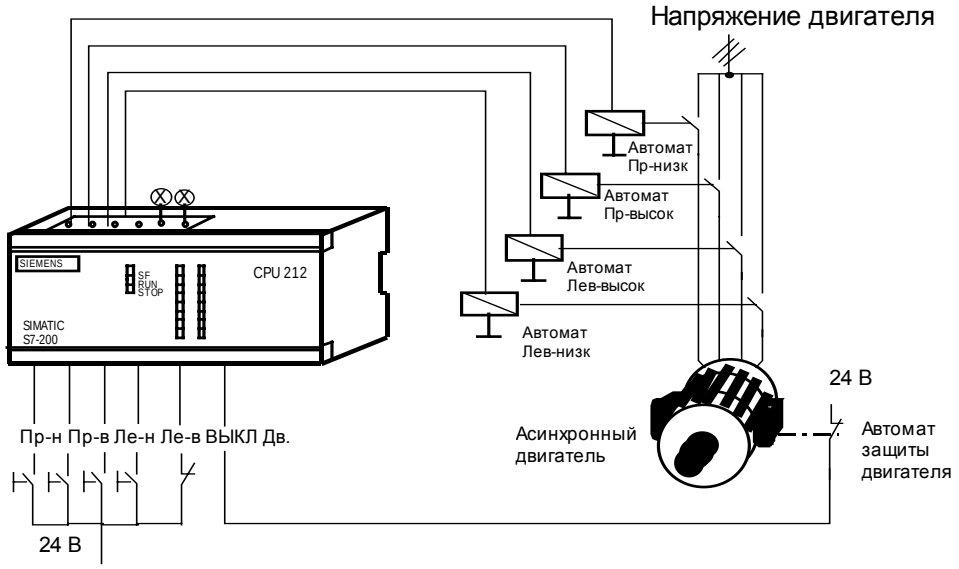
Группа	Пример к теме
4	**Реверсивный переключатель для полюсно-переключаемых асинхронных двигателей с выбором направления вращения**

Краткое описание

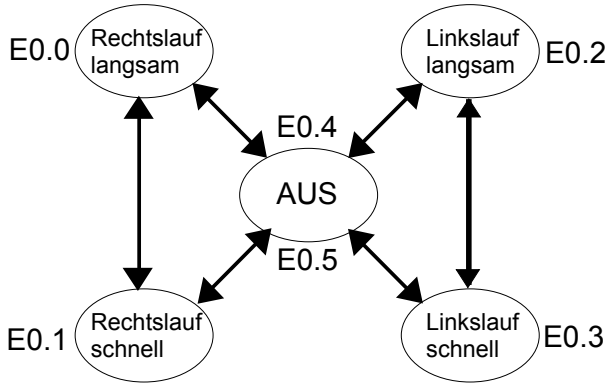
Данный пример служит для управления асинхронным двигателем с двумя раздельными обмотками для различного числа оборотов и двумя возможными направлениями вращения. Пример является дополнением к примеру No. 10.

Включение двигателя производится кнопками на входах, причем в любой момент возможно переключение между большим и меньшим числом оборотов направления вращения. Двигателю необходимо примерно 5 сек, для торможения и запуска в противоположном направлении. Время ожидания индицируется посредством мигания лампы на выходе A0.5.

Схема включения



Структура программы



Описание программы вкл. листинг

Данная программа служит для управления асинхронным двигателем с двумя раздельными обмотками для различных оборотов и двумя возможными направлениями движения.
Она является дополнением к контакторной схеме из примера No. 10. В этой программе можно дополнительно выбирать между меньшим и большим числом оборотов для каждого направления вращения.

Включение двигателя производится кнопками на входах

E0.0 для включения медленного вращения вправо,
E0.1 для включения быстрого вращения вправо,
E0.2 для включения медленного вращения влево и
E0.3 для включения быстрого вращения влево.
После включения возможно в любой момент времени произвести переключение между быстрым и медленным вращением в любом направлении. При переключении направления вращения необходимо сначала нажать кнопку ВЫКЛ. Двигателю необходимо примерно 5 сек для торможения и запуска в противоположном направлении. Время ожидания индицируется миганием лампочки на выходе A0.5.
Переключатель на E0.4 служит для выключения двигателя. На вход E0.5 подключен автомат защиты двигателя, который размыкается при перегрузке двигателя и производит отключение двигателя. При отключенном двигателе горит сигнальная лампа на выходе A0.4.

В начале цикла програмы вызывается подпрограмма ‘SBR 1’, чьей задачей является оценка сигналов на входах. Для того чтобы исключить ошибку сначала проверяется, не нажато ли два или более переключателя одновременно или закончено ли время ожидания при переключении. В этих случаях меркер блокировки M1.0 устанавливается в ‘1’, исключая тем самым деблокировку автомата защиты двигателя. Блокировка позволяет избежать неконтролируемого движения двигателя или движения в неверном направлении. Блокировка будет снята, т.е. M1.0 установится в логический ‘0’, только если все кнопки снова находятся в ненажатом положении или окончено время ожидания.

Деблокировка двигателя для режима работы производится только если ВЫКЛючатель (нормально замкнутый) на E0.4 не нажат, автомат защиты двигателя (нормально замкнутый) на E0.5, контролирующий перегрузку двигателя, не разомкнут и двигатель не вращается в противоположном направлении (бит состояния противоположного направления не установлен). Деблокировка снимается также, если подана команда на изменение числа оборотов. Тем самым гарантируется, что два переключателя, для быстрого и медленного движения в одном направлении, не смогут быть активированы одновременно. Например, выход A0.2, который отвечает за медленное вращение влево, будет сброшен, если нажатием кнопки на E0.3 подана команда быстрого вращения влево.

Самоподхват включенного режима работы, т.е. установка соответствующих битов состояния A0.0, A0.1 A0.2 или A0.3, возможен, если произошла деблокировка желаемого режима работы и не активна общая блокировка.

Определение фронта в конце подпрограммы служит для запуска времени ожидания после выключния двигателя, с тем чтобы двигатель имел время на торможение.

При выходе из подпрограммы биты состояния копируются на выходы с A0.0 по A0.3. Эти выходы управляют автоматами двигателя, в то время как выходы A0.4 и A0.5 служат для индикации состояния ‘ВЫКЛ’ или ‘Идет время ожидания’.

Размер программы составляет 125 слов.

KOP (S7-MicroDOS)	AWL (TOOLITE2)

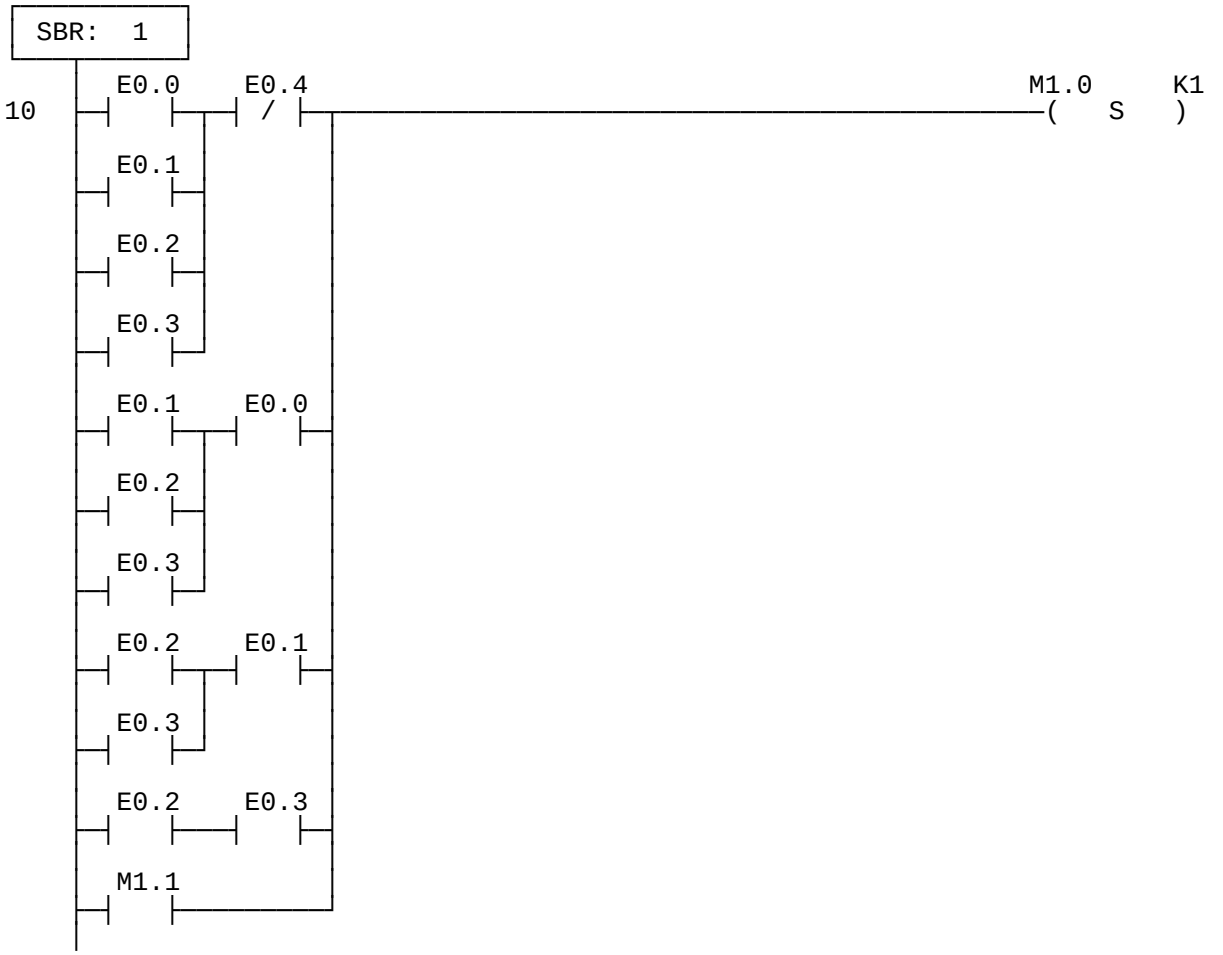
Основная программа

```
// TITEL = Полюсно-переключаемый асинхронный двигатель с двумя направлениями вращения

//************ ВХОДЫ ****************************
// E0.0   Медленное вращение вправо
// E0.1   Быстрое вращение вправо
// E0.2   Медленное вращение влево
// E0.3   Быстрое вращение влево
// E0.4   Кнопка ВЫКЛ (нормально замкнутый)
// E0.5   Автомат защиты двигателя (нормально замкнутый)

//**************** ВЫХОДЫ ************************
// A0.0   Переключатель медленного вращения вправо
// A0.1   Переключатель быстрого вращения вправо
// A0.2   Переключатель медленного вращения влево
// A0.3   Переключатель быстрого вращения влево
// A0.4   Сигнальная лампа ВЫКЛ
// A0.5   Сигнальная лампа Время ожидания
```

```
     | SM0.0                        1
1    |-|    |-------------------( CALL  )
     |
     |
     |  E0.0                  A0.1      K1
2    |-|    |-------------------(   R    )
     |
     |
     |  E0.1                  A0.0      K1
3    |-|    |-------------------(   R    )
     |
     |
     |  E0.2                  A0.3      K1
4    |-|    |-------------------(   R    )
     |
     |
     |  E0.3                  A0.2      K1
5    |-|    |-------------------(   R    )
     |
     |
     |  A0.0  A0.1  A0.2  A0.3   A0.4
6    |-| / |-| / |-| / |-| / |-(        )
     |
     |
     |  M1.1   SM0.5              A0.5
7    |-|    |---|    |----------(        )
     |
     |
     |
8    |-------------------------( MEND  )
     |
```

```
LD      SM0.0 // Устан. начала стека
CALL 1        // Вызов
              // подпрограммы 1

LD      E0.0  // Команда медленного
              // вращения вправо
R       A0.1, 1
              // Сброс вправо быстро

LD      E0.1  // Команда вправо быстро
R       A0.0, 1// Сброс вправо медленно

LD      E0.2 // Команда влево медленно
R       A0.3, 1// Сброс влево быстро

LD      E0.3  // Команда влево быстро
R       A0.2, 1// Сброс влево медленно

LDN     A0.0
UN      A0.1
UN      A0.2
UN      A0.3
=       A0.4
// ИНДИКАЦ.ВРЕМЕНИ ОЖИДАНИЯ
LD      M1.1  // Всп.меркер Вр.ожидан.
              // после выключения
U       SM0.5 // Мигание 1
=       A0.5  // Выход Сигн.лампа
              // Время ожидания

MEND          // Конец
              // основной программы
```

Подпрограмма




```
SBR 1                          // Начало подпрограммы

//***************VERRIEGELUNG****************************
LD      E0.0                   // Медленное движение вправо
O       E0.1                   // Или быстрое движение вправо
O       E0.2                   // Или медленное движение влево
O       E0.3                   // Или быстрое движение влево
UN      E0.4                   // И нажата кнопка ВЫКЛ

LD      E0.1                   // Или...
O       E0.2
O       E0.3
U       E0.0
OLD
LD      E0.2                   // Или...
O       E0.3
U       E0.1
OLD
LD      E0.2                   // Или...
U       E0.3
OLD
LD      M1.1                   // Или идет время ожидания
OLD
S       M1.0,1                 // То установить меркер блокировки
```

//**********ДЕБЛОКИРОВКА**********

```
     |  E0.4    E0.0    E0.1    E0.2    E0.3    M1.1              M1.0     K1
11   |--|    |----| / |----| / |----| / |----| / |----| / |----------------(   R   )
     |
```

LD	E0.4	// Если ни одна кнопка не нажата...
UN	E0.0	
UN	E0.1	
UN	E0.2	
UN	E0.3	
UN	M1.1	
R	M1.0,1	// То блокировку снять

// РАЗРЕШЕНИЕ МЕДЛЕННОГО ВРАЩЕНИЯ ВПРАВО

```
     |  E0.1    E0.4    E0.5    A0.2    A0.3                          M2.0
12   |--| / |----|    |----|    |----| / |----| / |-----------------------(       )
     |
```

LDN	E0.1	// Нет команды быстро
LD	E0.4	// Нет команды выкл
U	E0.5	// Не сработал автомат защиты двигателя
UN	A0.2	// Бит состояния медленное вращение влево не установлен
UN	A0.3	// Бит состояния быстрое вращение влево не установлен
=	M2.0	// Деблокировка медленного вращения вправо

// РАЗРЕШЕНИЕ БЫСТРОГО ВРАЩЕНИЯ ВПРАВО

```
     |  E0.0    E0.4    E0.5    A0.2    A0.3                          M2.1
13   |--| / |----|    |----|    |----| / |----| / |-----------------------(       )
     |
```

LDN	E0.0	// Нет команды медленно
LD	E0.4	// Нет команды выкл
U	E0.5	// Не сработал автомат защиты двигателя
UN	A0.2	// Бит состояния медленное вращение влево не установлен
UN	A0.3	// Бит состояния быстрое вращение влево не установлен
=	M2.1	// Деблокировка быстрого вращения вправо

// РАЗРЕШЕНИЕ МЕДЛЕННОГО ВРАЩЕНИЯ ВЛЕВО

```
    |  E0.3    E0.4    E0.5    A0.0    A0.1                          M2.2
14  |--| / |----|   |----|   |----| / |----| / |-----------------------(       )
    |
```

LDN	**E0.3**	**// Нет команды быстро**
LD	**E0.4**	**// Нет команды выкл**
U	**E0.5**	**// Не сработал автомат защиты двигателя**
UN	**A0.0**	**// Бит состояния медленное вращение вправо не установлен**
UN	**A0.1**	**// Бит состояния быстрое вращение вправо не установлен**
=	**M2.2**	**// Деблокировка медленного вращения влево**

// РАЗРЕШЕНИЕ БЫСТРОГО ВРАЩЕНИЯ ВЛЕВО

```
    |  E0.2    E0.4    E0.5    A0.0    A0.1                          M2.3
15  |--| / |----|   |----|   |----| / |----| / |-----------------------(       )
    |
```

LDN	**E0.2**	**// Нет команды медленно**
LD	**E0.4**	**// Нет команды выкл**
U	**E0.5**	**// Не сработал автомат защиты двигателя**
UN	**A0.0**	**// Бит состояния медленное вращение вправо не установлен**
UN	**A0.1**	**// Бит состояния быстрое вращение вправо не установлен**
=	**M2.3**	**// Деблокировка быстрого вращения влево**

```
    |  E0.0    M1.0    M2.0                                          A0.0
16  |--|   |---| / |----|   |-----------------------------------------(       )
    |      |
    |  A0.0 |
    |--|   |--|
    |
```

LD	**E0.0**	**// Команда вправо медленно**
O	**A0.0**	**// Или состояние вправо медленно**
UN	**M1.0**	**// И нет блокировки**
U	**M2.0**	**// Разрешение вправо медленно**
=	**A0.0**	**// Бит состояния Вправо медленно**

```
    |  E0.1    M1.0    M2.1                                          A0.1
17  |--|   |---| / |----|   |-----------------------------------------(       )
    |      |
    |  A0.1 |
    |--|   |--|
    |
```

LD	**E0.1**	**// Команда вправо быстро**
O	**A0.1**	**// Или состояние вправо быстро**
UN	**M1.0**	**// И нет блокировки**
U	**M2.1**	**// Разрешение вправо быстро**
=	**A0.1**	**// Бит состояния Вправо быстро**

```
18  |  E0.2    M1.0    M2.2                                    A0.2
    |-|   |-+-| / |--|   |-----------------------------------(      )
    |     |
    |  A0.2 |
    |-|   |-|
    |
```

LD	**E0.2**	**// Команда влево медленно**
O	**A0.2**	**// Или состояние влево медленно**
UN	**M1.0**	**// И нет блокировки**
U	**M2.2**	**// Разрешение влево медленно**
=	**A0.2**	**// Бит состояния Влево медленно**

```
19  |  E0.3    M1.0    M2.3                                    A0.3
    |-|   |-+-| / |--|   |-----------------------------------(      )
    |     |
    |  A0.3 |
    |-|   |-|
    |
```

LD	**E0.3**	**// Команда влево быстро**
O	**A0.3**	**// Или состояние влево быстро**
UN	**M1.0**	**// И нет блокировки**
U	**M2.3**	**// Разрешение влево быстро**
=	**A0.3**	**// Бит состояния Влево быстро**

```
20  |  E0.4                                                  M1.1     K1
    |-|   |--| P |-------------------------------------------(   S   )
    |
```

LD	**A0.4**	**// Определение фронта при выключении**
EU		
S	**M1.1,1**	**// Установка вспомогательного меркера для выключения**

```
21  |  M1.1                                                  MOV W---+
    |-|   |-+-------------------------------------------------|EN     |
    |     |                                                 |       |
    |     |                                                 K500-IN  OUT-VW20
    |     |                                                 |       |
    |     |                                                 |       |
    |     |                                                 TON---T33
    |     |                                                 |IN     |
    |     |                                                 |       |
    |     |                                                 VW20-PT  |
    |     |                                                 |       |
    |     |                                                 |       |
    |     |  T33                                            M1.1     K1
    |     |-|   |-------------------------------------------(   R   )
    |
```

LD	**M1.1**	
MOVW	**500,VW20**	**// Загрузка времени ожидания для перезапуска**
TON	**T33,VW20**	**// Пуск таймера ожидания для перезапуска**
U	**T33**	
R	**M1.1,1**	**// Сброс всп. меркера по прошествии времени ожидания**

```
    |
```

22	(RET)
RET	**// Конец подпрограммы**

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

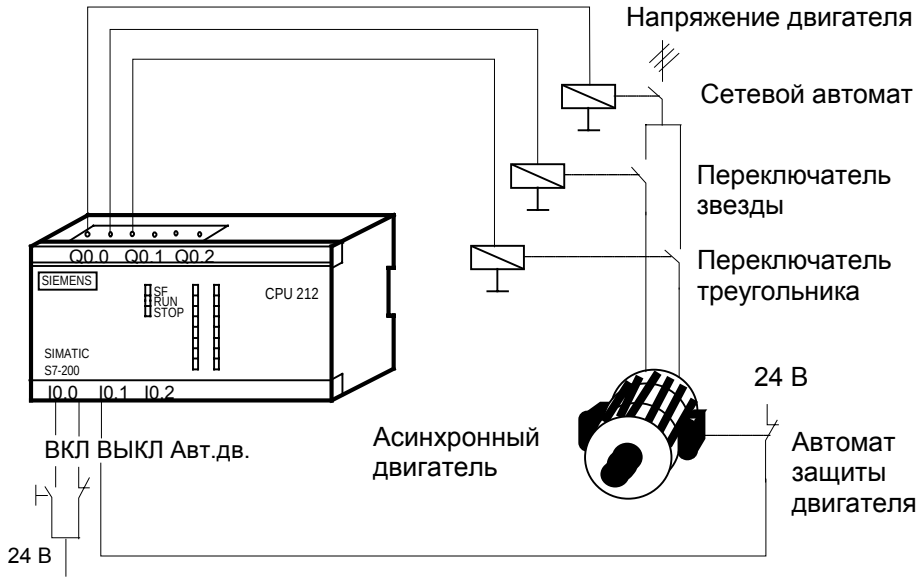
Группа	Пример к теме
4	Запуск звезда-треугольник

Краткое описание

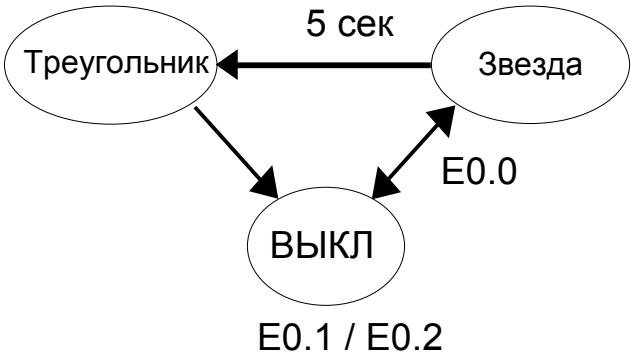
Данный пример управляет запуском звезда-треугольник асинхронного двигателя. Двигатель запускается после нажатия кнопки ВКЛ на входе E0.0 в режиме звезды. По прошествии установленного времени 5сек двигатель переключается в режим треугольник.

Если нажата кнопка ВЫКЛ или сработал автомат защиты двигателя, находящиеся на входах E0.1 и E0.2, двигатель выключается. Если одновременно нажаты кнопки ВЫКЛ и ВКЛ, то двигатель остается выключенным.

Схема включения



Структура программы



Описание программы вкл. листинг

В начале цикла программа проверяет, должен ли быть установлен вспомогательный меркер M10.0 для блокировки. Это будет сделано, если Кнопки ВЫКЛ (E0.1) и ВКЛ (E0.0) нажаты одновременно. Блокировка будет снята только, если обе кнопки снова находятся в исходном состоянии. Блокировка служит для защиты от неверных условий.

Для процесса включения имеется вспомогательный меркер M11.0. Он будет установлен при включении, т.е. если кнопка на входе E0.0 замкнута и переключатель сети еще не сработал. Он будет также установлен, если двигатель находится в режиме звезды, т.е. переключатель сети (A0.0) и переключатель звезды (A0.1) активны. Установка вспомогательного меркера M11.0 происходит впрочем только тогда, если не разомкнуты автомат защиты двигателя (E0.2) или кнопка ВЫКЛ (E0.1), а переключатель треугольника (A0.2) неактивен.

Выход A0.1 переключателя звезды будет установлен, если установлен вспомогательный меркер запуска M11.0, отсутствует блокировка (M10.0) и время переключения (5 сек таймера T37) еще не закончилось.
Время переключения запускается в момент установки вспомогательного меркера запуска M11.0, в случае если отсутствует блокировка. Временной базис таймера T37 - 100 мсек, т.е. при загрузке значения 50 время переключения составит 5 сек.

Выход A0.0 переключателя сети устанавливается, если или кнопка ВКЛ на E0.0, выход A0.1 переключателя звезды или переключатель сети уже активны и нет команды ВЫКЛ от кнопки на E0.1 или сработала защита двигателя на E0.2 и отсутствует блокировка.

Переключатель треугольника, т.е. выход A0.2, устанавливается при активном переключателе сети и уже неактивном переключателе звезды.

Размер программы составляет 40 слов.

KOP (S7-MicroDOS)	AWL (IEC)

Основная программа

```
// TITEL =  ЗАПУСК ЗВЕЗДА-ТРЕУГОЛЬНИК БЕЗ ОТВЕТНОГО СООБЩЕНИЯ

// E0.0      Кнопка вкл.
// E0.1      Кнопка выкл. Нормально замкнута
// E0.2      Защита двигателя Нормально замкнута

// A0.0      Переключатель сети
// A0.1      Переключатель звезды
// A0.2      Переключатель треугольника
// T37       Таймер переключения 5 сек


// БЛОКИРОВКА
```

```
    |  E0.1    E0.0                                          M10.0     K1
1   |--| / |----|   |-----------------------------------------(   S   )
    |
```

```
LDN     E0.1        // Кнопка Выкл нажата
U       E0.0        // Кнопка Вкл нажата
S       M10.0 ,1    // Вспомогательный меркер для блокировки
```

```
    |  E0.1    E0.0                                          M10.0     K1
2   |--|   |----| / |-----------------------------------------(   R   )
    |
```

```
LD      E0.1        // Кнопка Выкл не нажата
UN      E0.0        // Кнопка Вкл не нажата
R       M10.0 ,1    // Блокировка снята
```

```
// ВКЛЮЧЕНИЕ
```

```
    |  E0.0    A0.0    E0.2    E0.1    A0.2                    M11.0
3   |--|   |----| / |---|   |----|   |----| / |-------------------(       )
    |                |
    |  A0.0    A0.1  |
    |--|   |----|   |--|
    |
```

```
LD      E0.0        // Кнопка Вкл нажата
UN      A0.0        // Нет переключателя сети
LD      A0.0        // Переключатель сети
U       A0.1        // Переключатель звезды
OLD
```

```
LD      E0.2        // Защита двигателя OK
U       E0.1        // Кнопка ВЫКЛ не нажата
UN      A0.2        // Нет переключателя треугольника
ULD
=       M11.0       // Вспомогательный меркер
```

// ВКЛЮЧЕНИЕ ПЕРЕКЛЮЧАТЕЛЯ ЗВЕЗДЫ

```
    | M11.0   M10.0    T37                                    A0.1
4   |-|    |---| / |---| / |-----------------------------(        )
    |
```

```
LD      M11.0       // Вспомогательный меркер
UN      M10.0       // Нет блокировки
UN      T37         // Время переключения не закончилось
=       A0.1        // Переключатель звезды
```

// ЗАПУСК ТАЙМЕРА ПЕРЕКЛЮЧЕНИЯ

```
    | M10.0   M11.0                                        TON---T37
5   |-| / |---|    |-----------------------------------|IN      |
    |                                                    |        |
    |                                                K50-|PT      |
    |                                                    |        |
    |                                                    |_______|
```

```
LDN     M10.0       // Нет блокировки
U       M11.0       // Вспомогательный меркер
TON     T37,50      // Запуск таймера переключения (5 сек)
```

// ВКЛЮЧЕНИЕ ПЕРЕКЛЮЧАТЕЛЯ СЕТИ

```
    | E0.1    E0.2   M10.0    E0.0    A0.1                    A0.0
6   |-|    |---|    |---| / |---|    |---|    |-------------(        )
    |                      |                |
    |                      | A0.0           |
    |                      |-|    |---------|
    |
```

```
LD      E0.1        // Кнопка ВЫКЛ не нажата
U       E0.2        // Защита двигателя OK
UN      M10.0       // Нет блокировки
LD      E0.0        // Кнопка Вкл нажата
U       A0.1        // Переключатель звезды
O       A0.0        // Переключатель сети
ULD
=       A0.0        // Переключатель сети
```

// ВКЛЮЧЕНИЕ ПЕРЕКЛЮЧАТЕЛЯ ТРЕУГОЛЬНИКА

```
7   |  A0.0    A0.1                                                  A0.2
    |--|    |----| / |-------------------------------------------(        )
    |
```

LD	**A0.0**	**// Переключатель сети**
UN	**A0.1**	**// Переключатель звезды**
=	**A0.2**	**// Переключатель треугольника**

```
8   |
    |-------------------------------------------------------------( MEND  )
    |
```

MEND **// Конец основной программы**

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведённого здесь.

За правильную работу системы заказчик несёт ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

Copyright ©1996 by SIEMENS

SIMATIC
S7-200 Примеры

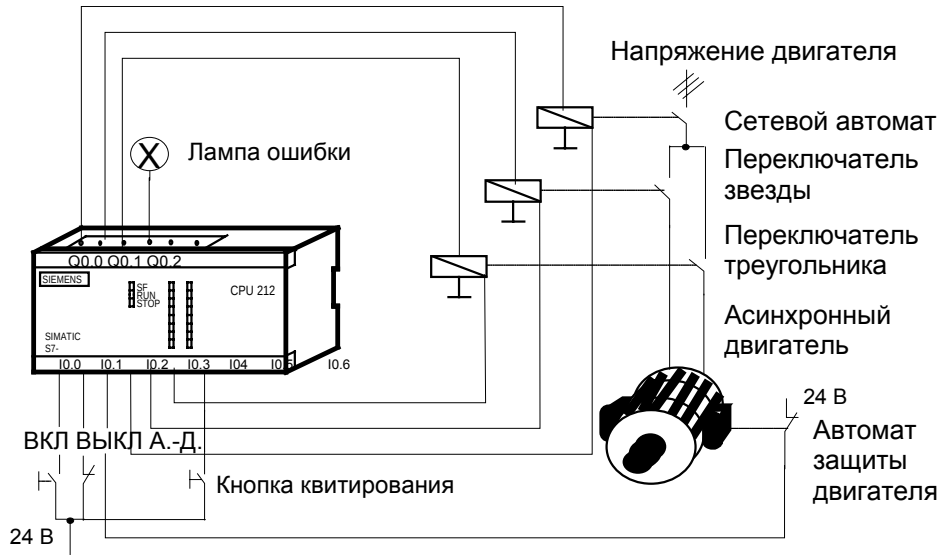
Группа	Пример к теме
4	Запуск звезда-треугольник с ответным сообщением

Краткое описание

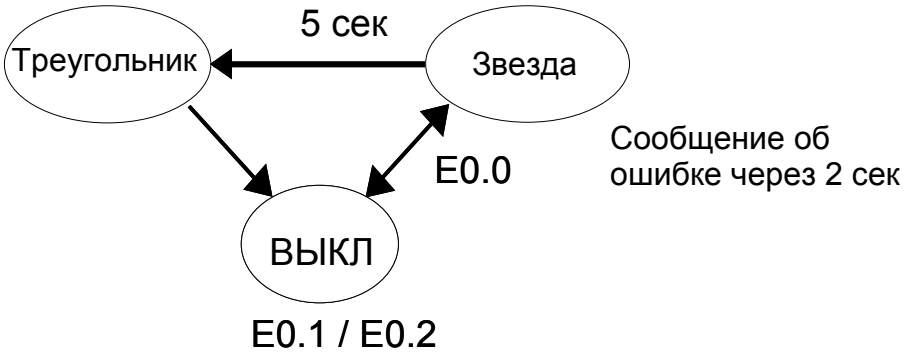
Данный пример управляет запуском звезда-треугольник асинхронного двигателя. Пример No. 12 дополнен ответным сообщением действительного состояния переключателей. Если, например, переключатель звезды дефектен, то это будет опознано с ответным сообщением и SIMATIC S7-200 не переключится через время задержки 5 сек в режим треугольника, исключая тем самым возможность поломки.

Двигатель запускается после нажатия кнопки ВКЛ на входе E0.0 в режиме звезды. По прошествии установленного времени двигатель переключается в режим треугольник. Возможное сообщение об ошибке индицируется посредством лампы на выходе A0.3.

Схема включения



Структура программы



Описание программы вкл. листинг

Двигатель запускается после нажатия кнопки ВКЛ на входе E0.0 в режиме звезды. По прошествии установленного времени 5 сек двигатель переключается в режим треугольник, если отсутствует сообщение об ошибке переключателей. Сообщение об ошибке индицируется лампой на выходе A0.3. Кнопкой квитирования на входе E0.6 можно сбросить сообщение об ошибке, в случае если ошибка устранена. Ответные сообщения переключателей подаются на входы с E0.3 по E0.5.

Если нажата кнопка ВЫКЛ или сработал автомат защиты двигателя, находящиеся на входах E0.1 и E0.2, двигатель отключается. Если кнопки ВЫКЛ и ВКЛ нажаты одновременно, двигатель остается выключенным.

Части программы ”Блокировка и деблокировка”, ”Включение” и ”Включение переключателя треугольника” полностью совпадают с примером No.12, поэтому для них не делается пояснений. Части программы ”Включение переключателя звезда”, ”Запуск таймера переключения” и ”Включение переключателя сети” дополнены условием, что отсутствует сообщение об ошибке (A0.3), что позволит установить меркер для переключателя звезды (A0.1) и переключателя сети (A0.0) и запустить таймер переключения (T37).

Появилась новая часть программы ”Сообщение об ошибке”. В принципе обработка сообщения действует таким образом, что сравнивается состояние выходноых сигналов со входами, на которые подано действительное состояние переключателей.

Состояние этих выходов сравнивается с состоянием входа E0.3 для переключателя сети, E0.4 для переключателя звезды и E0.5 для переключателя треугольника. При отклонении будет включен таймер T38 с временем задержки 2 сек, соответствующим максимальному времени переключения автоматов.

Если по прошествии данного времени состояния все еще различаются, устанавливыается выход ошибки A0.3. Кнопкой квитирования на входе E0.6 он может быть снова сброшен.

Размер программы составляет 70 слов.

KOP (S7-MicroDOS)	AWL (IEC)

Основная программа

```
// TITEL = ЗАПУСК ЗВЕЗДА-ТРЕУГОЛЬНИК С ОТВЕТНЫМ СООБЩЕНИЕМ

// E0.0      Кнопка вкл.
// E0.1      Кнопка выкл. Нормально замкнута
// E0.2      Защита двигателя Нормально замкнута // E0.3
// E0.4      RM. Переключатель звезды
// E0.5      RM. Переключатель треугольника
// E0.6      Кнопка квитирования

// A0.0      Переключатель сети
// A0.1      Переключатель звезды
// A0.2      Переключатель треугольника
// A0.3      Лампа ошибки

// T37       Таймер переключения 5 сек
// T38       Таймер сообщения об ошибке 2 сек
```

```
// ************ БЛОКИРОВКА ****************************
```

```
    │  E0.1    E0.0                                                M10.0    K1
1   ├─┤ / ├───┤   ├──────────────────────────────────────────────────(   S   )
    │
```

```
LDN     E0.1        // Кнопка Выкл нажата
U       E0.0        // Кнопка Вкл нажата
S       M10.0 ,1    // Вспомогательный меркер для блокировки
```

```
    │  E0.1    E0.0                                                M10.0    K1
2   ├─┤   ├───┤ / ├──────────────────────────────────────────────────(   R   )
    │
```

```
LD      E0.1        // Кнопка Выкл не нажата
UN      E0.0        // Кнопка Вкл не нажата
R       M10.0 ,1    // Блокировка снята
```

// ********** ВКЛЮЧЕНИЕ **********

```
3   |  E0.0    A0.0    E0.2    E0.1    A0.2                          M11.0
    |--|    |----| / |--|--|    |----|    |----| / |-----------------------(        )
    |                 |
    |  A0.0    A0.1   |
    |--|    |----|    |--|
    |
```

LD	E0.0	// Кнопка Вкл нажата
UN	A0.0	// Нет переключателя сети
LD	A0.0	// Переключатель сети
U	A0.1	// Переключатель звезды
OLD		
LD	E0.2	// Защита двигателя OK
U	E0.1	// Кнопка ВЫКЛ не нажата
UN	A0.2	// Нет переключателя треугольника
ULD		
=	M11.0	// Вспомогательный меркер

// ********** ВКЛЮЧЕНИЕ ПЕРЕКЛЮЧАТЕЛЯ ЗВЕЗДЫ **********

```
4   |  M11.0   M10.0    T37     A0.3                                  A0.1
    |--|    |----| / |----| / |----| / |-----------------------------(        )
    |
```

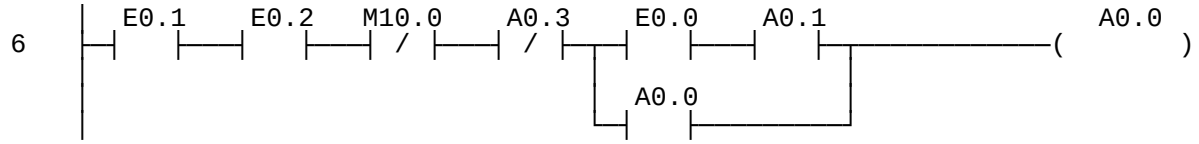
LD	M11.0	// Вспомогательный меркер
UN	M10.0	// Нет блокировки
UN	T37	// Время переключения не закончилось
UN	A0.3	// Нет сообщения об ошибке
=	A0.1	// Переключатель звезды

// ********** ЗАПУСК ТАЙМЕРА ПЕРЕКЛЮЧЕНИЯ **********

```
5   |  M10.0   M11.0    A0.3                                        TON---T37
    |--| / |----|    |----| / |-----------------------------------|IN      |
    |                                                              |        |
    |                                                        K50-|PT      |
    |                                                              |        |
    |                                                              |________|
```

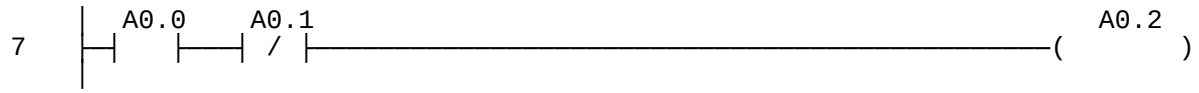
LDN	M10.0	// Нет блокировки
U	M11.0	// Вспомогательный меркер
UN	A0.3	// Нет сообщения об ошибке
TON	T37,50	// Запуск таймера переключения (5 сек)

// ********** ВКЛЮЧЕНИЕ ПЕРЕКЛЮЧАТЕЛЯ СЕТИ **********




```
LD      E0.1      // Кнопка ВЫКЛ не нажата
U       E0.2      // Защита двигателя ОК
UN      M10.0     // Нет блокировки
UN      A0.3      // Нет сообщения об ошибке
LD      E0.0      // Кнопка Вкл нажата
U       A0.1      // Переключатель звезды
O       A0.0      // Переключатель сети
ULD
=       A0.0      // Переключатель сети
```

// ********** ВКЛЮЧЕНИЕ ПЕРЕКЛЮЧАТЕЛЯ ТРЕУГОЛЬНИКА **********




```
LD      A0.0      // Переключатель сети
UN      A0.1      // Переключатель звезды
=       A0.2      // Переключатель треугольника
```

// **********СООБЩЕНИЕ ОБ ОШИБКЕ**********

```
LD      A0.0        // Переключатель сети
UN      E0.3        // RM. Переключатель сети
LDN     A0.0        // Нет переключателя сети
U       E0.3        // RM. Переключатель сети
OLD
LD      A0.1        // Переключатель звезды
UN      E0.4        // Нет RM. Переключателя звезды
OLD
LDN     A0.1        // Нет переключателя звезды
U       E0.4        // RM. Переключатель звезды
OLD
LD      A0.2        // Переключатель треугольника
UN      E0.5        // Нет RM. Переключателя треугольника
OLD
LDN     A0.2        // Нет переключателя треугольника
U       E0.5        // RM. Переключатель треугольника
OLD
TON     T38,20      // Таймер задержки для сообщения об ошибке (2 сек)
```

```
    |  T38                                                    A0.3    K1
9   |--|   |-------------------------------------------------------(   S   )
    |
```

```
LD      T38
S       A0.3, 1     // Ошибка
```

```
    |  E0.6                                                   A0.3    K1
10  |--|   |-------------------------------------------------------(   R   )
    |
```

```
LD      E0.6        // Кнопка квитирования
R       A0.3, 1
```

```
    |
11  |-------------------------------------------------------------( MEND  )
    |
```

MEND

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведённого здесь.

За правильную работу системы заказчик несёт ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

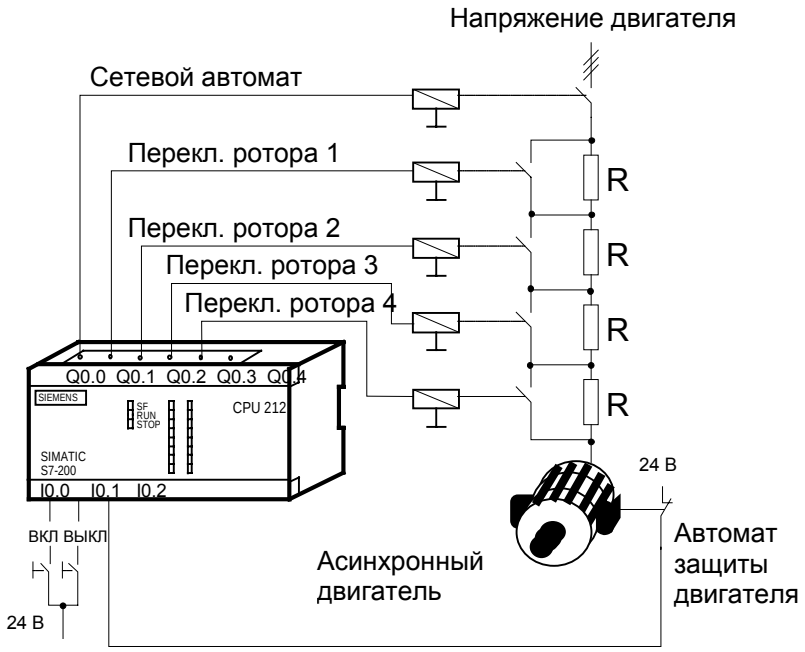
Группа	Пример к теме
4	Фазный ротор

Краткое описание

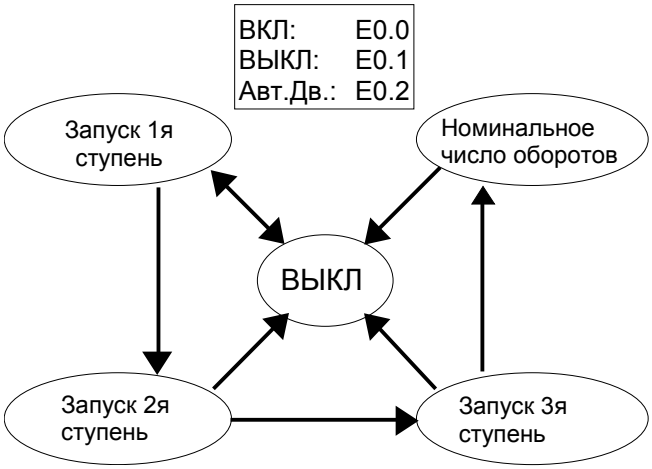
В данном примере описан автоматический запуск асинхронного двигателя с фазным ротором из состоящий четырех ступеней. Двигатель при этом запускается с полным сопротивлением ротора. По прошествии определенного времени включается первый переключатель ротора и перемыкает часть сопротивления ротора. Через каждый последующий интервал времени включаются следующие переключатели, причем сопротивление ротора каждый раз уменьшается, пока не будет полностью перемкнуто и двигатель выходит на номинальные обороты.

Кнопка ВКЛ на входе E0.0 осуществляет плавный пуск двигателя, а кнопка ВЫКЛ на E0.1 отключает двигатель. На вход E0.2 подключен автомат защиты двигателя, который срабатывает при перегрузке двигателя и отключает его.

Схема включения



Структура программы



Описание программы вкл. листинг

Если нажата кнопка ВКЛ на входе E0.0 и не активны переключатели ротора или уже выполнен самоподхват автомата защиты двигателя (A0.0 = 1), то устанавливается промежуточный меркер M2.0. Этот промежуточный меркер используется для установки меркера защиты двигателя A0.0, в случае если не разомкнут ВЫКЛючатель или автомат защиты двигателя. Блокировка в этот момент должна отсутствовать. Меркер блокировки M10.0 устаналивается при одновременном нажатии кнопок ВКЛ и ВЫКЛ и сбрасывается только после того, как обе кнопки окажутся в исходном положении.

После того как установлен выход защиты двигателя A0.0, запускается первый таймер T37. По прошествии 2 сек. устанавливается выход A0.1 для первого переключателя ротора. Затем запускается второй таймер T38 и по прошествии 2 сек. устанавливается выход A0.2 для второго переключателя ротора. Этот шаг повторяется для таймеров T39 и T40, которые со своей стороны включают переключатели ротора 3 и 4 установкой соответствующих выходов A0.3 и A0.4, так что двигатель в конце концов работает с номинальным числом оборотов. Двигатель будет отключен, если на входах E0.1 или E0.2 исчезает напряжение, т.е. если разомкнута кнопка ВЫКЛ или автомат защиты двигателя.

Размер программы составляет 77 слов.

KOP (S7-MicroDOS)	AWL (IEC)

Основная программа

```
// TITEL =  ФАЗНЫЙ РОТОР

// E0.0                 Кнопка вкл.
// E0.1                 Кнопка выкл. Нормально замкнута
// E0.2                 Защита двигателя Нормально замкнута
// A0.0                 Защита двигателя
// A0.1                 Переключатель ротора 1
// A0.2                 Переключатель ротора 2
// A0.3                 Переключатель ротора 3
// A0.4                 Переключатель ротора 4

// T37                  Таймер 1я ступень
// T38                  Таймер 2я ступень
// T39                  Таймер 3я ступень
// T40                  Таймер 4я ступень

// БЛОКИРОВКА
```

```
    |  E0.1    E0.0                                          M10.0     K1
1   ├─┤ / ├───┤   ├──────────────────────────────────────────(   S   )
    |
```

```
LDN     E0.1        // Кнопка Выкл нажата
U       E0.0        // Кнопка Вкл нажата
S       M10.0 ,1    // Вспомогательный меркер для блокировки
```

```
    |  E0.1    E0.0                                          M10.0     K1
2   ├─┤   ├───┤ / ├──────────────────────────────────────────(   R   )
    |
```

```
LD      E0.1        // Кнопка Выкл не нажата
UN      E0.0        // Кнопка Вкл не нажата
R       M10.0 ,1    // Блокировка снята
```

```
// ЗАПУСК ДВИГАТЕЛЯ
```

```
    |  E0.0    A0.1    A0.2    A0.3    A0.4                    M2.0
3   ├─┤   ├───┤ / ├───┤ / ├───┤ / ├───┤ / ├─┬──────────────────(       )
    |                                       |
    |  A0.0                                 |
    ├─┤   ├───────────────────────────────────┘
```

```
LD      E0.0        // Кнопка Вкл нажата
UN      A0.1        // Нет переключателя ротора 1
UN      A0.2        // Нет переключателя ротора 2
UN      A0.3        // Нет переключателя ротора 3
UN      A0.4        // Нет переключателя ротора 4
LD      A0.0
OLD
=       M2.0        // Промежуточный меркер
```

```
4   |  M2.0    E0.1    E0.2   M10.0                          A0.0
    |-|   |----|   |----|   |----| / |-------------------------(       )
    |
```

LD	**M2.0**	**// Промежуточный меркер**
U	**E0.1**	**// Кнопка Выкл не нажата**
U	**E0.2**	**// Защита двигателя OK**
UN	**M10.0**	**// Блокировка**
=	**A0.0**	**// Двигатель в движении**

// ЗАПУСК ТАЙМЕРА T37

```
5   |  A0.0    A0.1    A0.2    A0.3    A0.4                 TON---T37
    |-|   |----| / |----| / |----| / |----| / |-------------------|IN     |
    |                                                          |       |
    |                                                      K20-|PT     |
    |                                                          |       |
    |                                                          |_______|
```

LD	**A0.0**	**// Двигатель в движении**
UN	**A0.1**	**// Нет переключателя ротора 1**
UN	**A0.2**	**// Нет переключателя ротора 2**
UN	**A0.3**	**// Нет переключателя ротора 3**
UN	**A0.4**	**// Нет переключателя ротора 4**
TON	**T37,20**	**// Запуск T37 с 2 сек**

// ВКЛЮЧЕНИЕ ПЕРЕКЛЮЧАТЕЛЯ РОТОРА 1

```
6   |  T37     A0.0    A0.2    A0.3    A0.4                   A0.1
    |-|   |---|   |----| / |----| / |----| / |-------------------(       )
    |       |
    |  A0.1 |
    |-|   |-|
    |
```

LD	**T37**	**// T37 завершил работу**
O	**A0.1**	**// Самоподхват переключателя ротора 1**
LD	**A0.0**	**// Двигатель в движении**
UN	**A0.2**	**// Нет переключателя ротора 2**
UN	**A0.3**	**// Нет переключателя ротора 3**
UN	**A0.4**	**// Нет переключателя ротора 4**
ULD		
=	**A0.1**	**// Переключатель ротора 1**

// ЗАПУСК ТАЙМЕРА T38

```
7   |  A0.0    A0.2    A0.3    A0.4    A0.1                 TON---T38
    |-|   |----| / |----| / |----| / |----|   |-------------------|IN     |
    |                                                          |       |
    |                                                      K20-|PT     |
    |                                                          |       |
    |                                                          |_______|
```

```
LD      A0.0        // Двигатель в движении
UN      A0.2        // Нет переключателя ротора 2
UN      A0.3        // Нет переключателя ротора 3
UN      A0.4        // Нет переключателя ротора 4
U       A0.1        // Переключатель ротора 1
TON     T38,20      // Запуск T 38 с 2 сек
```

// ВКЛЮЧЕНИЕ ПЕРЕКЛЮЧАТЕЛЯ РОТОРА 2

```
    |  T38     A0.0    A0.3    A0.4                          A0.2
8   |-|   |-+-|   |---| / |---| / |-----------------------(       )
    |       |
    |  A0.2 |
    |-|   |-|
    |
```

```
LD      T38         // T38 завершил работу
O       A0.2        // Самоподхват переключателя ротора 2
LD      A0.0        // Двигатель в движении
UN      A0.3        // Нет переключателя ротора 3
UN      A0.4        // Нет переключателя ротора 4
ULD
=       A0.2        // Переключатель ротора 2
```

// ЗАПУСК ТАЙМЕРА T39

```
    |  A0.0    A0.2    A0.3    A0.4                      TON---T39
9   |-|   |---|   |---| / |---| / |-----------------------|IN     |
    |                                                     |       |
    |                                                 K20-|PT     |
    |                                                     |       |
    |                                                     |_______|
```

```
LD      A0.0        // Двигатель в движении
U       A0.2        // Переключатель ротора 2
UN      A0.3        // Нет переключателя ротора 3
UN      A0.4        // Нет переключателя ротора 4
TON     T39,20      // Запуск T 39 с 2 сек
```

// ВКЛЮЧЕНИЕ ПЕРЕКЛЮЧАТЕЛЯ РОТОРА 3

```
    |  T39     A0.0    A0.4                                A0.3
10  |-|   |-+-|   |---| / |-----------------------------(       )
    |       |
    |  A0.3 |
    |-|   |-|
    |
```

```
LD      T39         // T39 завершил работу
O       A0.3        // Самоподхват переключателя ротора 3
LD      A0.0        // Двигатель в движении
UN      A0.4        // Нет переключателя ротора 4
ULD
=       A0.3        // Переключатель ротора 3
```

// ЗАПУСК ТАЙМЕРА T40

```
11  |  A0.0    A0.3    A0.4                                  TON——T40
    |—|   |——|   |——| / |————————————————————————————————————|IN     |
    |                                                       |       |
    |                                                   K20-|PT     |
    |                                                       |       |
    |                                                       |_______|
```

LD	**A0.0**	**// Двигатель в движении**
U	**A0.3**	**// Переключатель ротора 3**
UN	**A0.4**	**// Нет переключателя ротора 4**
TON	**T40,20**	**// Запуск T 40 с 2 сек**

// ВКЛЮЧЕНИЕ ПЕРЕКЛЮЧАТЕЛЯ РОТОРА 4

```
12  |  T40     A0.0                                          A0.4
    |—|   |——|   |————————————————————————————————————————(       )
    |       |
    |  A0.4 |
    |—|   |—|
    |
```

LD	**T40**	**// T40 завершил работу**
O	**A0.4**	**// Самоподхват переключателя ротора 4**
LD	**A0.0**	**// Двигатель в движении**
ULD		
=	**A0.4**	**// Переключатель ротора 4**

```
13  |————————————————————————————————————————————————————( MEND  )
    |
```

MEND	**// Конец**

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 →K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведённого здесь.

За правильную работу системы заказчик несёт ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

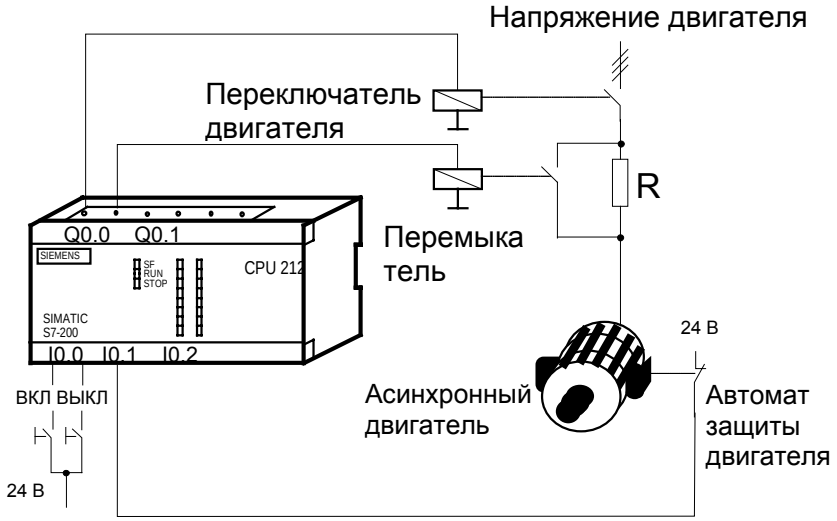
Группа	Пример к теме
4	КОПЛ-включение

Краткое описание

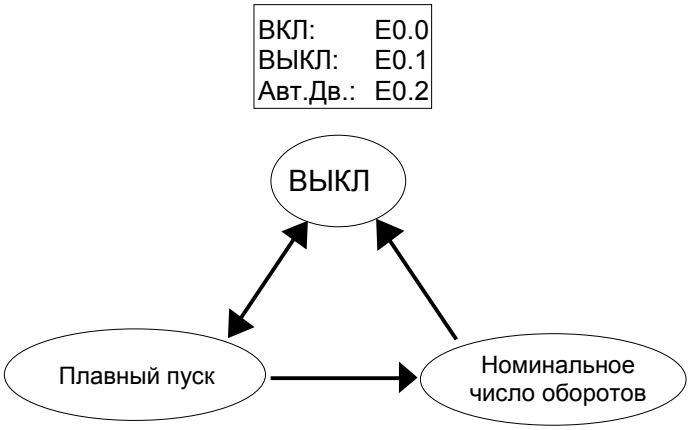
Данная программа позволяет автоматический запуск асинхронного двигателя с короткозамкнутым ротором с помощью КОПЛ-включения. При таком Короткозамкнуто-Плавном пуске двигатель запускается с минимальным числом оборотов и достигает номинального числа оборотов только через определенное время.

Нажатие кнопки ВКЛ на входе E0.0 включает плавный пуск двигателя, а кнопка ВЫКЛ на E0.1 отключает двигатель. На вход E0.2 подключен автомат защиты двигателя, который срабатывает при перегрузке двигателя и отключает его.

Схема включения



Структура программы



Описание программы, вкл. листинг

Меркер блокировки M1.0 устаналивается при одновременном нажатии кнопок ВКЛ (нормально разомкнута) и ВЫКЛ (нормально замкнута) на входах E0.0 и E0.1 и сбрасывается только после того, как обе кнопки окажутся в исходном положении.

Переключатель двигателя подключенный на выход A0.0 сработает после того как будет нажата кнопка ВКЛ и при этом не сработала ни блокировка (M1.0) ни автомат защиты двигателя (E0.2-нормально замкнуто) или кнопка ВЫКЛ (E0.1). Логическая операция ИЛИ с A0.0 реализует самоподхват переключателя. Двигатель запускается теперь с минимальным числом оборотов, так как сопротивление запуска еще не перемкнуто.

Если включается переключатель двигателя и выход A0.1 для перемыкателя еще не установлен, то запускается таймер T37. По прошествии установленного времени 5 сек устанавливается выход A0.1 перемыкателя, пока включен переключатель двигателя, т.е. пока установлен A0.0.

Размер программы составляет 28 слов.

KOP (S7-MicroDOS)	AWL (IEC)

Основная программа

```
// TITEL = КОПЛ-включение

// E0.0      Кнопка вкл.
// E0.1      Кнопка выкл. Нормально замкнута
// E0.2      Защита двигателя Нормально замкнута
// A0.0      Защита двигателя
// A0.1      Перемыкатель
// T37       Таймер для переключения (5 сек)
```

```
// ******************** БЛОКИРОВКА ********************************
```

```
    |  E0.1    E0.0                                                M1.0     K1
1   |--| / |----|   |-------------------------------------------------(   S   )
    |
```

```
LDN     E0.1        // Кнопка Выкл нажата
U       E0.0        // Кнопка Вкл нажата
S       M10.0 ,1    // Вспомогательный меркер для блокировки
```

```
    |  E0.1    E0.0                                                M1.0     K1
2   |--|   |----| / |-------------------------------------------------(   R   )
    |
```

```
LD      E0.1        // Кнопка Выкл не нажата
UN      E0.0        // Кнопка Вкл не нажата
R       M10.0 ,1    // Блокировка снята
```

```
// ***************ЗАПУСК ДВИГАТЕЛЯ*****************************
```

```
    |  E0.0    E0.1    E0.2    M1.0                                  A0.0
3   |--|   |----|   |----|   |----| / |-------------------------------(       )
    |       |
    |  A0.0 |
    |--|   |--|
    |
```

```
LD      E0.0        // Кнопка Вкл нажата
O       A0.0        // Самоподхват переключателя двигателя
LD      E0.1        // Кнопка Выкл не нажата
U       E0.2        // Защита двигателя OK
UN      M1.0        // Меркер блокировки
ULD
=       A0.0        // Защита двигателя
```

```
// *************ЗАПУСК ТАЙМЕРА T37***********************************

4   |  A0.0    A0.1                                                  TON——T37
    |—|    |——| / |————————————————————————————————————————————————|IN      |
    |                                                                |        |
    |                                                          K50-|PT      |
    |                                                                |        |
    |                                                                |_______|
```

LD	**A0.0**	**// Двигатель в движении**
UN	**A0.1**	**// Перемыкатель**
TON	**T37,50**	**// Запуск T37**

```
// ***********ВКЛЮЧЕНИЕ ПЕРЕМЫКАТЕЛЯ*****************************

5   |  T37     A0.0                                                  A0.1
    |—|    |——|    |————————————————————————————————————————————(        )
    |        |
    |  A0.1  |
    |—|    |—|
    |
```

LD	**T37**	**// T37 завершил работу**
O	**A0.1**	**// Самоподхват перемыкателя**
LD	**A0.0**	**// Двигатель в движении**
ULD		
=	**A0.1**	**// Перемыкатель**

```
6   |
    |————————————————————————————————————————————————————————————( MEND  )
    |
```

MEND	**// Конец**

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведённого здесь.

За правильную работу системы заказчик несёт ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

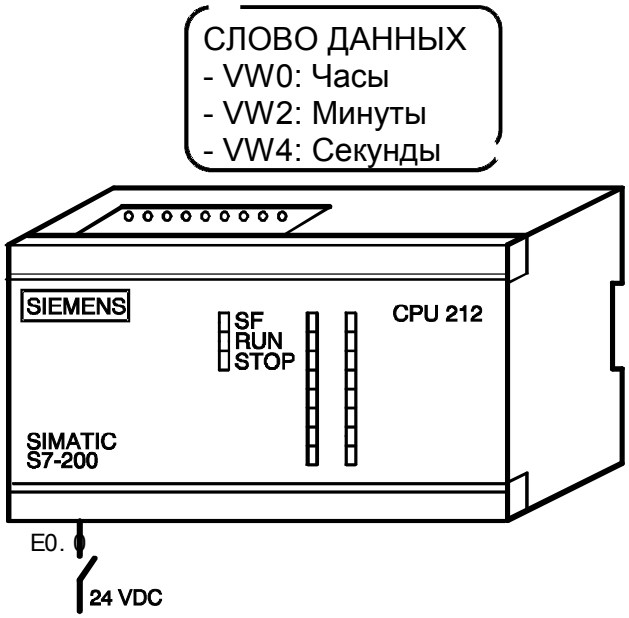
Группа	Пример к теме
1	Счетчик часов работы

Краткое описание

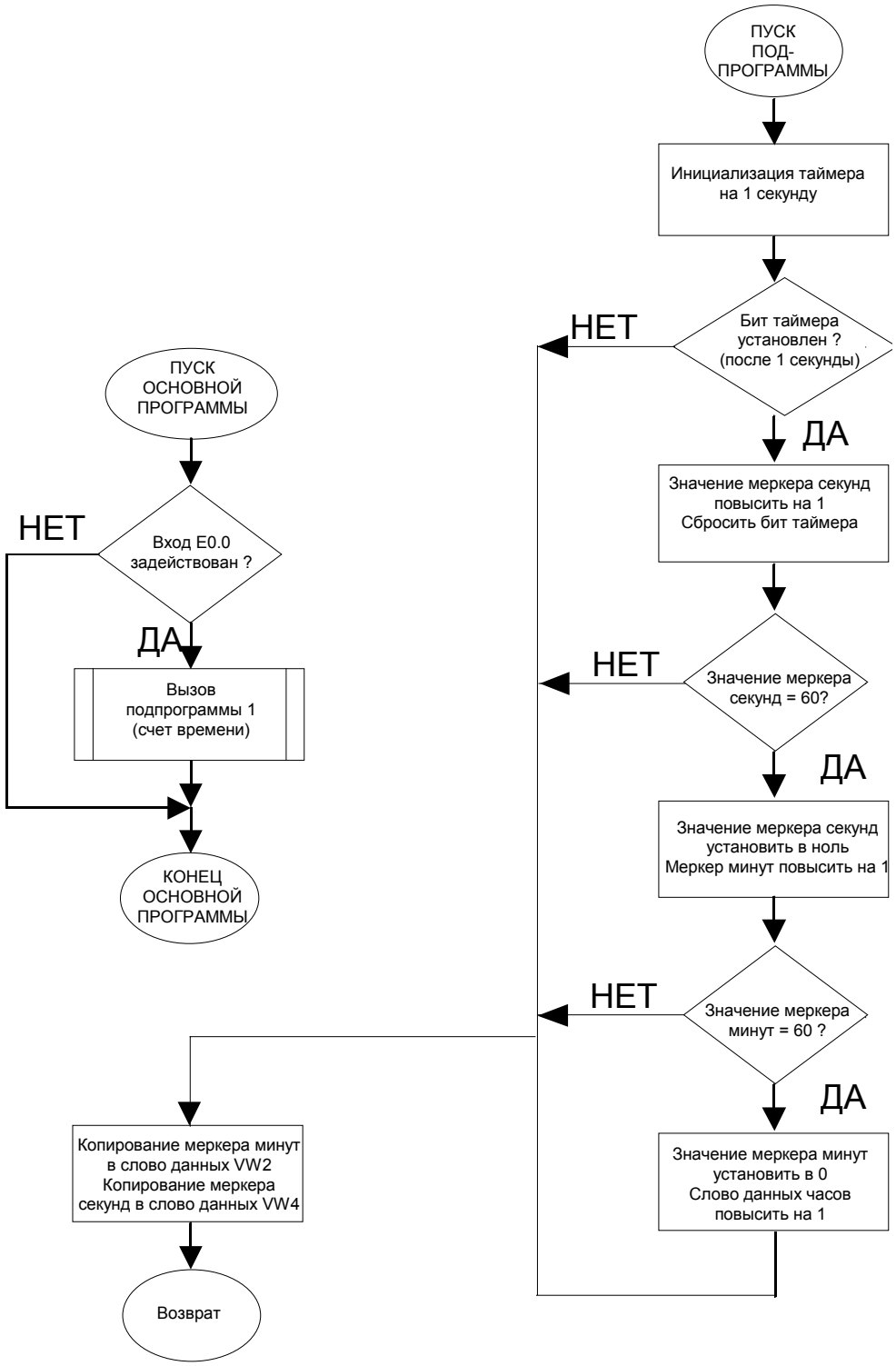
Данная программа-пример служит для того, чтобы определять длительность работы привода (тормоза, автоматы, и т.д.). Предпосылкой к этому является то, что при работающем приводе на вход E0.0 подается сигнал 24В, а при стоящем двигателе напряжение отсутствует.

Появление сигнала запускает счет времени, а при отсутствии сигнала счет времени прерывается пока сигнал не появится снова. Число сосчитанных часов заносится в слово данных VW0, минуты в слово данных VW2, а секунды ы слово данных VW4.

Схема включения



Структура программы



Описание программы вкл. листинг

Цикл программы начинается с опроса состояния входа E0.0. Если логическое состояние ‘1’, то вызывается Подпрограмма 1. В Подпрограмме 1 таймер T5 (задержка включения с запоминанием) устанавливается на одну секунду. По прошествии этой секунды устанавливается бит таймера ‘T5’, а значение слова меркеров MW4 для значения секунд увеличивается на 1. Для того чтобы при этом при следующем цикле таймер снова мог быть запущен, бит таймера ‘T5’ сбрасывается.

Если меркер секунд достигает значения 60, то меркер минут MW2 повышается на 1, а меркер секунд MW4 снова сбрасывается в ноль. (Знак ‘>‘ служит здесь для повышения надежности).

Если меркер минут достигает значения 60, то он сбрасывается в ноль, а слово данных VW0, работающее как память для прошедших часов, увеличивается на 1. Кроме того меркер минут MW2 копируется в слово данных VW2 для текущего значения минут, а меркер секунд MW4 в слово данных VW4 для текущего значения секунд. На этом работа подпрограммы завершается.

Последняя инструкция в конце основной программы служит только для вывода текущего значения секунд на светодиодный индикатор.

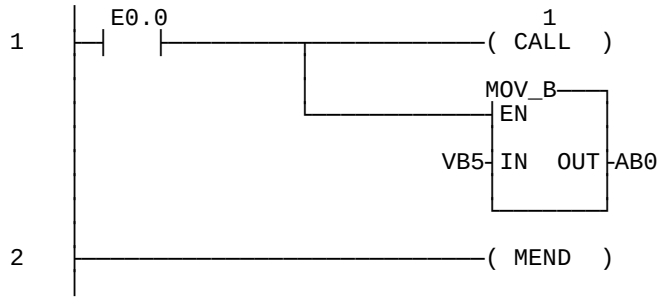
Размер программы составляет 41 слово.

Подробнее о таймере Вы найдёте в главе 4.1 “Операции времени” в руководстве по программированию SIMATIC S7-200.

KOP (S7-MicroDOS)	AWL (IEC)

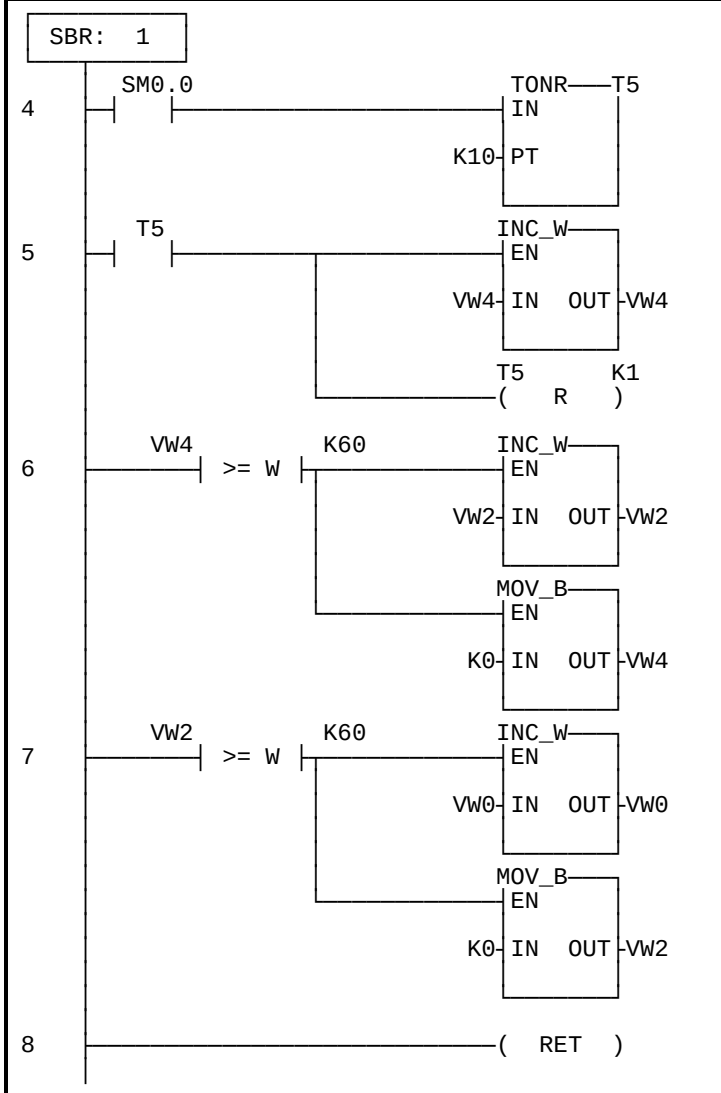
Основная программа

// TITEL= СЧЕТЧИК ЧАСОВ РАБОТЫ




```
LD      E0.0        // Время работы
                    // входа E0.0 будет
                    // измерено
CALL    1
MOVB    VB5, AB0// Выдача теста

MEND                // Конец программы
```

Подпрограмма




```
SBR 1                  // Начало подпрограммы
                       //  измерений
LD      SM0.0          // Установка начала стека
TONR    T5,10          // Задание времени
                       // тацмера: 100мсек *
                       // 10 = 1 сек

LD      T5             // По прошествии
                       // одной секунды
INCW    VW4            // меркер такта
                       // секунд
                       // инкрементируется

R       T5,1           // Сброс бита таймера

LDW>=   VW4,60         // Через  60 сек
INCW    VW2            // инкрементируется
                       // меркер минут

MOVW    0, VW4         // Меркер секунд
                       // сбрасывается

LDW>=   VW2,60         // Через  60 минут
INCW    VW0            // значение часов в VW0
                       // инкрементируется

MOVW    0, VW2         // Меркер минут
                       // сбрасывается

RET                    // Конец подпрограммы
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведённого здесь.

За правильную работу системы заказчик несёт ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

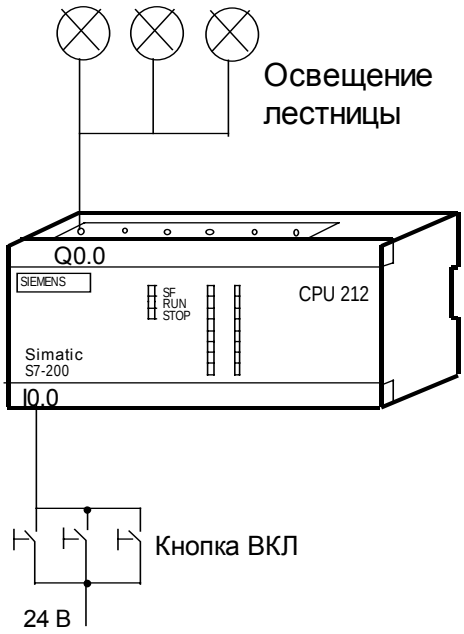
SIMATIC
S7-200 Примеры

Группа	Пример к теме
1	**Выключение освещения лестницы по времени**

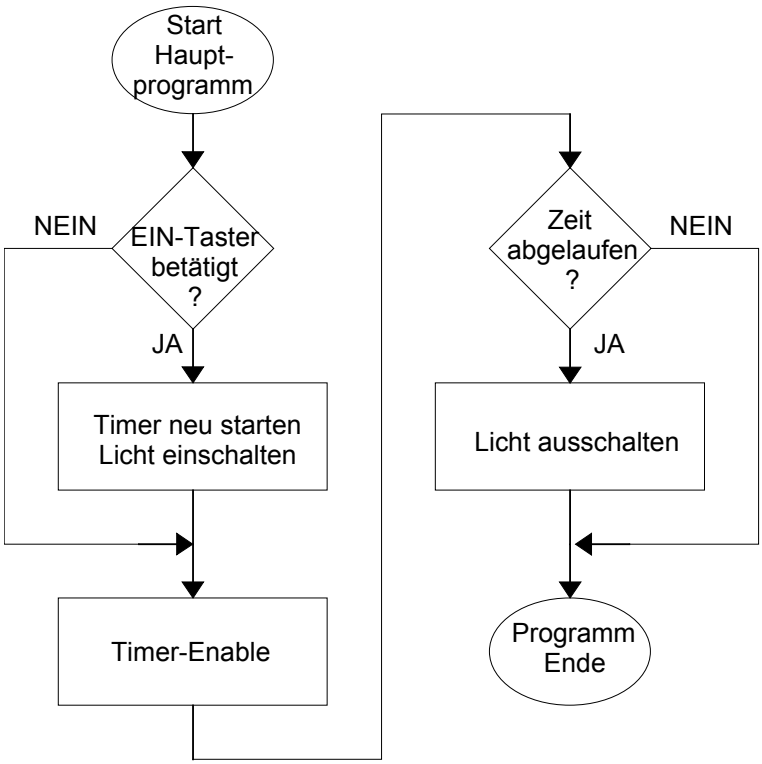
Краткое описание

Эта программа служит примером управления освещения лестницы. Кнопки ВКЛ с разных этажей подключены на вход E0.0 системы управления. После того как кнопка ВКЛ нажата, свет будет включен в течение 30 секунд, пока установлен предусмотренный для этого вход A0.0. Повторное нажатие в течение этого времени кнопки ВКЛ, начинает отсчет времени заново, что гарантирует, что включенное освещение через 30 секунд после последнего нажатия кнопки будет выключено.

Схема включения



Структура программы



Описание программы вкл. листинг

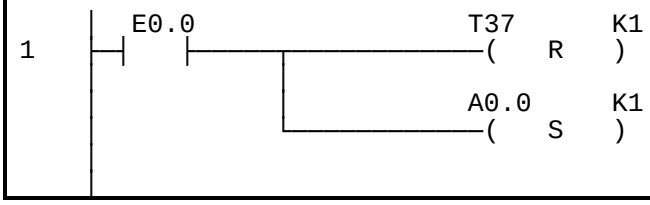
Если на входе E0.0 установлен сигнал ВКЛ (E0.0 = логическая ‘1’), то бит таймера T37 сбрасывается, причем таймер T37 начинает отсчет сначала. Одновременно утанавливается выход A0.0. Установка в начало стека необходимо для того, чтобы таймер был активен в каждом цикле. По прошествии 30 секунд таймер устанавливает таймерный бит T37. При этом выход A0.0 снова выключается.

Размер программы составляет 17 слов.

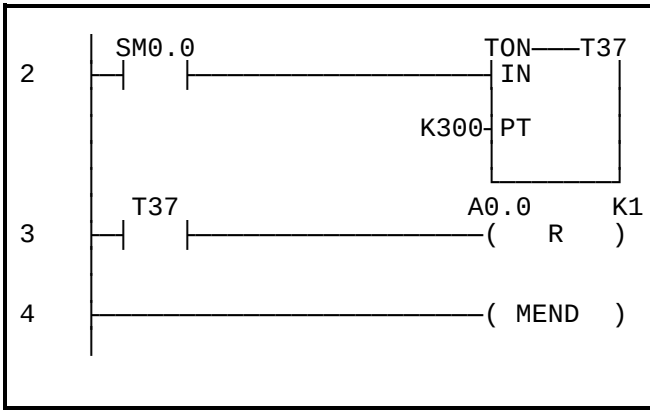
KOP (S7-MicroDOS)	AWL (TOOLITE2)

Основная программа

// TITEL = ВЫКЛЮЧЕНИЕ ПО ВРЕМЕНИ




```
LD    E0.0    // Если кнопка вкл
              // нажата
R     T37, 1  // то интервал времени
              // запускается заново
S     A0.0,1  // и свет включается
```




```
LD      SM0.0  // Устанавл. начало
               // стека для таймера
TON     T37, 300
               // Таймер считает 30 с

LD      T37    // Если время вышло
               //
R       A0.0, 1// свет выключается

MEND           // Конец
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

Copyright ©1996 by SIEMENS

SIMATIC
S7-200 Примеры

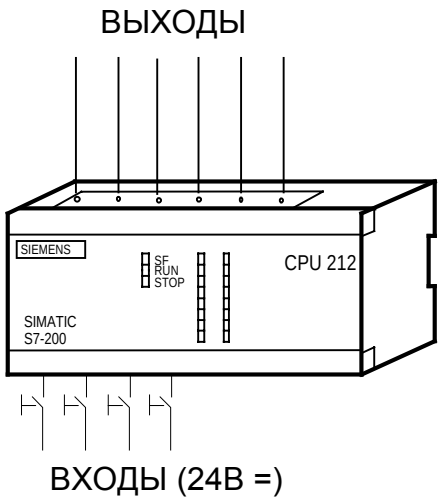
Группа	Пример к теме
1	**Шаговая цепочка**

Краткое описание

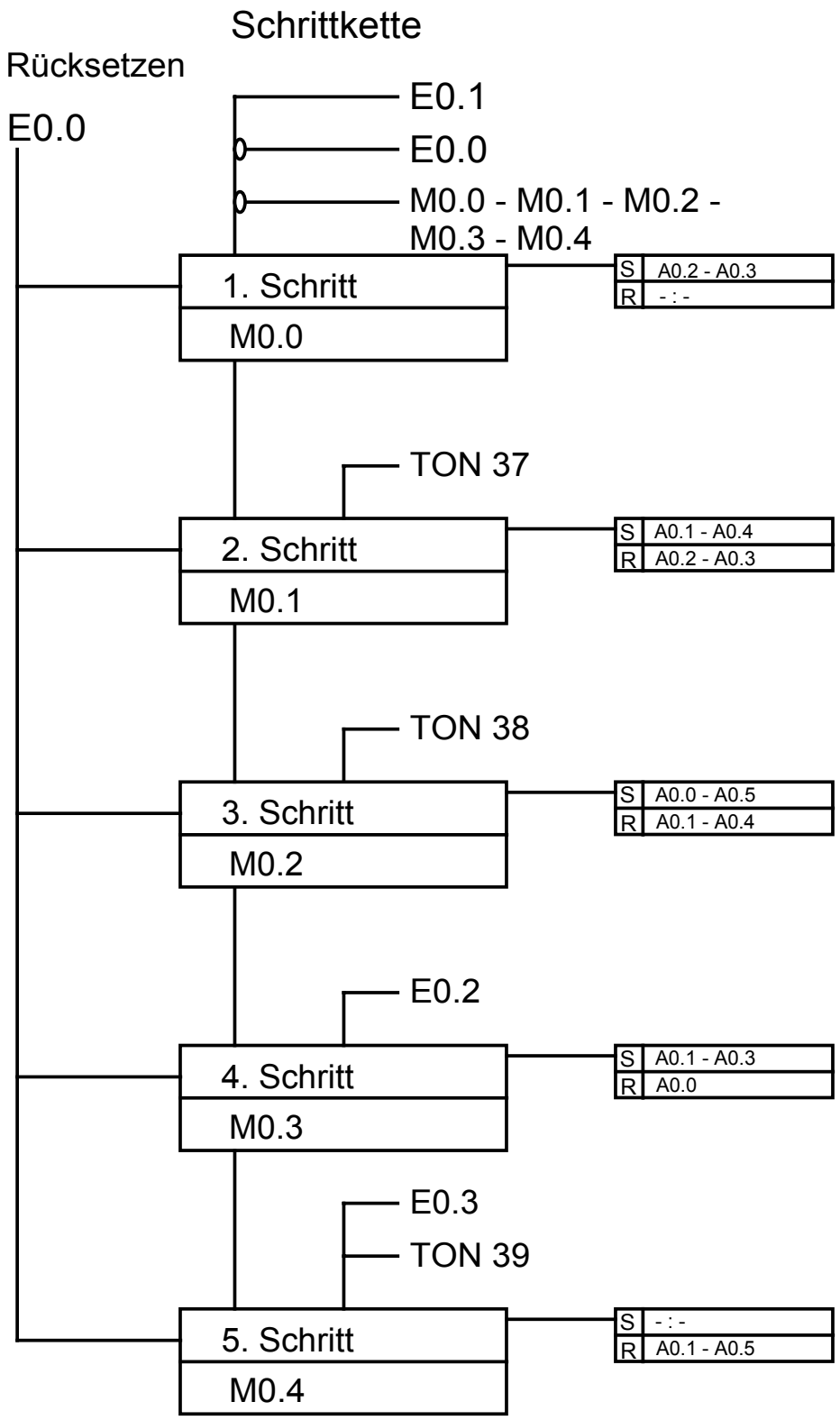
Данная программа реализует пример шаговой цепочки, причем в каждом шаге выполняется определенное действие. Шаги выполняются последовательно, причем шаг выполняется только, если выполнены все предусмотренные для этого условия. При этом ействительно следующее:

Предварительные условия		*Активные выходы*
1ый шаг	E0.1 установлен	A0.2 - A0.3
2ой шаг	Выдержка 5 сек (таймер T37)	A0.1 - A0.4
3ий шаг	Выдержка 5 сек (таймер T38)	A0.0 - A0.5
4ый шаг	E0.2 установлен	A0.1 - A0.3 - A0.5
5ый шаг	Выдержка 5 сек (таймер T39) & E0.3 установлен	A0.3
Сброс шаговой цепочки (E0.0 установлен)		нет

Схема включения



Структура программы



Описание программы вкл. листинг

Шаговая цепочка состоит из пяти шагов, выполняемых последовательно. Один шаг состоит из установки и сброса определенных выходов. Для того чтобы шаг мог быть сделан, должны быть выполнены определенные предварительные условия, как, например, нажатие выключателя или окончание времени ожидания. Кроме того нажатие выключателя E0.0 позволяет сбросить шаговую цепочку в любой момент времени x. Условия включения и активные выходы каждого шага представлены в разделе 'Структура программы'.

Размер программы составляет 116 слов.

Подробнее о таймере Вы найдёте в главе 4.1 “Операции времени” в руководстве по программированию SIMATIC S7-200.

KOP (S7-MicroDOS)	AWL (IEC)

Основная программа

// TITEL = ШАГОВАЯ ЦЕПОЧКА

// 1ый ШАГ

```
1   |  E0.1    E0.0    M0.0    M0.1    M0.2    M0.3    M0.4     M0.0     K1
    |-|    |----| / |----| / |----| / |----| / |----| / |----| / |-------(   S   )
    |
```

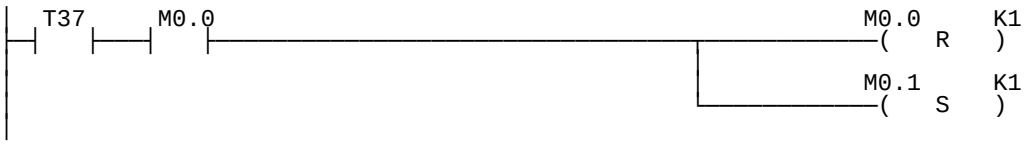
LD	**E0.1**	**// Условия пуска**
UN	**E0.0**	**// и нет сброса**
UN	**M0.0**	**// и не установлены следующие шаги**
UN	**M0.1**	
UN	**M0.2**	
UN	**M0.3**	
UN	**M0.4**	
S	**M0.0,1**	**// 1ый шаг установить**

```
2   |  M0.0                                                    A0.2     K2
    |-|    |-------------------------------------------------------(   S   )
    |                                                  |
    |                                                  |          TON----T37
    |                                                  |-----------|IN      |
    |                                                  |          |        |
    |                                                  |     K50-|PT      |
    |                                                  |          |        |
    |                                                  |          |________|
```

LD	**M0.0**	**// 1ый шаг**
S	**A0.2,2**	**// Установить выходы**
TON	**T37,50**	**// Интервал времени для 2го шага**

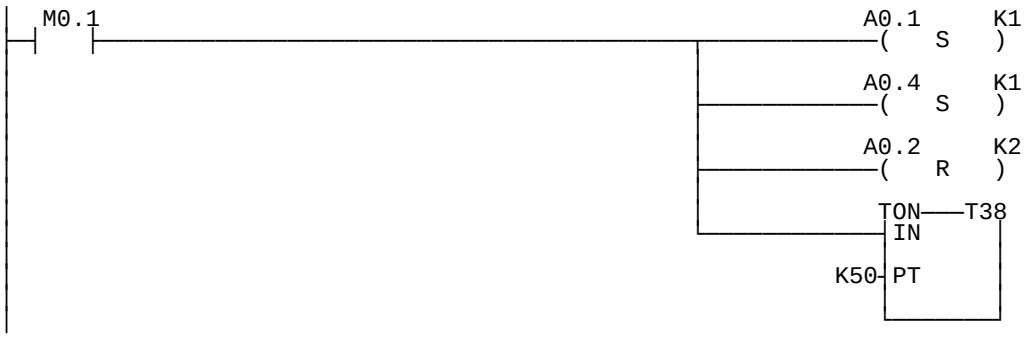
// 2ой ШАГ

3



LD	**T37**	**// По окончании первого интервала времени**
U	**M0.0**	**// и после выполнения первого шага**
R	**M0.0,1**	**// 1ый шаг сбросить,**
S	**M0.1,1**	**// а 2ой шаг установить**

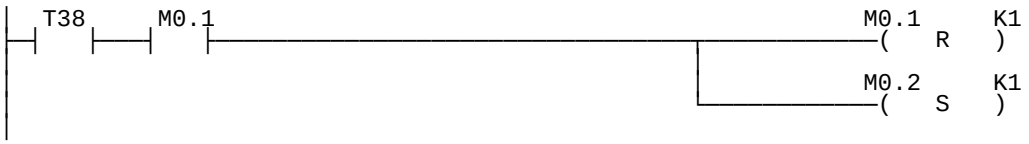
4



LD	**M0.1**	**// 2ой шаг**
S	**A0.1,1**	**// Установить выходы**
S	**A0.4,1**	
R	**A0.2,2**	**// Сбросить выходы**
TON	**T38,50**	**// Интервал времени для 3го шага**

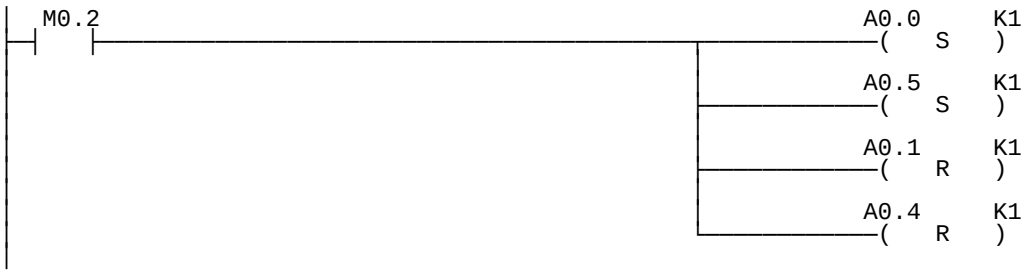
// 3ий ШАГ

5



LD	**T38**	**// По окончании второго интервала времени**
U	**M0.1**	**// и после выполнения второго шага**
R	**M0.1,1**	**// 2ой шаг сбросить,**
S	**M0.2,1**	**// а 3ий шаг установить**

6



LD	M0.2	// 3ий шаг
S	A0.0,1	// Установить выходы
S	A0.5,1	
R	A0.1,1	// Сбросить выходы
R	A0.4,1	

// 4ый ШАГ

```
7   |  E0.2    M0.2                                          M0.2     K1
    |-|   |----|   |-------------------------------------------(   R   )
    |                                                  |
    |                                                  |       M0.3     K1
    |                                                  |-------------(   S   )
    |
```

LD	E0.2	// После появления условий включения 4го шага
U	M0.2	// и после выполнения третьего шага
R	M0.2,1	// 3ий шаг сбросить,
S	M0.3,1	// а 4ый шаг установить

```
8   |  M0.3                                                 A0.1     K1
    |-|   |-------------------------------------------------(   S   )
    |                                                  |
    |                                                  |       A0.3     K1
    |                                                  |-------------(   S   )
    |                                                  |
    |                                                  |       A0.0     K1
    |                                                  |-------------(   R   )
    |                                                  |
    |                                                  |        TON----T39
    |                                                  |-------------| IN     |
    |                                                  |           |        |
    |                                                  |       K50-| PT     |
    |                                                  |           |        |
    |                                                  |           |_______|
```

LD	M0.3	// 4ый шаг
S	A0.1,1	// Установить выходы
S	A0.3,1	
R	A0.0,1	// Сбросить выходы
TON	T39,50	// Интервал времени для 5го шага

// 5ый ШАГ

```
9   |  E0.3    T39     M0.3                                  M0.3     K1
    |-|   |----|   |----|   |-------------------------------(   R   )
    |                                                  |
    |                                                  |       M0.4     K1
    |                                                  |-------------(   S   )
    |
```

LD	E0.3	// После появления условий включения 5го шага
U	T39	
U	M0.3	// и после выполнения четвертого шага
R	M0.3,1	// 4ый шаг сбросить,
S	M0.4,1	// а 5ый шаг установить

```
10  |  M0.4                                                    A0.1     K1
    |-|  |-------------------------------------------------------(   R   )
    |                                                 |
    |                                                 |          A0.5     K1
    |                                                 |----------(   R   )
    |
```

LD	**M0.4**	**// 5ый шаг**
R	**A0.1, 1**	**// Сброс выходов**
R	**A0.5, 1**	

// СБРОС ШАГОВОЙ ЦЕПОЧКИ

```
11  |  E0.0                                                    M0.0     K5
    |-|  |-------------------------------------------------------(   R   )
    |                                                 |
    |                                                 |          A0.0     K6
    |                                                 |----------(   R   )
    |
```

LD	**E0.0**	**// Если кнопка сброса была нажата,**
R	**M0.0, 5**	**// все 5 шаговых меркеров сбросить,**
R	**A0.0, 6**	**// все 6 выходов сбросить**

```
12  |-------------------------------------------------------------( MEND  )
    |
```

MEND **// Конец**

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 → K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF → KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведённого здесь.

За правильную работу системы заказчик несёт ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

Группа	Тема
3	**Подключение параллельного принтера к S7-200 в режиме свободнопрограммируемого интерфейса**

Краткое описание

В данном примере описывается подключение S7-200 CPU 212 к принтеру и передача информации на него используя режим свободнопрограммируемого интерфейса ПЛК.

Программа выполняет следующие функции: Когда вход I0.0 активен, программа выводит текст "SIMATIC S7-200.” Когда активны входы I0.1 - I0.7, выводится соответствующее сообщение "ВХОД 0.x АКТИВЕН !".

В программе подразумевается, что Вы подсоединили принтер к ПЛК через параллельный интерфейс с скоростью передачи 9600 Бод.

Рисунок 19.1

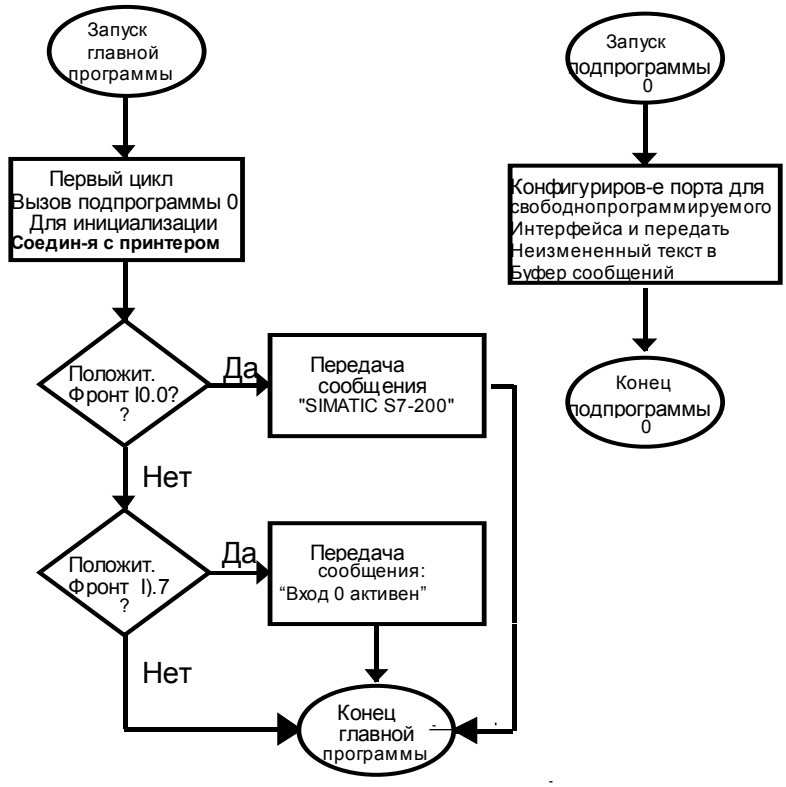
Аппаратные требования

Для нормальной работы данной программы Вам необходимо:

1 Simatic S7-200 CPU 212 или CPU 214
1 PC\PPI кабель
1 переходник 9 штырьковый "папа" на 25 штырьковый "папа"
1 последовательный в параллельный конвертор
1 нуль-модемный кабель с перекрестием линий 2 и 3 (если необходимо)
1 параллельный принтер

Существует одна особенность, которую надо учитывать:
Т.к. SIMATIC S7-200 и принтер оба ведут себя как пассивные устройства (slave) (DCE), то передача данных двух устройств ожидается в одном направлении, например обе линии приема данных и обе линии передачи данных соединены одна с друой (линии 2 и 3). Проблема может быть решена при помощи конвертора или используя соответствующий адаптер (нуль-модемный адаптер).

Структура программы



Описание программы

Если переключатель режима S7-200 установлен в RUN при запуске программы, главная программа переводит его в режим свободнопрограммируемого интерфейса. Затем, в зависимости от того, какой вход активен, программа передает соответствующее сообщение на принтер. (Главная программа так же определяет, какие байты будут вызываться).

Подпрограмма 0 содержит параметры настройки свободнопрограммируемого интерфейса и текст, который будет распечатываться, если активен соответствующий вход. Если активен вход I0.0, принтер печатает “SIMATIC S7-200”. Если один из входов I0.1 - I0.7 активен, принтер печатает соответствующее сообщение (“ВХОД 0.1 АКТИВЕН !” - “ ВХОД 0.7 АКТИВЕН !”).

Программа состоит из двух основных частей: Главного цикла (который инициализирует соединение и управляет запросами входов) и попрограммы 0 (которая подготовляет программу печати).

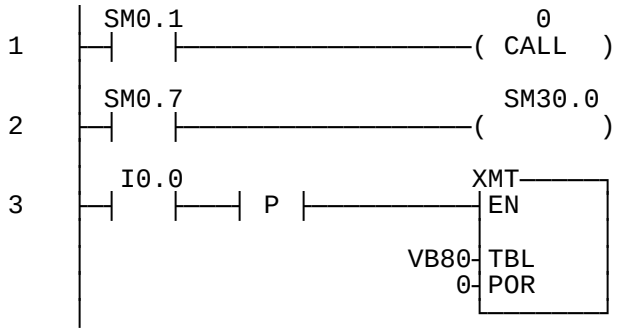
Размер программы 120 слов.

Более подробную информацию о свободнопрограммируемом интерфейсе Вы найдете в Главе 2.6 “Специальные Биты Памяти” *Руководства по Программированию Step 7-Micro*; дополнительную информацию о программах передаче - в Главе 6.4 “Команды передачи”.

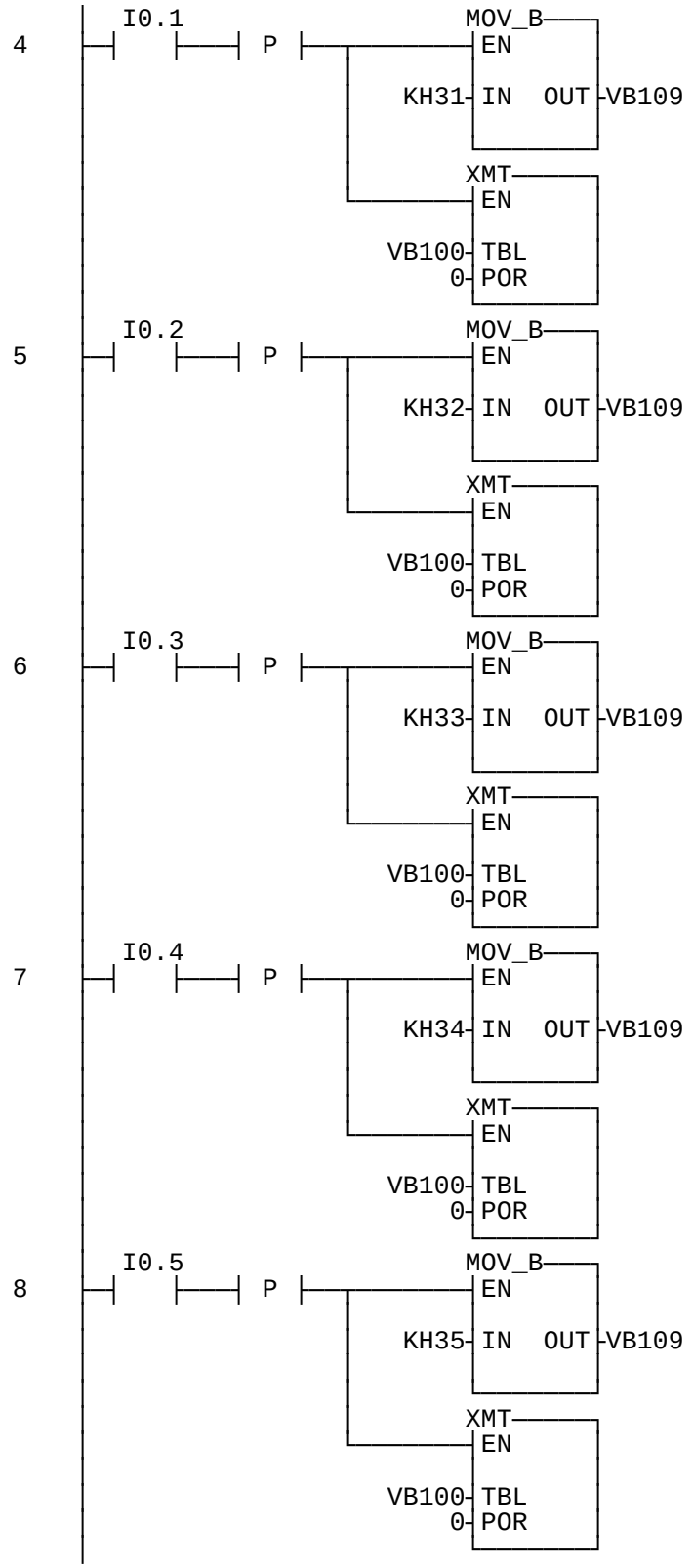
LAD (S7-MicroDOS)	STL (IEC)

Главная программа

// TITLE = ПРИНТЕР

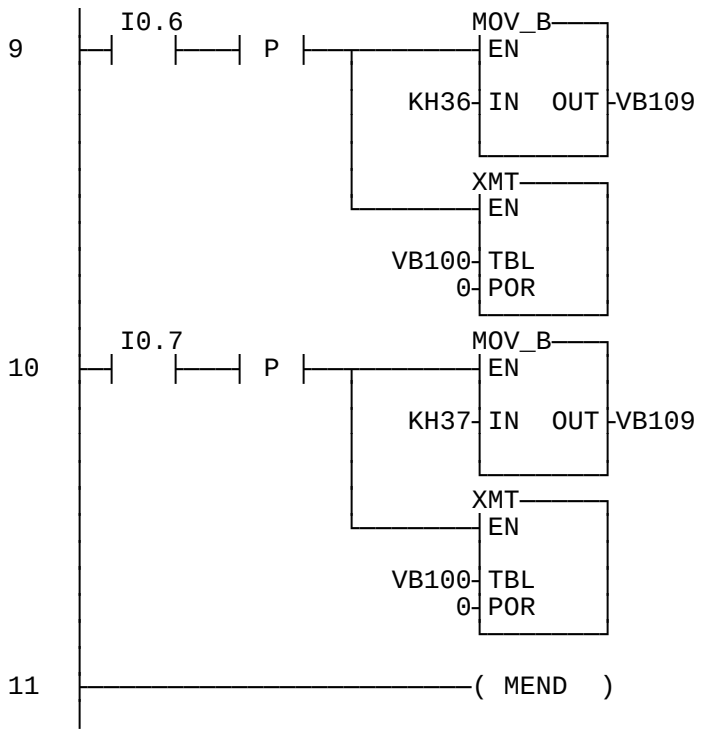



```
LD      SM0.1     // Бит первого цикла
CALL    0       // Вызов
                  // подпограммы 0
LD      SM0.7     // Если режим term,
                  // установить
                  // протокол PPI
=       SM30.0    // Если режим run,
                  // установить
                  // свободнопрограм-
                  // мируемый
        // протокол

LD      I0.0    // Запуск процесса
                // печати
EU              // Определение
            // положительного
                // фронта
XMT     VB80,0  // Передача таблицы
                // начиная с VB80
```




```
LD      I0.1        // Запуск печати для
        // входа I0.1
EU
MOVB    16#31,VB109
                    // Поместить 1 в
                    // VB109
XMT     VB100,0

LD      I0.2        // Запуск печати для
        // входа I0.2
EU
MOVB    16#32,VB109
                    // Поместить 2 в
                    // VB109
XMT     VB100,0

LD      I0.3        // Запуск печати для
        // входа I0.3
EU
MOVB    16#33,VB109
                    // Поместить 3 в
                    // VB109
XMT     VB100,0

LD      I0.4        // Запуск печати для
        // входа I0.4
EU
MOVB    16#34,VB109
                    // Поместить 4 в
                    // VB109
XMT     VB100,0

LD      I0.5        // Запуск печати для
        // входа I0.5
EU
MOVB    16#35,VB109
                    // Поместить 5 в
                    // VB109
XMT     VB100,0
```

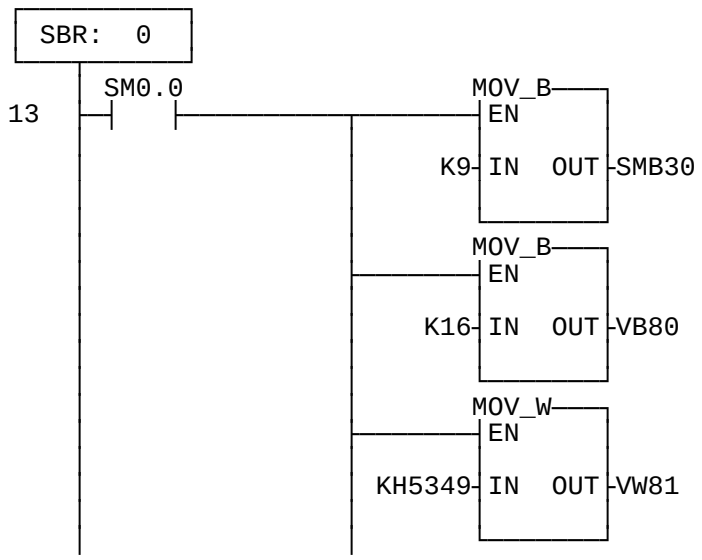
Copyright ©1995 by SIEMENS




```
LD      I0.6        // Запуск печати для
        // входа I0.6
EU
MOVB    16#36,VB109
                    // Поместить 6 в
                    // VB109

XMT     VB100,0

LD      I0.7        // Запуск печати для
        // входа I0.7
EU
MOVB    16#37,VB109
                    // Поместить 7 в
                    // VB109

XMT     VB100,0

MEND    // Конец главной
                    // программы
```

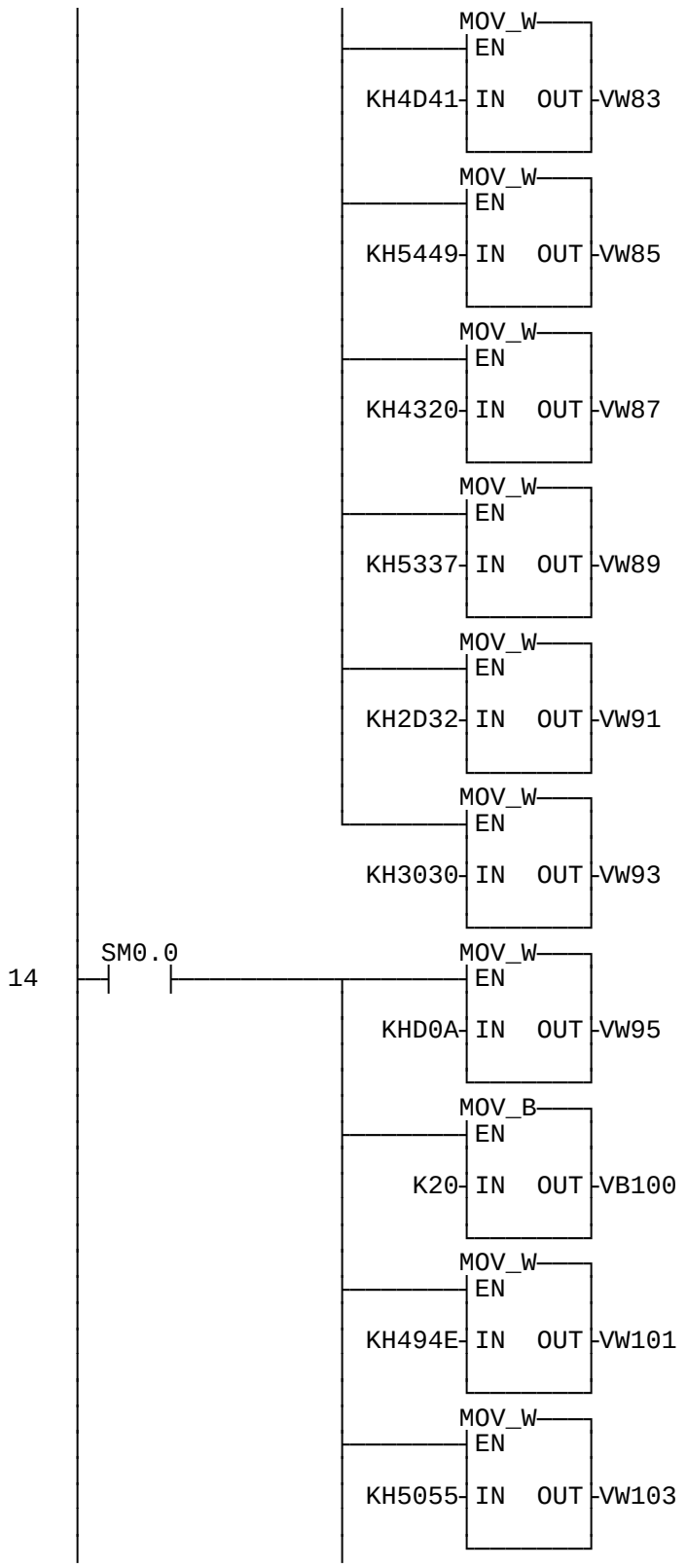
Подпрограммы

```
// ***************************************************************
// Подпрограмма 0
```




```
SBR     0           // Настройка печати

LD      SM0.0       // Бит всегда "1"
MOVB    9,SMB30
                    // 9600 Бод, без
                    // четности, 8 бит/знак

MOVB    16,VB80
                    // Длина сообщения
                    // 16 ASCII-знаков

MOVW    16#5349,VW81
```

Copyright ©1995 by SIEMENS



MOVW **16#4D41,VW83**

MOVW **16#5449,VW85**
// Печатаемый текст:

MOVW **16#4320,VW87**
// SIMATIC S7-200

MOVW **16#5337,VW89**

MOVW **16#2D32,VW91**

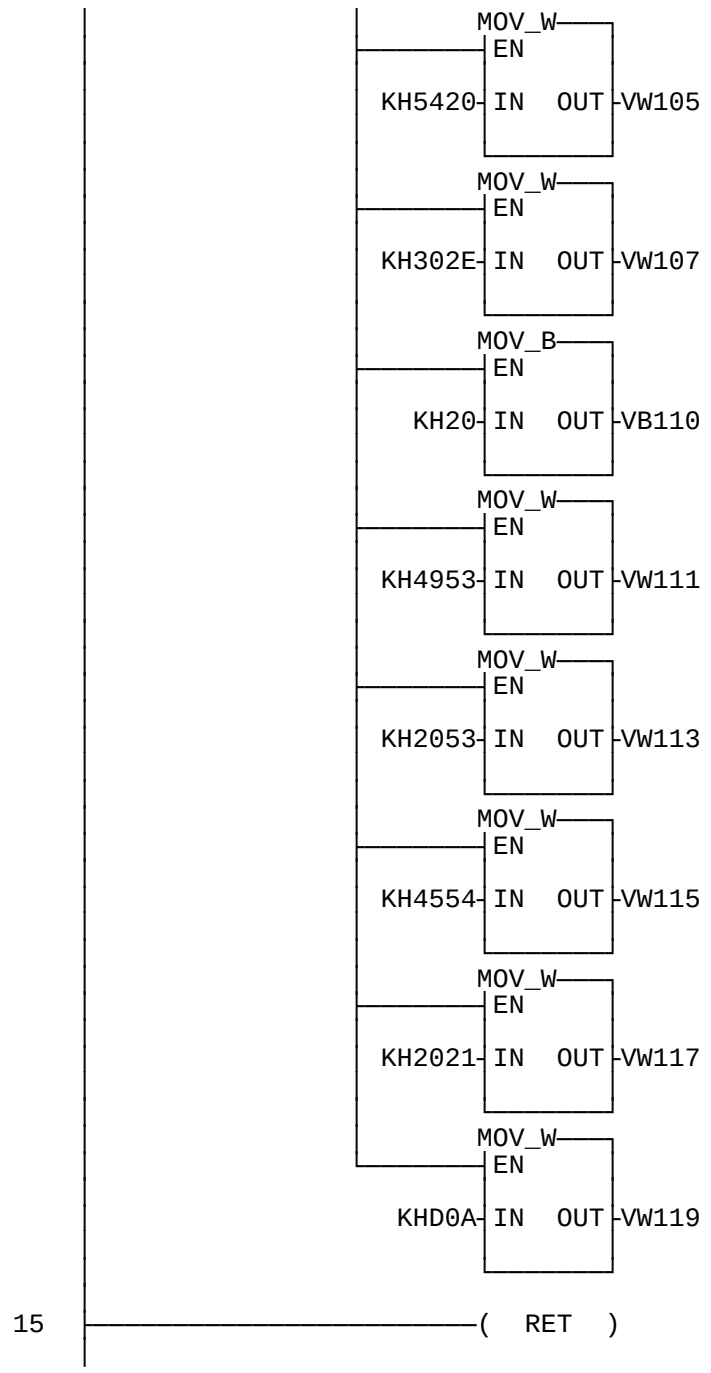
MOVW **16#3030,VW93**

LD **SM0.0 // Бит всегда "1"**
MOVW **16#0D0A,VW95**

MOVB **20,VB100**
// Длина сообщения
// 20 ASCII-знаков

MOVW **16#494E,VW101**

MOVW **16#5055,VW103**



MOVW **16#5420,VW105**
// Печатаемый текст:

MOVW **16#302E,VW107**
// INPUT 0.x IS SET !

MOVB **16#20,VB110**
// VB109 загружается
// в главной
// программе

MOVW **16#4953,VW111**

MOVW **16#2053,VW113**

MOVW **16#4554,VW115**

MOVW **16#2021,VW117**

MOVW **16#0D0A,VW119**

RET **// Конец**
// подпрограммы

Copyright ©1995 by SIEMENS

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" , не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все права защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC
S7-200 Примеры

Группа	Тема
3	**Прием данных от считывателя штрих-кода через свободнопрограммируемый интерфейс**

Краткое описание

Этот пример показывает использование S7-200 CPU 212 или CPU 214 со считывателем штрих - кода.

После считывания штрих-кода, декодер интерпретирует информацию и передает данные в **режиме свободнопрограммтруемого интерфейса** в SIMATIC. Два буфера используются для хранения кодов в памяти CPU 212 или CPU 214. С каждым новым получаемым кодом буферы переключаются с одного на другой.

Обычно данные используются в процессе. Но в этом примере информация только запоминается в буфере приема и м.б. просмотрена используя пакет программирования S7-200.

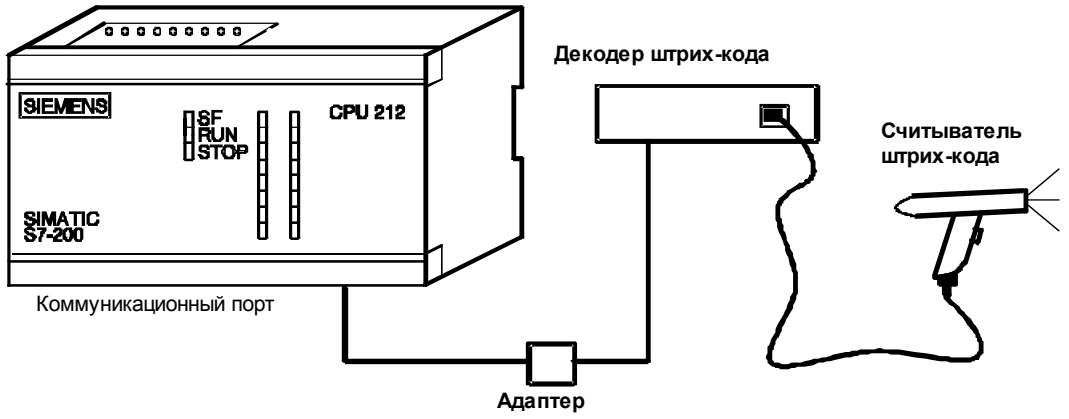


Рисунок 20.1

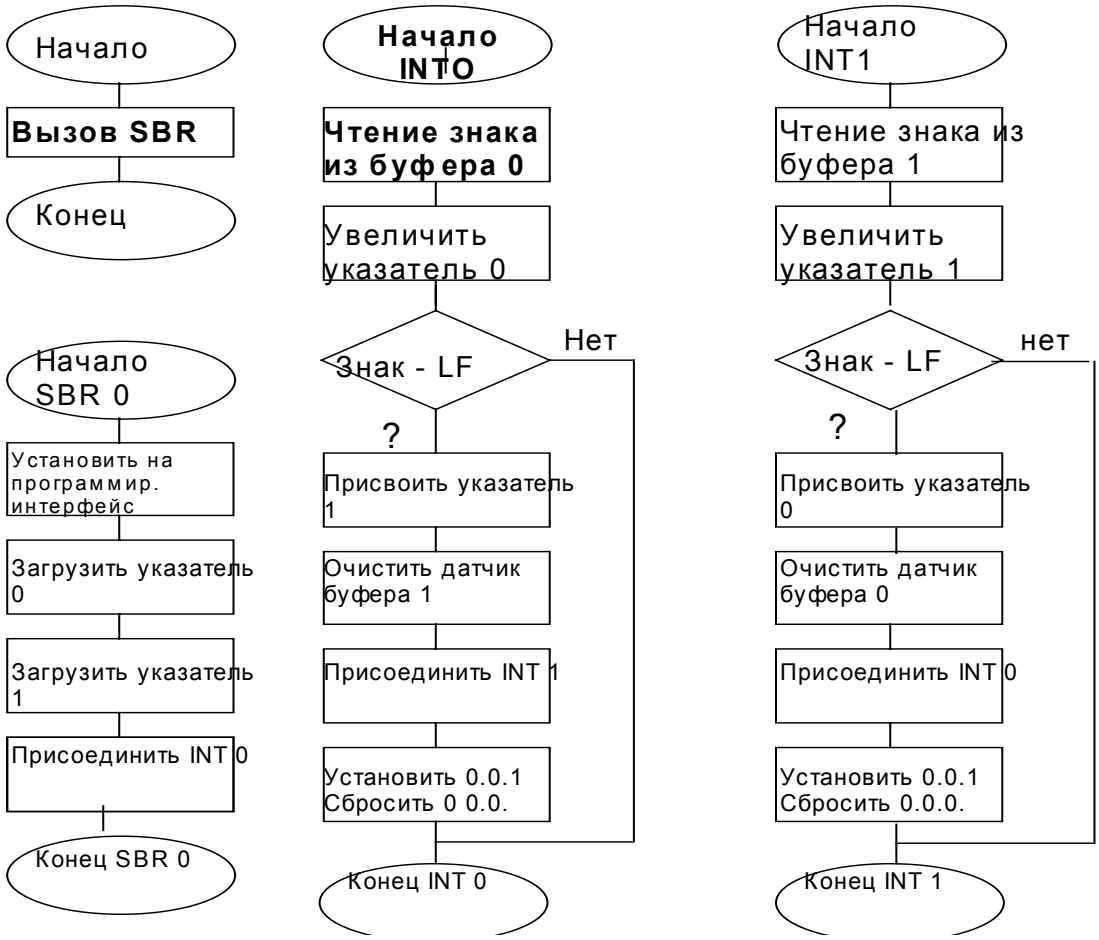
Аппаратные требования

Для правильной работы этого примера требуются:

- 1 SIMATIC S7-200 CPU 212 или CPU 214
- 1 кабель PC\PPI
- 1 соответствующий адаптер (в зависимости от интерфейса декодера штрих-кода, например 9 штырьковый "мама" на 25 штырьковый "папа", нуль-модемный с перекрестием линий 2 и 3)
- 1 считыватель штрих-кода
- 1 декодер штрих-кода (иногда считыватель и декодер - одно устройство)

Существует одна особенность, которую надо учитывать:
Т.к. SIMATIC S7-200 и считыватель штрих-кода оба ведут себя как пассивные устройства (slave) (DCE), то передача данных двух устройств ожидается в одном направлении, например обе линии приема данных и обе линии передачи данных соединены одна с друой (линии 2 и 3). Проблема может быть решена при помощи конвертора или используя соответствующий адаптер (нуль-модемный адаптер).

Структура программы



Описание программы

Эта программа сохраняет информацию, получаемую от считывателя штрих-кода в двух буферах.

Информация, передаваемая декодером штрих-кода, должна быть представлена в ASCII кодах. Принимаемые штрих-коды запоминаются в памяти SIMATIC. Данные могут быть использованы в программе. Но в данном примере информация только запоминается в буфере приема и и м.б. просмотрена используя пакет программирования SIMATIC S7-200.

Структура программы:

MAIN	Инициализация программы.
SBR 0	Настройка приема штрих-содов
INT 0	Буфер приема 0
INT 1	Буфер приема 1

Размер программы составляет 73 слова.

Более подробную информацию о свободнопрограммируемом интерфейсе Вы найдете в Главе 2.6 “Специальные Биты Памяти” *Руководства по Программированию Step 7-Micro*; дополнительную информацию о программах прерываний - в Главе 6.2 “Команды прерываний”.

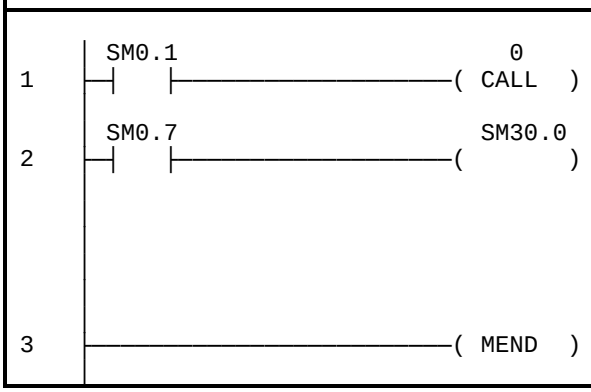
LAD (S7-MicroDOS)	STL (IEC)

Главная программа

// TITLE=Штрих-код

// Главная программа

// Основная задача главной программы - инициализация типа протоколоа.
// Если переключатель в позиции RUN, то устанавливается специальный меркер SM 0.7, и
// разрешается режим свободнопрограммируемого интерфейса. Текущий режим протокола
// свободнопрограммируемого интерфейса определяется в специальном байте памяти
// SM30 (см. Руководство по Программированию Step 7-Micro, Глава 2.6). В позиции TERM
// значение SM 0.7 равно 0 и как протокол передачи устанавливается протокол интерфейс
// точка к точке (PPI) и считыватель штрих-кода не может обмениваться с PLC.

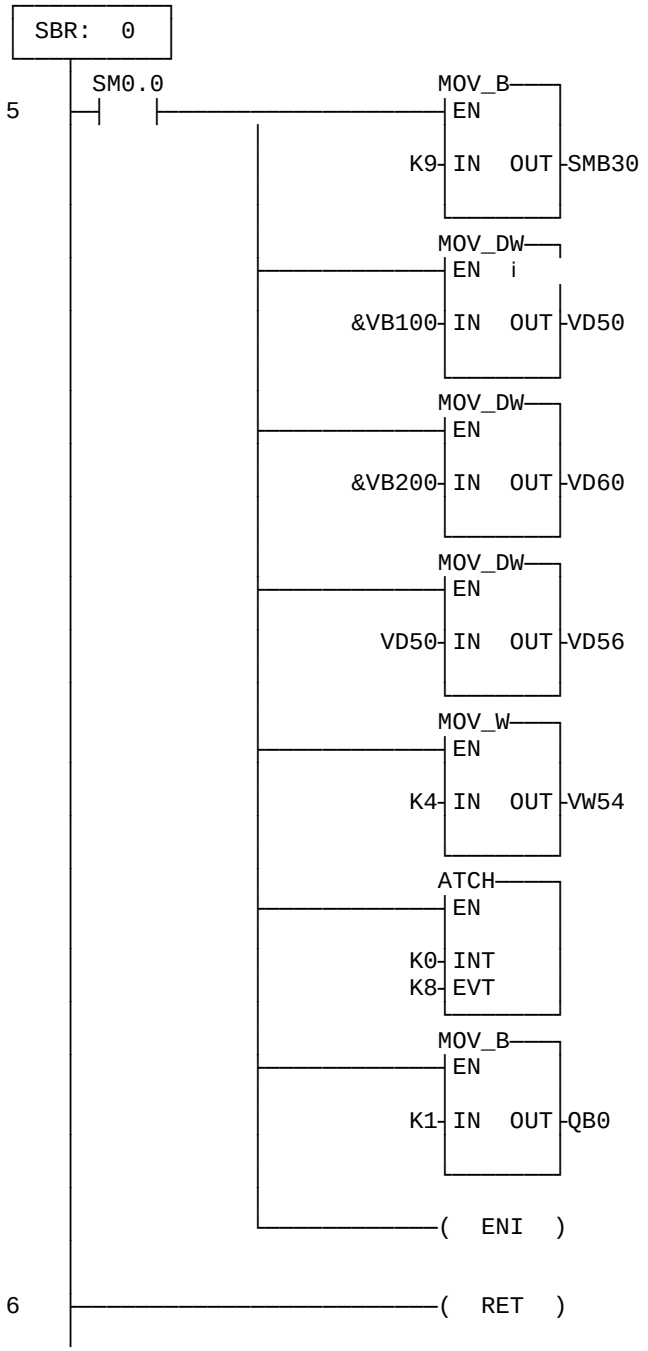



```
LD      SM0.1     // Меркер первого цикла
CALL  0 // Вызов
                  // подпрограммы 0

LD      SM0.7     // Если режим term,
                  // установить
                  // протокол PPI
=     SM30.0      // Если режим run,
                  // установить свободно-
                  // программируемый
                  // протокол
MEND
```

Подпрограммы

// Подпрограмма 0

// Если переключатель находится в позиции RUN, устанавливается режим
// свободнопрограммируемого интерфейса и SIMATIC получает данные от декодера
// штрих-кода. Выбирается режим свободнопрограммируемого протокола и определяются
// два указателя. Начальный адрес буфера 0 (VB100) загружается в указатель VD50, а
// начальный адрес буфера 1 (VB200) загружается в указатель VD60.
// Слова VW54 и VW64 исппользуются как счетчики знаков. Программа обработки
// прерываний 0 назначается и прерывания разрешаются.
// Когда считыватель штрих-кода опрашивается, данные загружаются или в буфер 0 или 1.

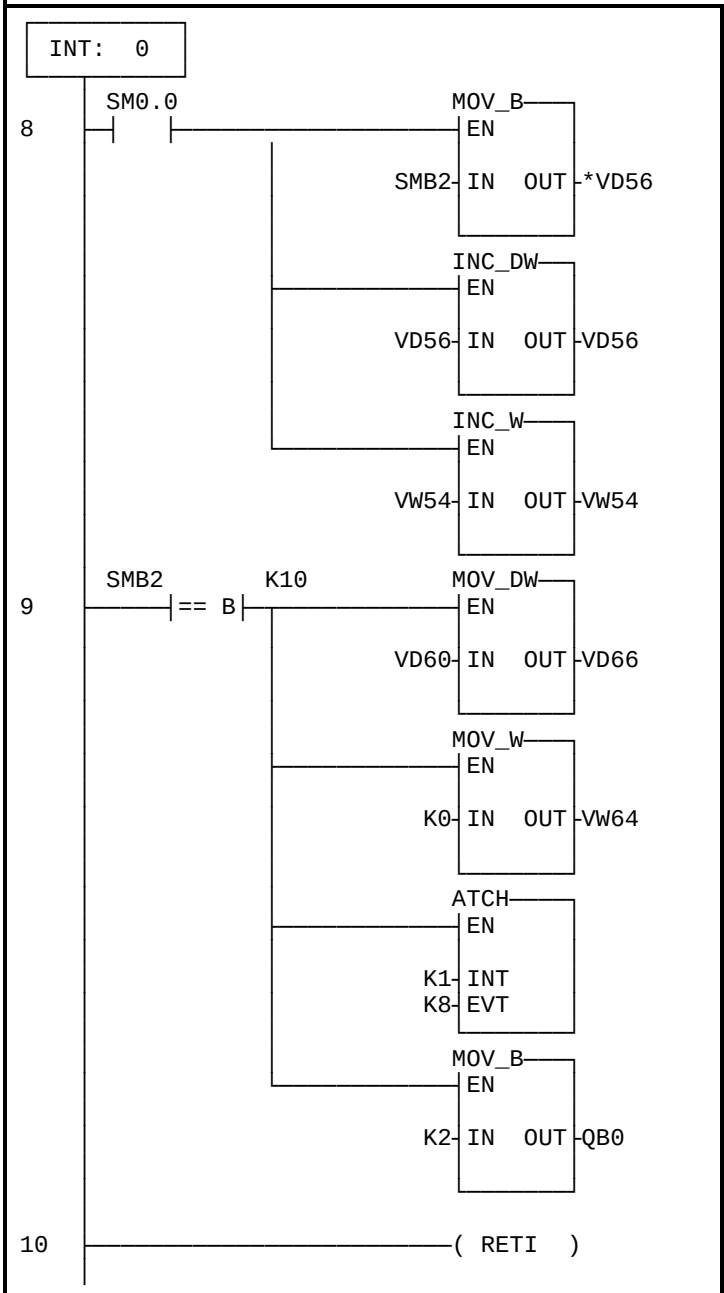



```
SBR     0            // Настройка приема
                     // штрих-кодов
LD      SM0.0
MOVB    9,SMB30      // 9600 Бод, без
                     // четности, 8 бит/знак

MOVD    &VB100,VD50
                     // Загрузить указатель
                     // на буфер 0

MOVD    &VB200,VD60
                     // Загрузить указатель
                     // на буфер 1

MOVD    VD50,VD56
                     // Загрузить V56 с
                     // указателем на
                     // буфер 0

MOVW    4,VW54       // Очистить счетчик
                     // знаков для буфера 0

ATCH    0,8          // Прием буфера 0 -
                     // программа обработки
                     // прерываний 0

MOVB    1,QB0        // Перевести Q0.0 в
                     // Вкл., а Q0.1 в Выкл.

ENI                  // Разрешить
                     // прерывания

RET                  // Прервать
                     // подпрограмму
```

Программы обработки прерываний

// Прерывание 0

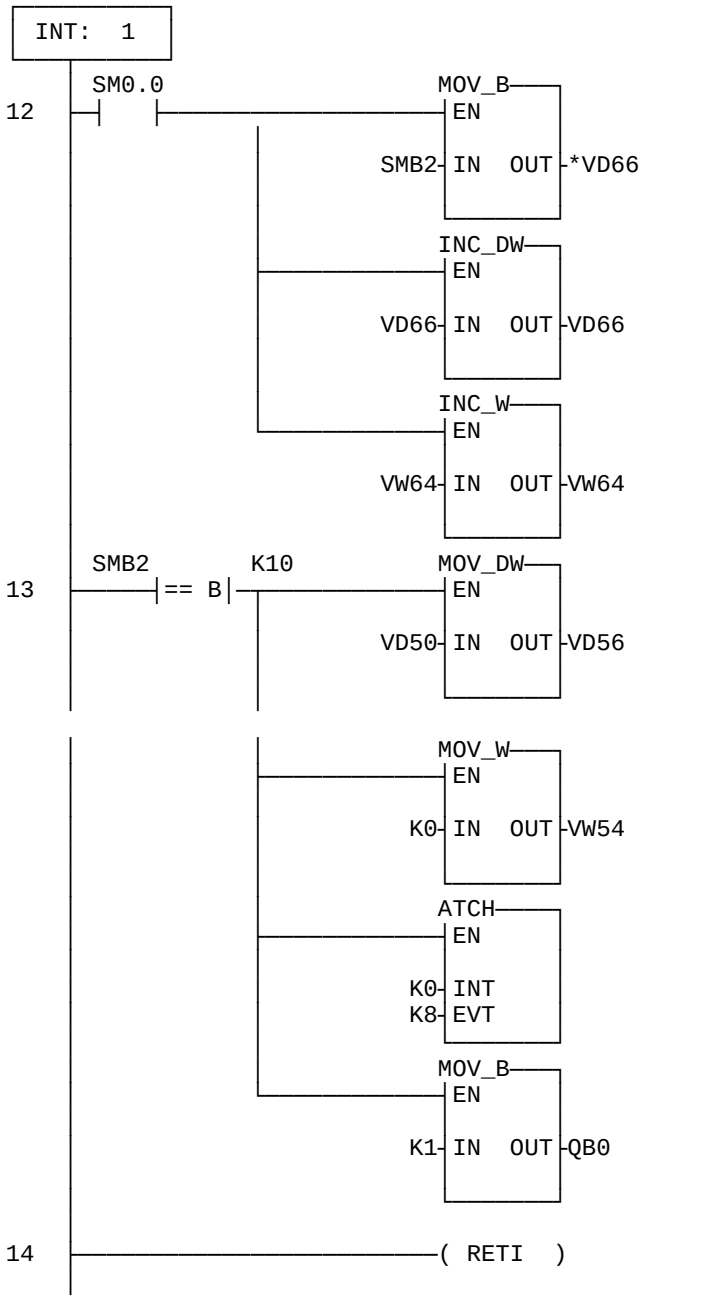
// Если разрешен буфер 0, то прерывание 0 прерывает программу и выполняется
// программа, описанная ниже.
// Указатель (содержимое VD56) загружается в VD100, указатель увеличивается на 1 для
// указания на следующий буфер и увеличивается на 1 счетчик знаков. Если знак -
// перевод строки буфера, то 1 загружается. Разрешается прерывание приема 1.
// Выход Q0.1 устанавливается в 1, выход Q0.0 - выключается.




```
INT     0              // Буфер приема 0

LD      SM0.0
MOVB    SMB2,*VD56
                       // Поместить знак в
                       // буфер 0

INCD    VD56 // Увеличить на 1
                       // указатель на
                       // следующий буфер

INCW    VW54 // Увеличить на 1
                       // счетчик знаков

LDB=    SMB2,10
                       // Если знак - LF,
MOVD    VD60,VD66
                       // Загрузить V66 с
                       // указателем на буфер1

MOVW    0,VW64
                       // Очистить счетчик для
                       // буфера 1

ATCH    1,8    // Прием в буфер 1-
                       // программа обработки
                       // прерываний 1

MOVB    2,QB0          // Перевести Q0.0 в
                       // Выкл.,а Q0.1 в Вкл.

RETI
```

// Прерывание 1

// Если разрешен буфер 1, то прерывание 1 прерывает программу и выполняется
// программа, описанная ниже.
// Указатель (содержимое VD66) загружается в VD200, указатель увеличивается на 1 для
// указания на следующий буфер и увеличивается на 1 счетчик знаков. Если знак -
// перевод строки буфера, то 0 загружается. Разрешается прерывание приема 0.
// Выход Q0.0 устанавливается в 1, выход Q0.1 - выключается.




```
INT    1                  // Буфер приема 1

LD     SM0.0
MOVB   SMB2,*VD66
                          // Поместить знак в
                          // буфер 1

INCD   VD66 // Увеличить на 1
                          // указатель на
                          // следующий буфер

INCW   VW64 // Увеличить на 1
                          // счетчик знаков

LDB=   SMB2,10
                          // Если знак - LF,
MOVD   VD50,VD56
                          // Загрузить V56 с
                          // указателем на
                          // буфер 0

MOVW   0,VW54
                          // Очистить счетчик для
                          // буфера 0

ATCH   0,8   // Прием в буфер 0 -
                          // программа обработки
                          // прерываний 0

MOVB   1,QB0              // Перевести Q0.0 в
                          // Вкл.,а Q0.1 в Выкл.

RETI
```

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" , не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все прав защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

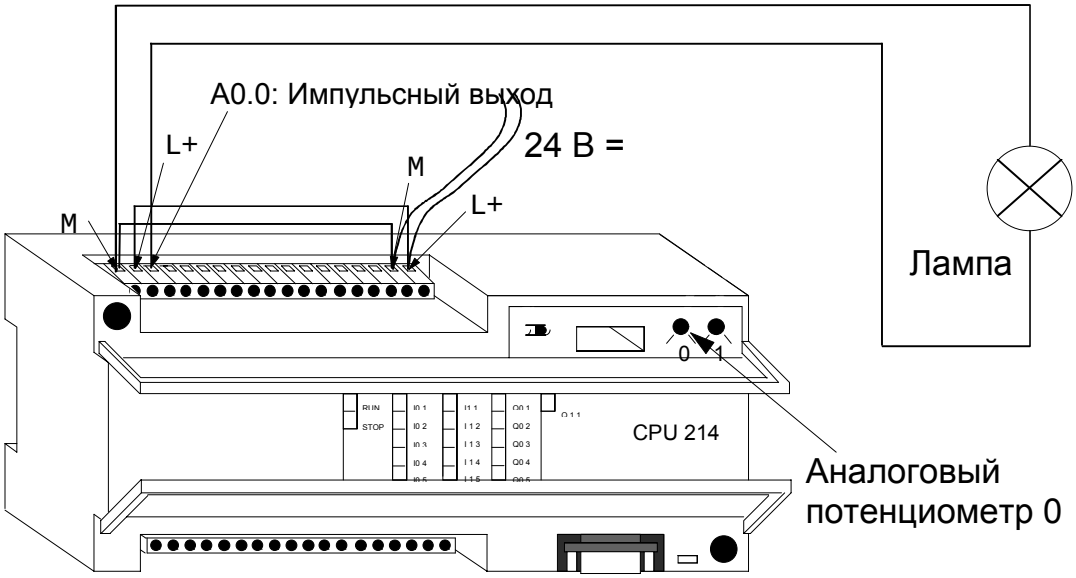
SIMATIC
S7-200 Примеры

Группа	Пример к теме
1	**Регулирование лампы от S7-200 CPU 214 DC/DC/DC**

Краткое описание

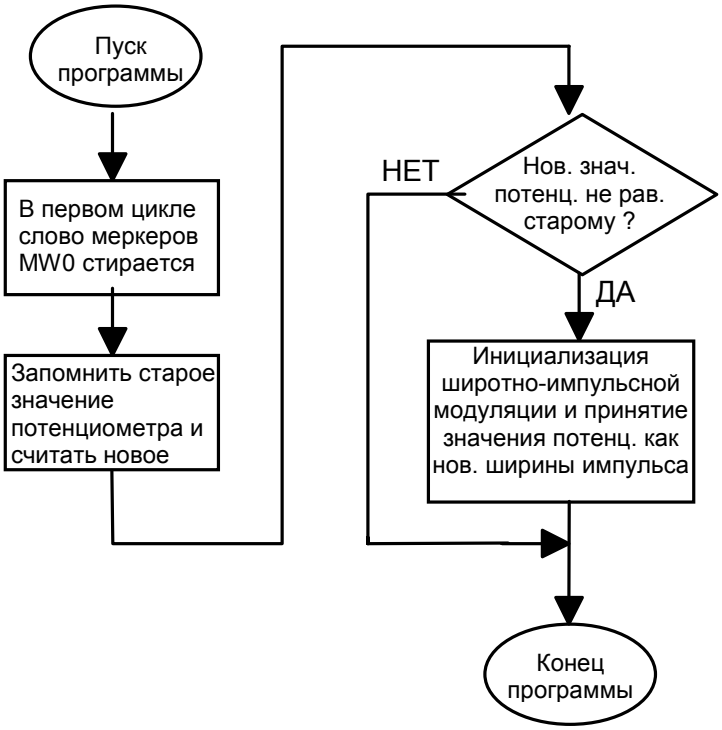
Тема данного примера - регулирование яркости лампы (24В / 1Вт) с помощью встроенных быстрых операций с выходами S7-200. Яркость лампы будет регулироваться через аналоговый потенциометр 0. В зависимости от установки будет изменяться ширина импульса прямоугольного сигнала на выходе A0.0 а следовательно и яркость лампы. Для установки потенциометра необходима отвертка (2,5 мм).

Схема включения



CPU 214 DC/DC/DC

Структура программы



Описание программы вкл. листинг

При каждом выполнении программы значение аналогового потенциометра 0 копируется из специального байта меркеров SMB28 в младший байт MB1 слова меркеров MW0. Для того чтобы согласовать значение с удобным для лампы периодом в 25 мсек, оно делится на 8. Если значение изменилось, то проводится новая инициализация широтно-импульсной модуляции на выходе A0.0. При этом новое значение будет воспринято как ширина импульса в милисекундах.

<u>Пример:</u> SMB28 = 80 (Значение с потенциометра 0)
80 / 8 = 10
10 / 25 (=ширина импульса / длительность периода) => 40% (соотношение напряжение-время)
=> 40% максимальной яркости

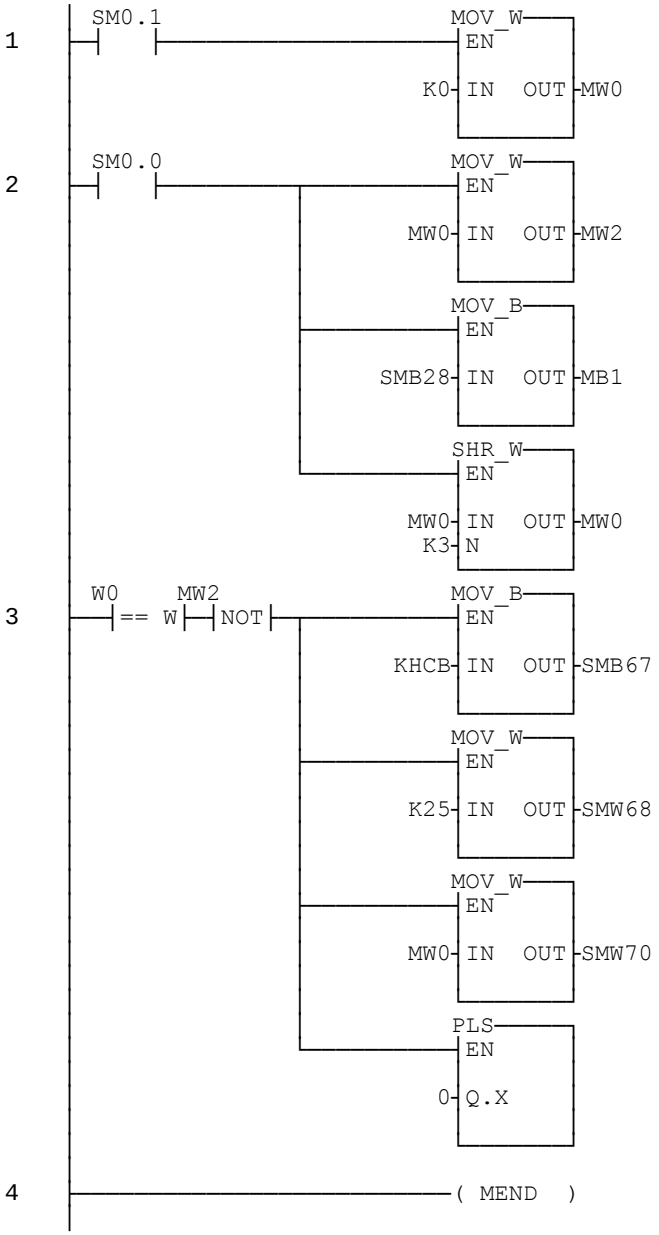
Размер программы составляет 30 слов.

Подробнее о последовательности импульсов Вы найдете в главе 6.3 “Быстрые операции с выходами”. Подробные данные об использовании аналогового потенциометра Вы получите в главе 2.6 “Специальные меркеры” и 2.12 “Дополнительные свойства CPU 214” в руководстве по программированию SIMATIC S7-200.

KOP (S7-MicroDOS)	AWL (IEC)

Основная программа

// TITEL: Регулирование лампы от потенциометра




```
LD      SM0.1     // В первом цикле
MOVW    0, MW0    // слово меркеров MW0
                  // стирается

LD      SM0.0     // Начало стека
MOVW    MW0, MW2
                  // Запомнить старое
                  // значение потенциом.

MOVB    SMB28, MB1
                  // Значение потенциом.
                  // запомнить в
                  // младшем байте MW0

SRW     MW0, 3    // Значение потенциом.
                  // поделить на 8

LDW=    MW0, MW2
                  // При новом устан.знач:
NOT               // (MW0 ( MW2)
MOVB    16#CB, SMB67
                  // Установить контрольн
                  // байт для PWM
MOVW    25, SMW68
                  // Установить длительн.
                  // периода 25 мсек

MOVW    MW0, SMW70
                  // Установить ширину
                  // импульса согласно
                  // значению потенциом.

PLS     0         // Выдать импульсы
                  // на выход A0.0

MEND              // Конец
                  // основной программы
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведённого здесь.

За правильную работу системы заказчик несёт ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

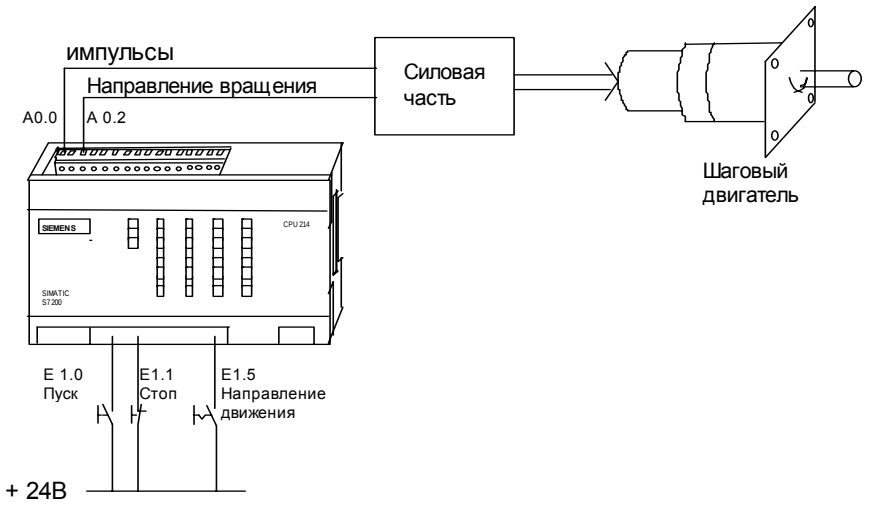
Группа	Пример к теме
5	**Управление приводом шагового двигателя с помощью импульсных выходов S7-200 CPU214 DC/DC/DC**

Краткое описание

CPU 214 оснащен двумя импульсными выходами, разработанные специально для создания управляющих импульсов для привода шагового двигателя. Силовая часть преобразует управляющие имульсы в ток обмоток двигателя по определенному образцу. При этом возникает переменное поле, в котором ротор может изменять свое положение только с фиксированым шагом на шаговый угол (. Постоянная последовательность управляющих импульсов приводит к соответствующей последовательности шагов с постоянной частотой (синхромашина). При достаточно высокой управляющей частоте пошаговое движение вала двигателя переходит в постоянное вращение.

В этом примере описано управление силовой части шагового двигателя через импульсный выход A0.0. По сигналу „ПУСК двигателя“ на E1.0 система управления выдает заданное число прямоугольных импульсов, которое приводит к повороту вала двигателя на соответствующее число шагов. Контакт E1.1 снова отключает двигатель. Положение переключателя E1.5 задает направление вращения двигателя.

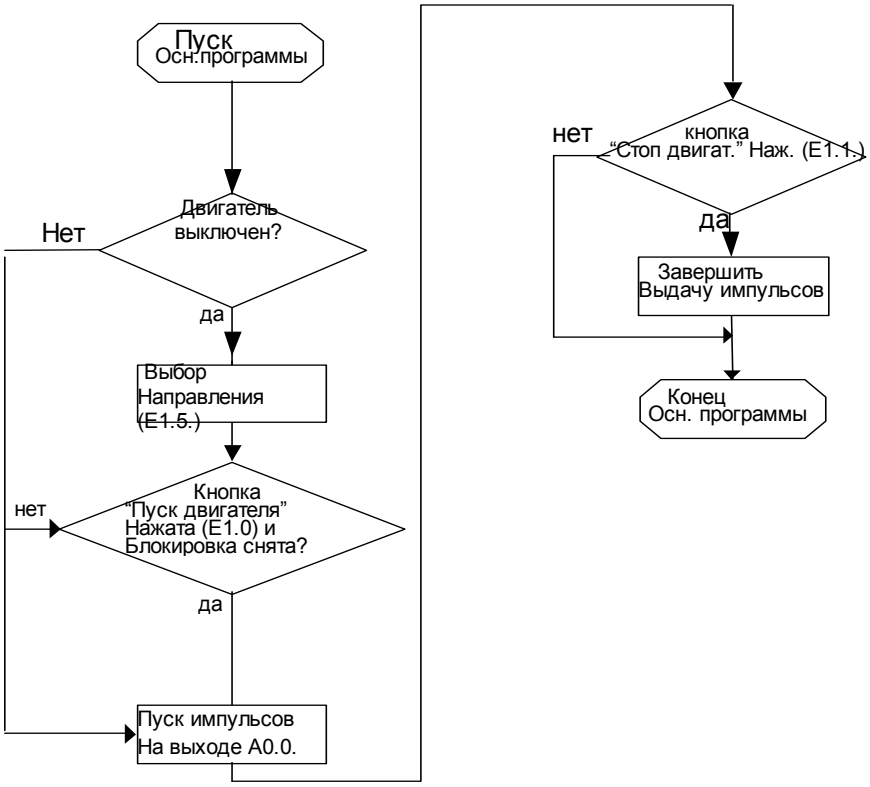
Схема включения:



Аппаратные средства

Кол-во	Приборы	Изготовитель/Закозной номер
1	SIMATIC S7-200 CPU 214	Siemens/ 6ES7 214 - 1AC00 - 0XB0
1	PC/PPI-Кабель	Siemens/ 6ES7 901 - 3BF00 - 0XA0
1	Программатор или PC	
1	Шаговый двигатель с соответствующими силовой частью и кабелем связи	
1	Кабель для управляющих сигналов к силовой части	
1	Переключатель	
2	Кнопка	

Структура программы



Описание программы вкл. листинг

Инициализация

В первом цикле программы (SM0.1=1) производится предустановка важнейших параметров для обеих импульсных функций. Здесь задаются число оборотов и число шаговых импульсов.

Выбор направления вращения

Переключателем E1.5 можно выбирать направление вращения. Если E1.5=1, то выход A 0.2 будет установлен в единицу и значит разрешено движение влево. При E1.5=0 A0.2 устанавливается в ноль и разрешается движение вправо. Для защиты двигателя и чтобы избежать потерь шагов, смена направления движения возможна только при выключенном двигателе (M0.1=0).

Включение привода

1. Нажата кнопка „ПУСК двигателя“, т.е. положительная смена фронта на входе E1.0
2. Блокировка отсутствует, т.е. меркер блокировки M0.2 сброшен.
3. Привод выключен, т.е. меркер работы M0.1 сброшен.

Если эти условия выполнены, то будет установлен меркер работы M0.1 и система управления, по команде „PLS 0“, начинает выдачу импульсов на выход A0.0. Предпосылкой для запуска процесса является то, что ключевые данные функции выдачи импульсов, такие как временной базис, длительность периода и число импульсов, были занесены в соответствующие специальные меркеры операциями PTO/PWM.

Выключениепривода

1. Нажата кнопка „ОСТАНОВ двигателя“, т.е. положительная смена фронта на входе E1.1.
2. Привод включен, т.е. M0.1 установлен.

Если эти условия выполнены, то меркер M0.1 сбрасывается. Затем следует прерывание выдачи импульсов на выход A0.0, в то время как активизируется модуляция ширины импульса в связи с командой „PLS 0“. Ширина импульса снижается при этом до нуля (необходимые для этого базовые установки были произведены уже при инициализации в первом цикле) и тем самым подавляется выходной сигнал.

После выдачи комплектной последовательности импульсов будет обработана подпрограмма прерывания 0, в которой снова сбрасывается меркер M0.1, для того чтобы позволить новое включение.
В диаграмме процесса эта часть программы для большей наглядности не выполняется.

Блокировка

На основании защиты персонала и установки предусмотрена блокировка привода после нажатие кнопки „ОСТАНОВ двигателя“ (E 1.1). При появлении сигнала ОСТАНОВа будет установлен меркер M0.2 и привод немедленно отключается. Снова двигатель может быть запущен, только если будет сброшен M0.2. Чтобы избежать неконтролируемого запуска двигателя при деблокировке клавиши „ОСТАНОВ“, сброс M0.2 возможен только в том случае, если деблокированы обе кнопки на E1.0 и E1.1. Для дальнейшего движения требуется повторная подача сигнала запуска.

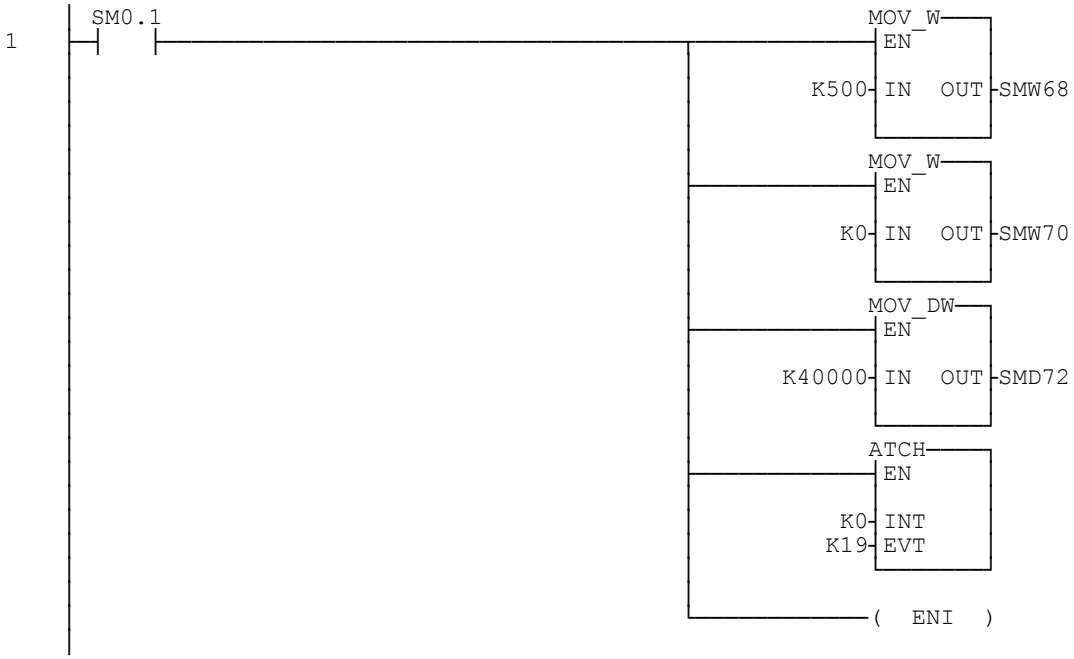
Размер программы составляет 64 Слова.

Подробнее о последовательностях импульсов Вы найдете в главе 6.3 “Быстрые операции вывода”. Подробную информацию о подпрограммах прерываний Вы найдете в главе 6.2 “Подпрограммы прерываний” в руководстве по программированию SIMATIC S7-200.

KOP (S7-MicroDOS)	AWL (TOOLITE2)

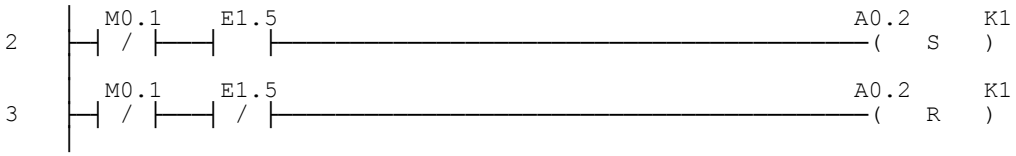
Основная программа

// TITEL = Функциональная проверка привода



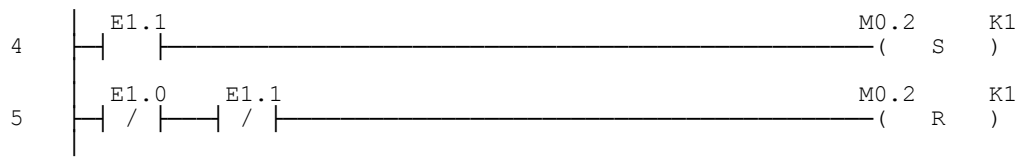

```
LD      SM0.1            // Устанавливается только в первом цикле
MOVW    500, SMW68       // Длительность периода 500 мкс
MOVW    0, SMW70         // Ширина импульса 0 (при широтно-импульсной модуляции)
MOVD    40000, SMD72     // выдача 40.000 импульсов
ATCH    0, 19            // Назначение прерывания 0 (последовательность импульсов
                         // завершена)
ENI                      // Деблокировка прерывания
```

// Деблокировка направления вращения



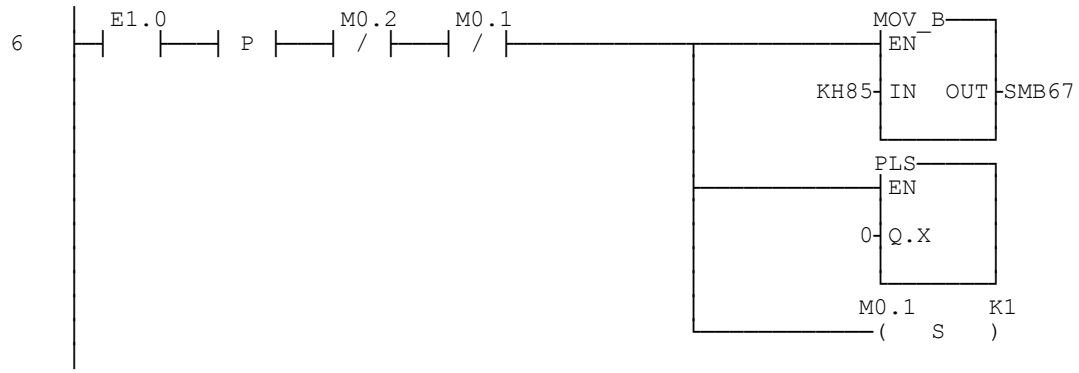

```
LDN     M0.1             // Двигатель выключен
U       E1.5             // ... и переключатель направления в 1
S       A0.2, 1          // Деблокировка вращения влево
LDN     M0.1             // Двигатель выключен
UN      E1.5             // ... и переключатель направления в 0
R       A0.2, 1          // Деблокировка вращения вправо
```

// Блокировка



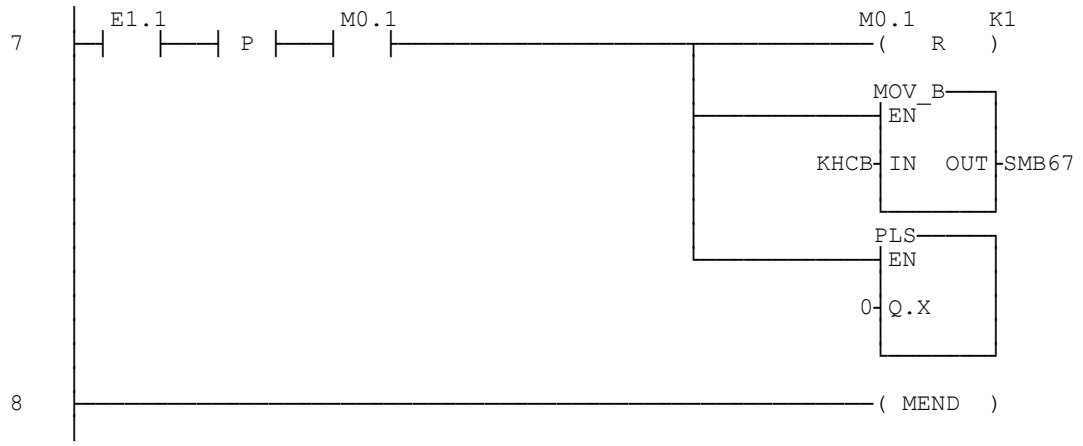
LD	**E1.1**	**// Нажатие кнопки „ОСТАНОВ двигателя“**
S	**M0.2, 1**	**// Блокировка активизирована**
LDN	**E1.0**	**// Сигнал ВКЛ отсутствует**
UN	**E1.1**	**// Обе кнопки деблокированы**
R	**M0.2, 1**	**// Блокировка отключена**

// Запуск привода



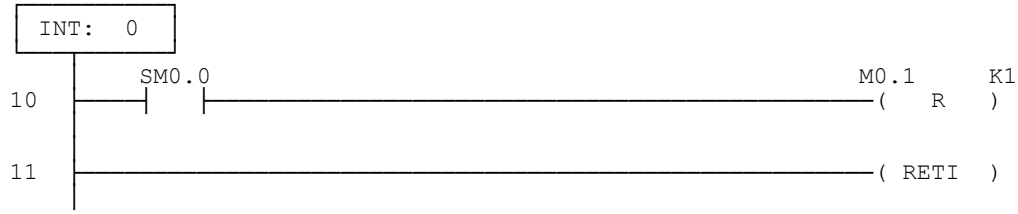
LD	**E1.0**	
EU		**// Нажатие кнопки „ПУСК двигателя“**
UN	**M0.2**	**// ... и блокировка неактивизирована**
UN	**M0.1**	**// ... и привод отключен**
MOVB	**16#85, SMB67**	**// Активизирована функция выдачи импульсов**
PLS	**0**	**// Начало выдачи импульсов на A0.0**
S	**M0.1, 1**	**// Установка меркера „Двигатель работает“**

// Останов привода



LD	**E1.1**	
EU		**// Нажатие кнопки „ОСТАНОВ двигателя“**
U	**M0.1**	**// ... и двигатель работает**
R	**M0.1, 1**	**// Сброс меркера „Двигатель работает“**
MOVB	**16#CB, SMB67**	**// Активизация широтно-импульсной модуляции**
PLS	**0**	**// Завершние выдачи импульсов на A0.0**
MEND		**// Конец основной программы**

Подпрограмма прерываний



INT 0		**// Подпрограмма прерываний 0 "Выдача импульсов завершена"**
LD	**SM0.0**	
R	**M0.1, 1**	**// Сброс меркера "Двигатель работает"**
RETI		**// Конец подпрограммы прерываний 0**

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в КОР-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме КОР. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

Copyright ©1996 by SIEMENS

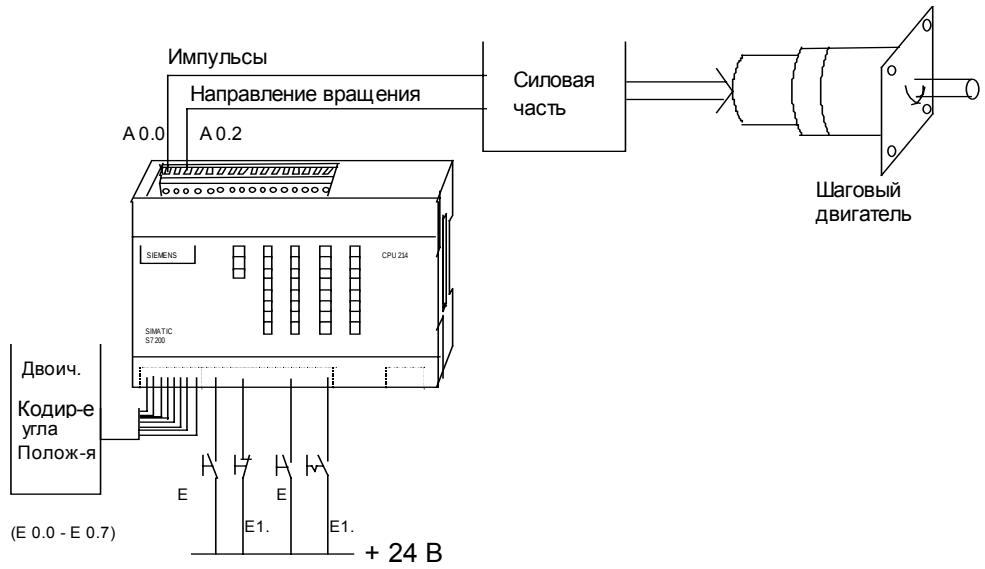
SIMATIC
S7-200 Примеры

Группа	Пример к теме
5	**Управляемое позиционирование привода шагового двигателя через импульсные выходы S7-200 CPU 214 DC/DC/DC**

Краткое описание

В задачах позиционирования различают режимы *регулирования* и *управления*. Шаговые двигатели не требуют постоянного регулирования положения и могут работать в режиме управления. В данном примере программирования производится *относительное* позиционирование с приводом шагового двигателя через встроенные импульсные выходы CPU 214. Хотя при таком режиме позиционирования базовая точка на требуется, здесь описан также простейшие действия по заданию базовой метки, так как на практике появляется необходимость, присвоить оси фиксированную точку отсчета. Затем пользователь задает ЦПУ угол позиционирования в двоичном виде через входной байт. Программа пользователя вычисляет число необходимых шагов позиционирования и приказывает ЦПУ выдать соответствующее число управляющих импульсов.

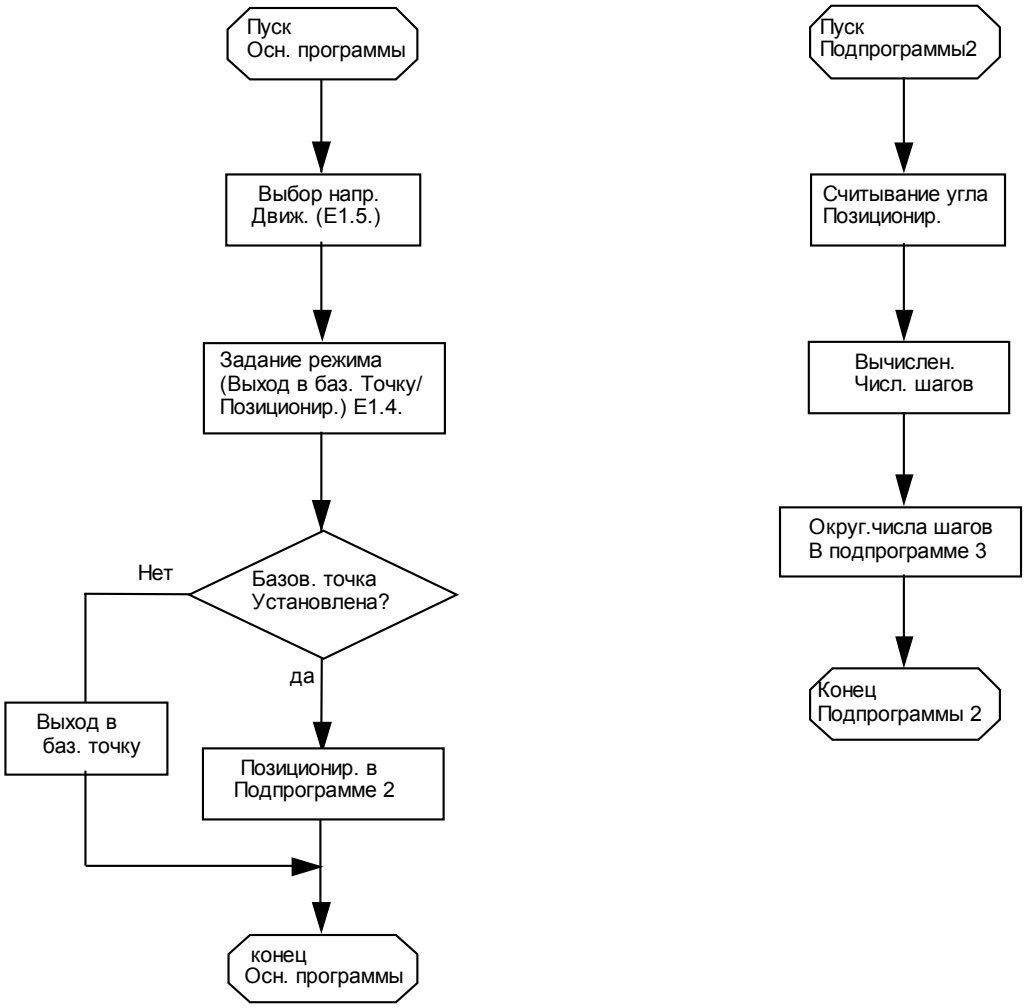
Схема включения



Аппаратные средства

Кол-во	Приборы	Изготовитель/Заказной номер
1	SIMATIC S7-200 CPU 214	Siemens/ 6ES7 214 - 1AC00 - 0XB0
1	PC/PPI-Кабель	Siemens/ 6ES7 901 - 3BF00 - 0XA0
1	Программатор или PC	
1	Шаговый двигатель с соответствующими силовой частью и кабелем связи	
1	Кабель для управляющих сигналов к силовой части	
9	Переключатель	
3	Кнопка	

Структура программы



Описание программы вкл. листинг

В первом цикле программы (SM0.1=1) производится предустановка важнейших параметров. Выбор направления вращения и блокировка кнопок производится аналогично примеру применения No. 22.

Установка и стирание базовой точки

Если базовая точка ещё не установлена, то нажатие кнопки „ПУСК двигателя“ (E 1.0) запустит выход в базовую точку. ЦПУ выдаёт при этом максималsное возможное число управляющих импульсов. В желаемой базовой точке нажмите кнопку „Установка/стирание базовой точки“ (E1.4). При этом будет вызвана подпрограмма, останавливающая двигатель. В заключение меркер базовой точки M0.3 будет установлен в 1, а выход A1.0 покажет новый режим работы „Позиционирование активизировано“.

Если нажата кнопка E1.4 и активизировано „Позиционирование“ (M0.3=1), то происходит переключение в режим работы „Выход в базовую точку“. В подпрограмме при этом будет сброшен M0.3 и сброшена индикация „Позиционирование активизировано“. Дополнительно система управления будет подготовлена к выдаче максимального числа управляющих импульсов.
Повторным нажатием кнопки E1.4 можно производить переключение с одного режима работы на другой. Если сигнал от кнопки приходит при работающем двигателе, то двигатель в любом случае будет остановлен.

На практике, вместо нажатия кнопки вручную, на привод жёстко крепится датчик базовой точки, позволяющий повторно выезжать в базовую метку.

Управляемое позиционирование

Если задана базовая точка (M0.3=1) и отсутствует блокировка (см. Пример No. 22), то будет проведено относительное позиционирование. Система управления считывает при этом в подпрограмме 2 угол позиционирования в двоичной форме из байта входов EB0 в байт меркеров MB11. Соответствующее этому углу число импульсов рассчитывается по следующей формуле:

$$n = \frac{\varphi}{360°} * S \,.$$

n = число управляющих импульсов
(= угол перемещения в градусах
S = число шагов на оборот двигателя.

Используемый в примере шаговый двигатель работает в режиме полушага с S=1000. Вычисленное число шагов округляется затем в подпрограмме 3. Теперь если на E1.0 появится сигнал „ПУСК двигателя“, то ЦПУ выдаст на выход A0.0 вычисленное число управляющих импульсов и двигатель повернется на соответствующее число шагов. Для внутреннего маркирования состояния „Двигатель в движении“ меркер M 0.1 устанавливается в 1.

После полной выдачи импульсов будет обработана подпрограмма прерываний 0 (ср. Пример No. 22, стр. 3). M0.1 сдесь сбрасывается, чтобы позволить новое включение привода. Как и в примере No. 22 этот шаг не включен в диаграмму работы, с тем чтобы оставить график наглядным.

Отключение привода
Вращение двигателя может быть, аналогично 1му примеру (стр. 3), в любой момент завершено нажатием кнопки „ОСТАНОВ двигателя“ (E1.1). Необходимые для этого команды находятся в Подпрограмме 0.

Размер программы составляет 155 слов.

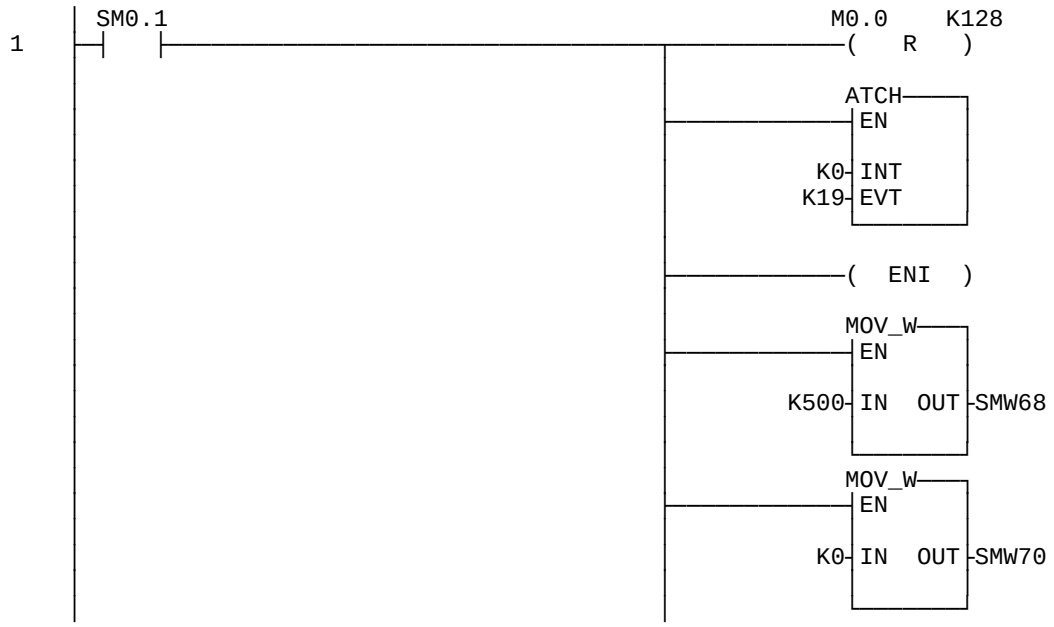
Подробнее о последовательностях импульсов Вы найдёте в главе 6.3 “Быстрые операции вывода”. Подробную информацию о подпрограммах прерываний Вы найдёте в главе 6.2 “Подпрограммы прерываний” в руководстве по программированию SIMATIC S7-200.

KOP (S7-MicroDOS)	AWL (TOOLITE2)

Основная программа

// TITEL = Управляемое позиционирование через импульсные выходы

// Входы:	**E0.0 - E0.7**	**Угол позиционирования в градусах (двоичный код)**
//	**E1.0**	**Кнопка „ПУСК двигателя“**
//	**E1.1**	**Кнопка „ОСТАНОВ двигателя“**
//	**E1.4**	**Кнопка „Установить/стереть базовую точку“**
//	**E1.5**	**Переключатель направления движения**
// Выходы:	**A0.0**	**Импульсный выход**
//	**A0.2 (0=вправо/1=влево)**	**Сигнал направления движения**
//	**A1.0**	**Индикация режима работы**
// Меркеры:	**M0.1**	**Двигатель в движении**
//	**M0.2**	**Меркер блокировки**
//	**M0.3**	**Меркер базовой точки**
//	**MD8, MD12**	**Вспомогательный меркер**



```
                                                  MOV_DW
                                                 EN
                                        KH1999997C IN   OUT SMD72
```

LD	SM0.1	// Устанавливается только в первом цикле
R	M0.0, 128	// Сброс MD0 - MD12
ATCH	0, 19	// Назначение подпрограммы прерыванийt 0 (Завершена // последовательность импульсов)
ENI		// Прерывание деблокировано
MOVW	500, SMW68	// Длительность периода T = 500 мксек
MOVW	0, SMW70	// Ширина импульса = 0 при широтно-импульсной модуляции
MOVD	16#1999997C, SMD72	// макс. число импульсов для выхода в базовую точку

// Деблокировка вращения влево

```
    M0.1    E1.5                                    A0.2     K1
2  ─┤ / ├──┤   ├──────────────────────────────────(   S   )
```

LDN	M0.1	// Двигатель выключен
U	E1.5	// а переключатель направления движения = 1
S	A0.2,1	// Деблокировка вращения влево

// Деблокировка вращения вправо

```
    M0.1    E1.5                                    A0.2     K1
3  ─┤ / ├──┤ / ├──────────────────────────────────(   R   )
```

LDN	M0.1	// Двигатель выключен
UN	E1.5	// ... а переключатель направления движения = 0
R	A0.2, 1	// Деблокировка вращения вправо

// Активирование блокировки

```
    E1.1                                            M0.2     K1
4  ─┤   ├──────────────────────────────────────────(   S   )
```

LD	E1.1	// Нажата кнопка "ОСТАНОВ двигвтеля"
S	M0.2, 1	// Блокировка активирована

// Подавление блокировки

```
   |  E1.1    E1.0                                          M0.2     K1
5  |-| / |---| / |-----------------------------------------(   R   )
   |
```

LDN	**E1.1**	
UN	**E1.0**	**// Обе кнопки деблокированы**
R	**M0.2, 1**	**// Подавление блокировки**

// Задание режима работ (Выход в базовую точку / Позиционирование)

```
   |  E1.4                                                   1
6  |-|   |---| P |-----------------------------------------( CALL  )
   |
```

LD	**E1.4**	
EU		**// Нажата кнопка "Установка/стирание базовой точки"**
CALL 1		**// Вызов подпрограммы 1**

// Запуск привода

```
   |  E1.0          M0.1    M0.2  SMD72     K1            MOV_B---┐
7  |-|   |---| P |---| / |---| / |-----| >= D |---┬-------------|EN     |
   |                                              |          |       |
   |                                              |     KH85-|IN  OUT|-SMB67
   |                                              |          |       |
   |                                              |          └-------┘
   |                                              |          PLS-----┐
   |                                              |-------------|EN     |
   |                                              |          0-|Q.X    |
   |                                              |          |       |
   |                                              |          └-------┘
   |                                              |          M0.1     K1
   |                                              └-------------(   S   )
   |
```

LD	**E1.0**	
EU		**// Нажата кнопка "ПУСК двигателя"**
UN	**M0.1**	**// ... и двигатель в покое**
UN	**M0.2**	**// ... и блокировка неактивна**
UD>=	**SMD72, 1**	**// и число шагов >= 1**
MOVB	**16#85, SMB67**	**// Активирование функции выдачи импульсов PTO**
PLS	**0**	**// Пуск выдачи импульсов на A0.0**
S	**M0.1, 1**	**// Установка меркера "Двигатель в движении"**

Copyright ©1995 by SIEMENS

// Позиционирование

```
    |  M0.3    M0.1                                          2
8   ├─┤   ├───┤ / ├──────────────────────────────────( CALL  )
    |
```

LD	**M0.3**	**// Активен режим работы "Позиционирование"**
UN	**M0.1**	**// ... и двигатель в покое**
CALL 2		**// Вызов подпрограммы 2**

// Останов привода

```
    |  E1.1          M0.1                                      0
9   ├─┤   ├───┤ P ├───┤   ├──────────────────────────────( CALL  )
    |
10  ├──────────────────────────────────────────────────────( MEND  )
    |
```

LD	**E1.1**	
EU		**// Нажата кнопка "ОСТАНОВ двигателя"**
U	**M0.1**	**// и двигатель в движении**
CALL 0		**// Вызов подпрограммы 0**
MEND		**// Конец основной программы**

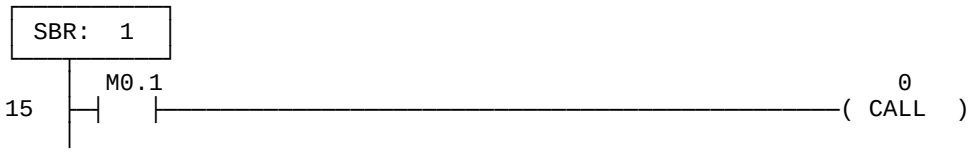
Подпрограммы

// Подпрограмма 0

```
 SBR:  0
    |
    |  SM0.0                                          MOV_B
12  ├──┤   ├──────────────────────────┬──────────────┤EN
    |                                 |              |
    |                                 |      KHCB-IN  OUT-SMB67
    |                                 |              |
    |                                 |              PLS
    |                                 ├──────────────┤EN
    |                                 |              |
    |                                 |          0-Q.X
    |                                 |              |
    |                                 |          M0.1     K1
    |                                 └──────────────(   R   )
    |
13  ├──────────────────────────────────────────────(  RET  )
    |
```

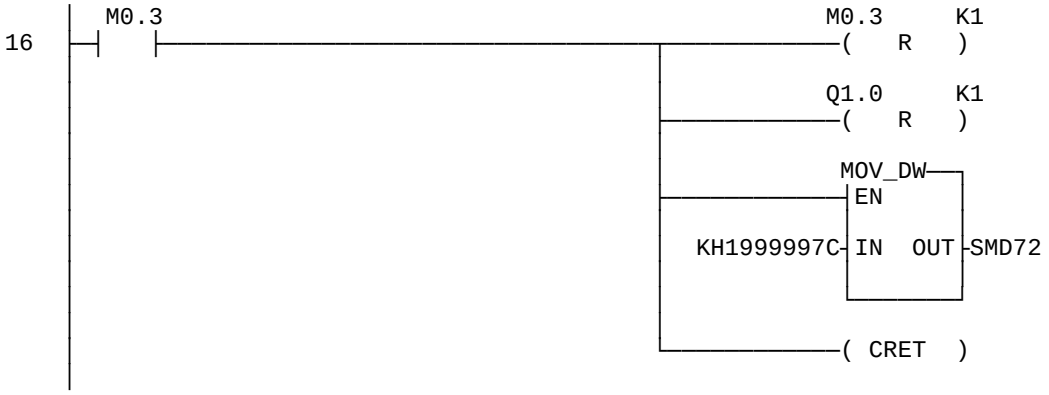
```
SBR 0                          // Подпрограмма 0 "Останов привода"
LD        SM0.0
MOVB      16#CB, SMB67         // Активирование широтно-импульсной модуляции
PLS       0                    // Останов выдачи импульсов на A0.0
R         M0.1, 1              // Сброс меркера "Двигатель в движении"
RET                            // Конец подпрограммы 0
```

// Подпрограмма 1



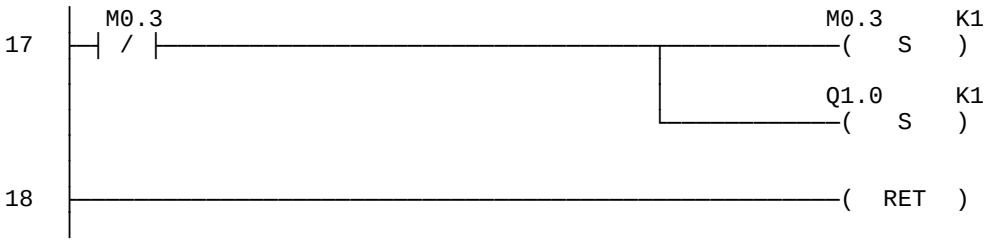

```
SBR 1                          // Подпрограмма 1 "Задание режима работы“
LD        M0.1                 // Двигатель в движении
CALL 0                         // Останов привода
```

// Требование выхода в базовую точку



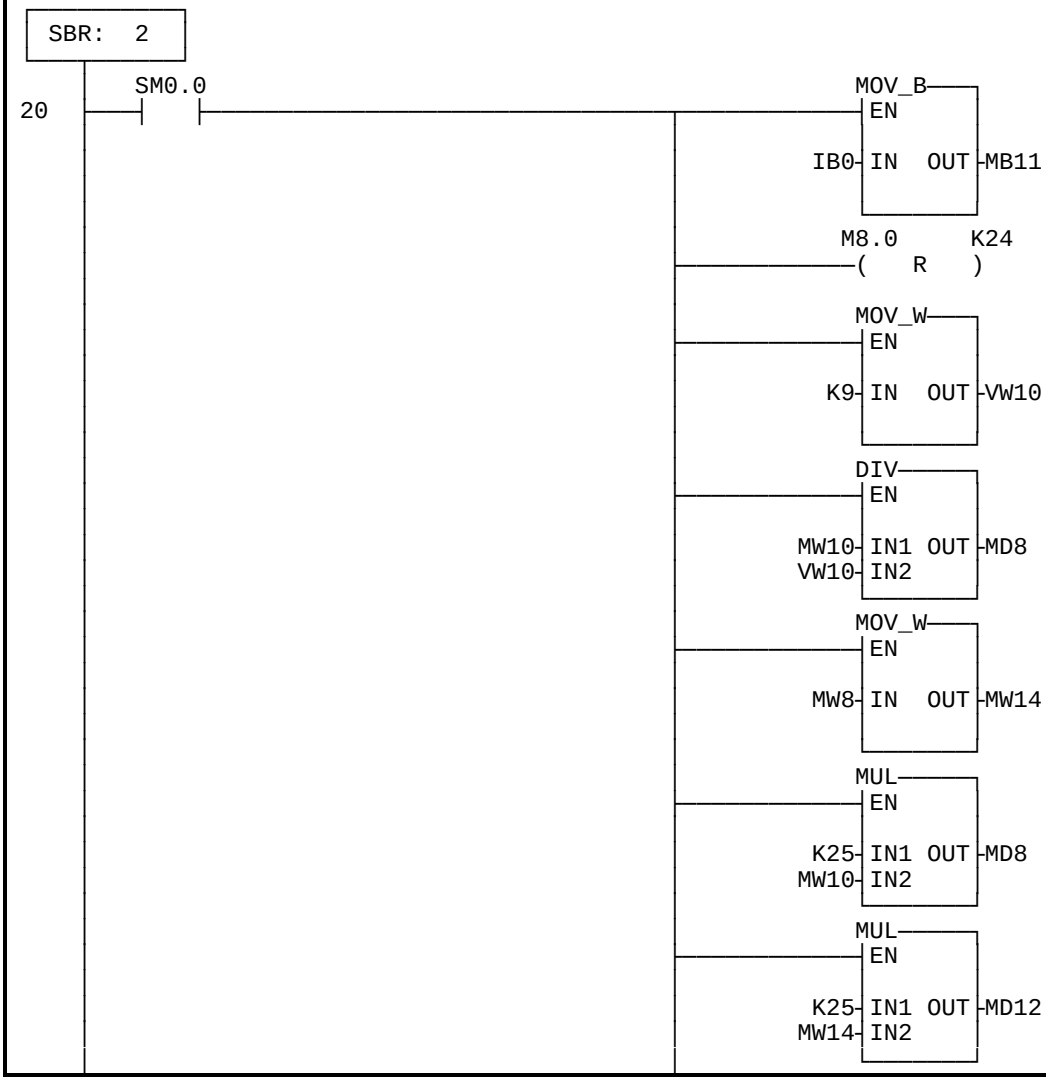

```
LD        M0.3                 // "Позиционирование" активно
R         M0.3, 1              // Сброс меркера базовой точки
R         A1.0, 1              // Стереть сообщение "Позиционирование активно"
MOVD      16#1999997C, SMD72 // макс. число импульсов для нового выхода в баз.точку
CRET                           // Условный переход в основную программу
```

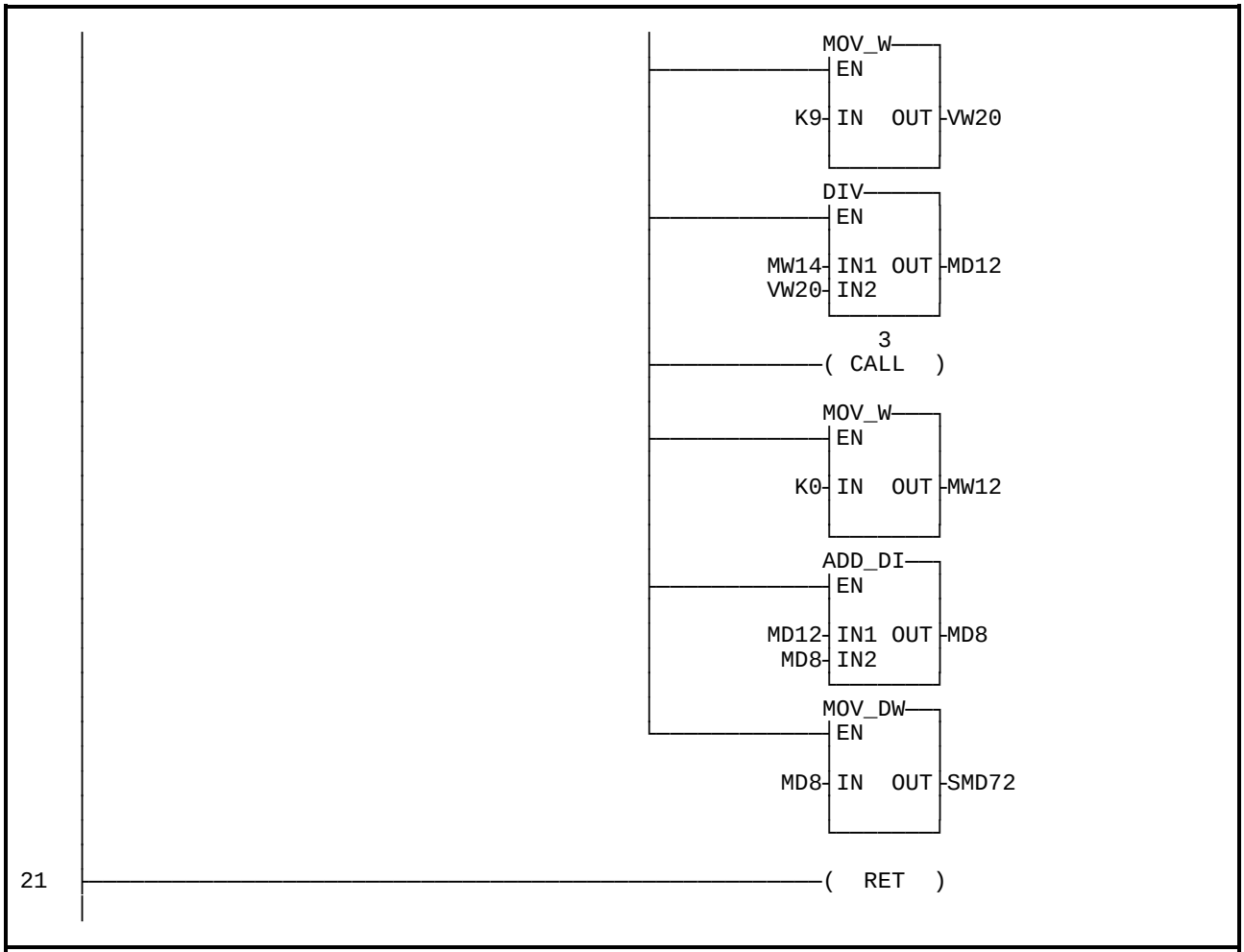
// Требование „Позиционирования“




```
LDN     M0.3        // Базовая точка еще не установлена
S       M0.3, 1     // Установить меркер базовой точки
S       A1.0, 1     // Вывод сообщения "Позиционирование активно"
RET                 // Конец подпрограммы 1
```

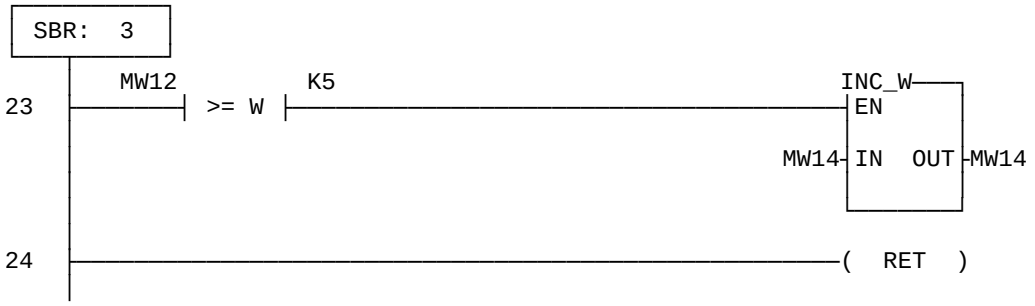
// Подпрограмма 2






```
SBR 2                    // Подпрограмма 2 "Позиционирование"
LD      SM0.0
MOVB    EB0, MB11        // Скопировать угол поворота из EB0 в MD8
R       M8.0, 24         // Стереть MB8 - MB10
MOVW    9, VW10
DIV     VW10, MD8        // Угол / 9 = q1 + r1
MOVW    MW8, MW14        // r1 занести в MD12
MUL     25, MD8          // q1*25 -> MD8
MUL     25, MD12
MOVW    9, VW20
DIV     VW20, MD12       // r1*25 / 9  = q2 + r2
CALL 3                   // Округлить число шагов в подпрограмме 3
MOVW    0, MW12          // Стереть r2
+D      MD12, MD8        // Занести число шагов в MD8
MOVD    MD8, SMD72       // Передать число шагов в SMD72
RET                      // Конец подпрограммы 2
```

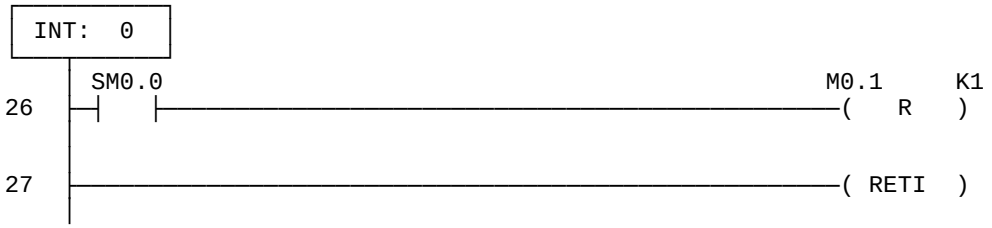
Подпрограмма 3




```
SBR 3                    // Подпрограмма 3 "Округление числа шагов"
LDW>=     MW12, 5        // если r2 >= 5 / 9
INCW      MW14           // число шагов увеличить на 1
RET                      // Конец подпрограммы 3
```

Подпрограммы прерываний

// Подпрограмма прерываний 0 "Завершение выдачи импульсов"




```
INT 0                    // Начало подпрограммы прерываний 0
LD        SM0.0
R         M0.1, 1        // Сброс меркера "Двигатель в движении"
RETI                     // Конец подпрограммы прерываний 0
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

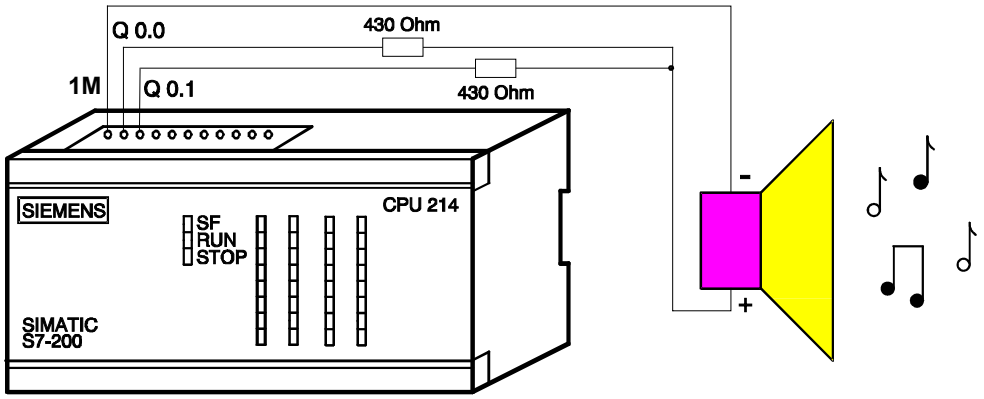
SIMATIC
S7-200 Примеры

Группа	Тема
2	**Исполнение мелодий с использованием импульсных выходов S7-200 CPU 214 DC/DC/DC**

Краткое описание

В данном примере описано, как заставить ПЛК S7-200 CPU 214 DC/DC/DC играть мелодию используя функцию импульсного выхода. Времена цикла 25 аккустических нот записано в таблице с соответствующим числом импульсов, требуемых для поддержания тона в течении 0.125 секунды. Т.к. две ноты играются в то же самое время, две разные таблицы содержат информацию о мелодии для канала 0 и канала 1.
Оба импульсных канала настроены как **Импульсные Серийные Выхода (ИСВ)**. Когда проигрывается первая нота каждого канала, вторая нота каждого канала запрашивается. Образуется очередь из одного элемента (один в обработке, один в очереди).
Программы обработки прерываний привязаны к двум каналам ИСВ. По завершении первой ноты, программа обработки прерываний запрашивает следующую ноту. Этот процесс продолжается пока не закончится мелодия.

Схема подключения



Аппаратные требования

SIMATIC S7-200 CPU 214 DC/DC/DC
Блок питания 115VAC / 24VDC, 0.9A (обычно достаточно 300mA - 400mA)
2 сопротивления - 430 Ом 0,5 Вт
1 - Динамик

Структура программы

Описание программы

Размер программы = 778 слов.
Более подробную информацию о функциях импульсного выхода смотри в Главе 6.3 “Команды быстрого выхода” в Справочном Руководстве по Программированию Step 7-Micro.

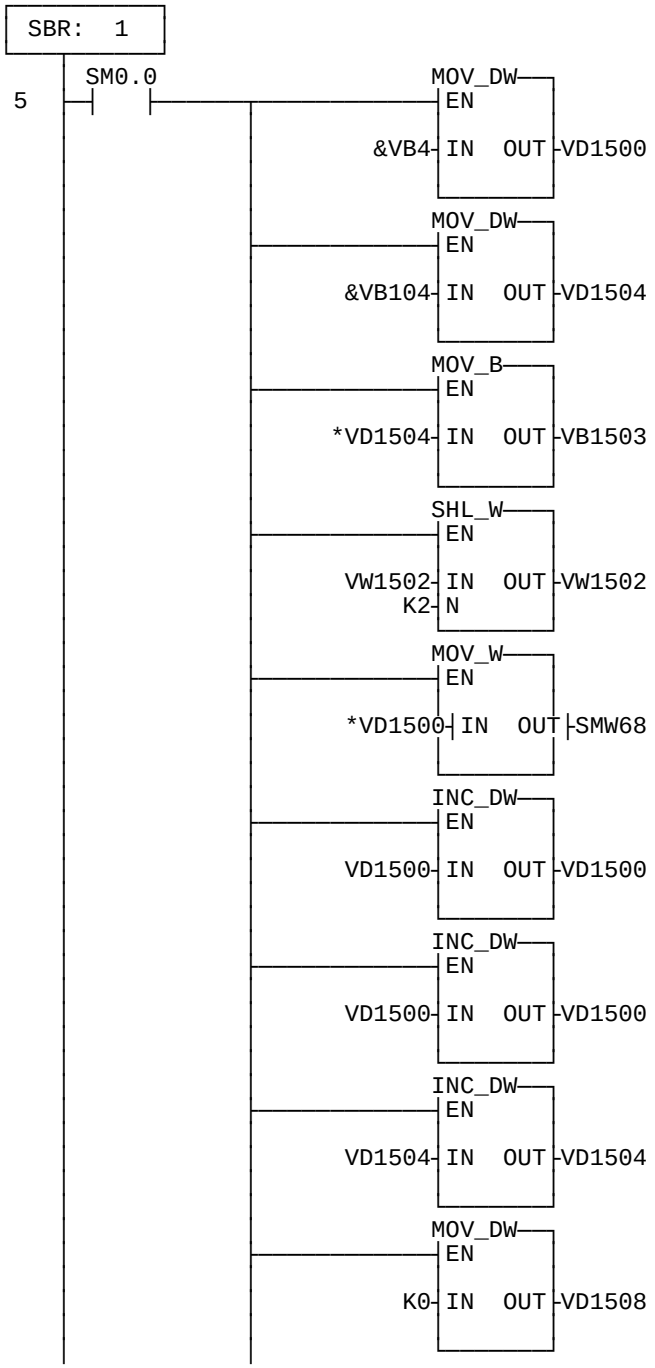
```
// ЗАГОЛОВОК=СВИСТ

// Данная программа играет мелодию.
// Пример работает только с SIMATIC CPU 214 DC/DC/DC, т.к. используются импульсные выходы.

// Подключите выходы Q0.0 и Q0.1 CPU 214 через два сопротивления
// (приблизительно 430 Ом, 0,5 Вт) к одному контакту динамика.
// Другой контакт динамика подключите к общему выходу (1M).
// Подключите +24 источника питания к контакту (1L+) CPU 214 и
// соответственно землю к общему выходу (1M).

// Используемые области памяти:
//      V4-V103             Таблица нот.
//      V104-V244           Песенная таблица канала 0.
//      V248-V504           Песенная таблица канала 1.
//      V1500-V1503         Указатель на таблицу нот.
//      V1504-V1507         Указатель на песенную таблицу канала 0.
//      V1508-V1511         Временный рабочий регистр.
//      V1554-V1557         Указатель на песенную таблицу канала 1.

// Описание программы:
// Времена цикла музыкальных нот для 25 chromatic нот начиная с
// “A” (440 Гц) хранится в таблица вместе с соответствующим числом
// импульсов, необходимых для поддержания тона в течении 0.125 секунд. Отдельная
// таблица содержит данные мелодии для канал 0, а третья таблица
// содержит данные мелодии для канал 1. Данные мелодии содержат два
// байта для каждой ноты: первый байт содержит индексный номер ноты
// (1-25), а второй байт содержит количество 0.125-секундных
// интервалов времени, необходимых для данной ноты.
// Оба импульсных канала настроены как Импульсные Серийные Выхода (ИСВ).
// Когда проигрывается первая нота каждого канала, вторая нота
// каждого канала запрашивается. Образуется очередь из одного элемента
// (один в обработке, один в очереди). Программы обработки прерываний
// привязаны к двум каналам ИСВ. По завершении первой ноты,
// программа обработки прерываний запрашивает следующую ноту.
// Этот процесс продолжается пока не закончится мелодия.

// Структура прогграммы:
//          MAIN            Инициализация программы.
//          SBR 0           Инициализация серии импульсов.
//          SBR 1           Проигрывание первой ноты для каждого канала.
//          SBR 2           Очередь второй ноты для каждого канала.
//          SBR 3           Задание прерываний для продолжения песни.
//          INT 0           Проигрывание следующей ноты для канала 0.
//          INT 1           Проигрывание следующей ноты для канала 1.

// Главная программа исполняется только в первом цикле. Если главное меню закончилось, то
// программа так же заканчивает выполнение. Обычно эта программа не выполняет ничего,
// кроме вызова подпрограмм с заданными действиями.
```

LAD (S7-MicroDOS)	STL (IEC)
1 SM0.1 (CALL 0) (CALL 1) (CALL 2) (CALL 3) 2 (MEND)	LD SM0.1 // Бит первого цикла CALL 0 // Инициализация серии // импульсов CALL 1 // Проигрывание 1ой ноты // для каждого канала. CALL 2 // Очередь второй ноты // для каждого канала. CALL 3 // Задание прерываний // для продолжения песни MEND

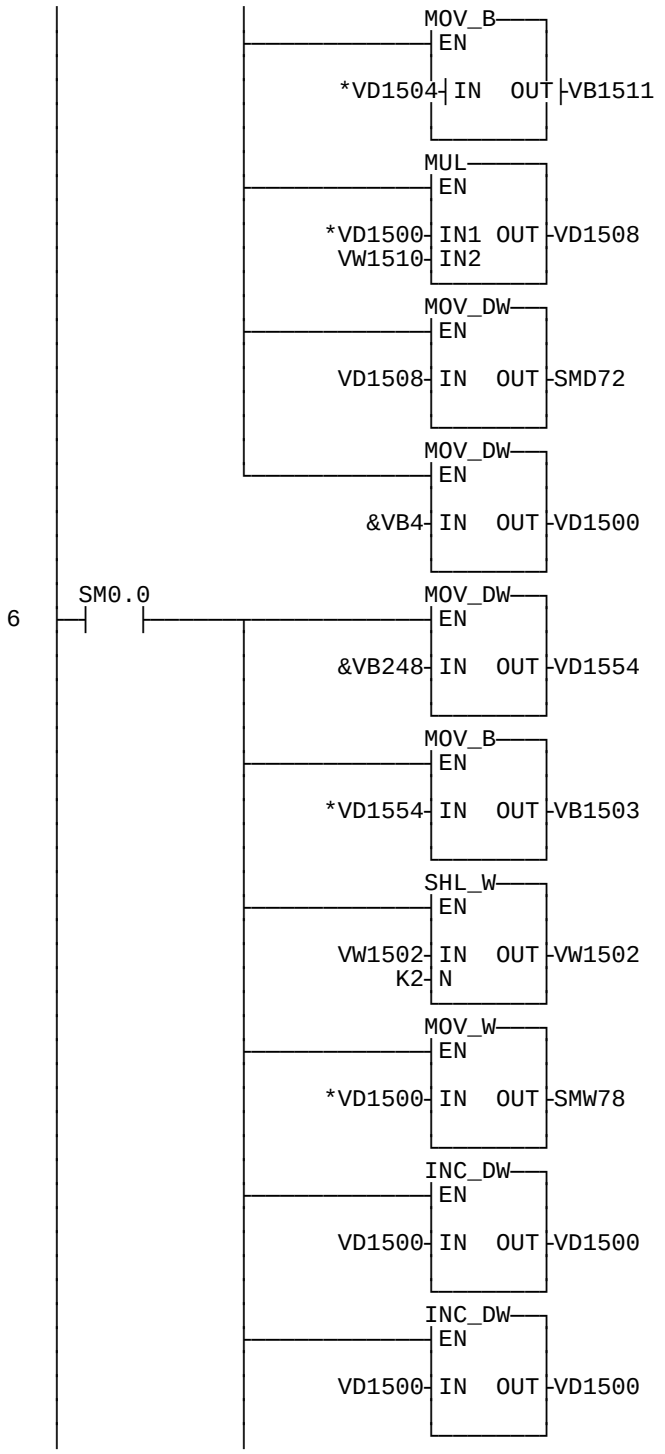
// Подпрограмма 0 инициализирует импульсные серийный выходы. Специальный флаг SMB 67
// задает параметры для выхода Q0.0, а SMB 77 задает параметры для выхода Q0.1.
// В данном случае шестнадцатиричное (16#) 85 означает:
// установить управляющий байт, разрешающий импульсный серийный выход,
// выбрать режим ИСВ = 1 msec / пульс,
// установить значения актуализации счетчика импульсов и времени цикла.

SBR: 0 3 SM0.0 MOV_B EN KH85 IN OUT SMB67 MOV_B EN KH85 IN OUT SMB77 4 (RET)	SBR 0 // Инициализация // серии импульсов LD SM0.0 MOVB 16#85,SMB67 // Задание управляющих // байтов MOVB 16#85,SMB77 // Переслать байт RET // Возврат

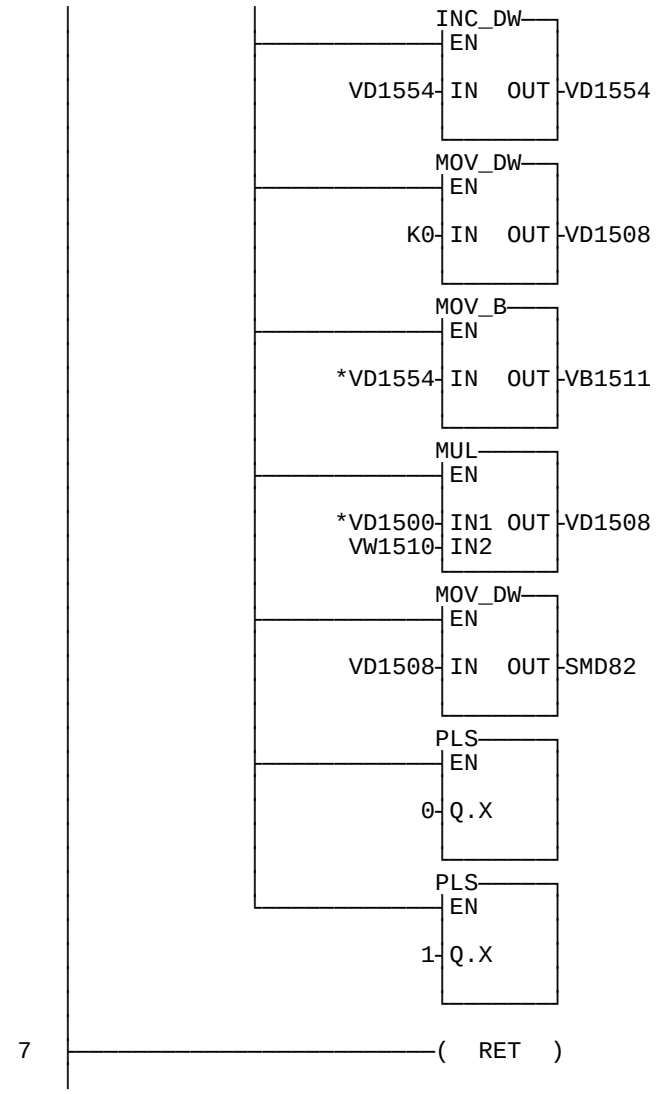
// Подпрограмма 1 проигрывает первую ноту для каждого канала. Индекс ноты канала 0
// и соответствующие единицы времени загружается и передается в импульсный выход 0.
// Та же самая процедура выполняется для канала 1, так что обе ноты проигрываются
// в то же самое время.




```
SBR    1

LD     SM0.0
MOVD   &VB4,VD1500  // Задание
                    // указателя на
                    // таблицу нот

MOVD   &VB104,VD1504 // Задание
                     // указателя на канал 0
                     // песенной таблицы

MOVB   *VD1504,VB1503 // Получить          //
первую ноту для
                     // канала 0

SLW    VW1502,2 // Индекс в таблице
                // (умноженный на 4)

MOVW   *VD1500,SMW68 // Загрузить          //
время цикла для
                     // канала 0

INCD   VD1500       // Указатель на число
                    // импульсов за единицу
                    // времени

INCD   VD1500       // Другой указатель

INCD   VD1504       // Указатель на число
                    // единиц времени

MOVD   0,VD1508 // Очистить V1508.
```

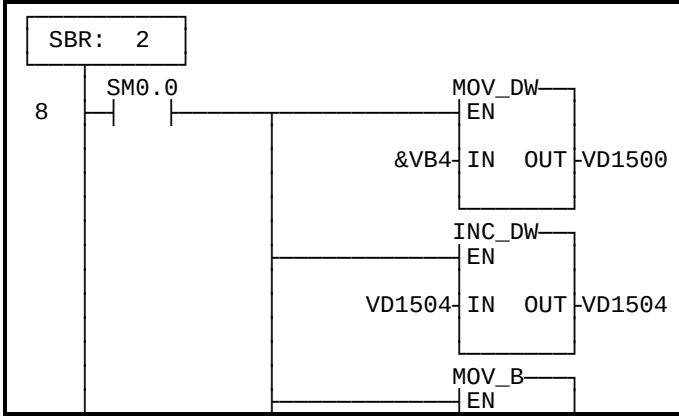



```
MOVB   *VD1504,VB1511 // Получить
                      //количество единиц
                      // времени для этой
                      // ноты

MUL    *VD1500,VD1508 // Вычислить
                      // общий импульсов для
                      // этой ноты

MOVD   VD1508,SMD72 // Загрузить
                    // количество импульсов
                    // для этой ноты

MOVD   &VB4,VD1500  // Указатель на
                    // таблицу нот

LD     SM0.0
MOVD   &VB248,VD1554 // Указатель на
                     // канал 1 песенной
                     // таблицы

MOVB   *VD1554,VB1503  // Получить
                       // первую ноту для
                       // канала 1

SLW    VW1502,2  // Индекс в таблице
                 // (умноженный на 4)

MOVW   *VD1500,SMW78 // Загрузить
                     // время цикла для
                     // канала 1

INCD   VD1500        // Указатель на
                     // количество импульсов
                     // за единицу времени

INCD   VD1500        // Другой указатель
```

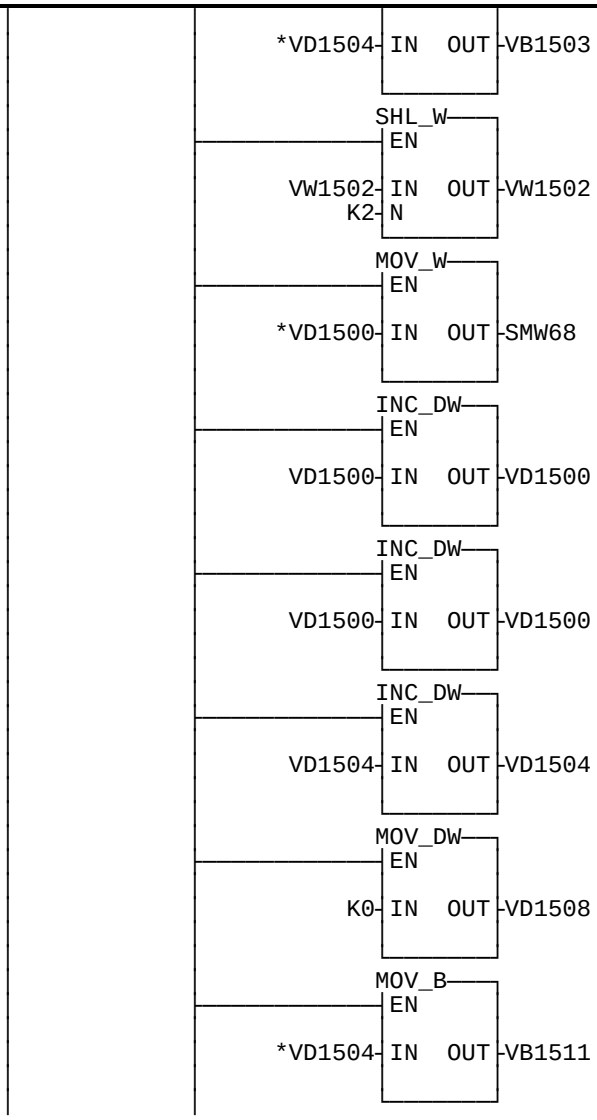



```
INCD    VD1554        // Указатель на
                      // количество единиц
                      // времени

MOVD    0,VD1508  // Очистить V1508

MOVB    *VD1554,VB1511  // Получить
                      // количество единиц
                      // времени для этой
                      // ноты

MUL     *VD1500,VD1508  // Вычислить
                      // общее импульсов
                      // для этой ноты

MOVD    VD1508,SMD82  // Загрузить
                      // количество импульсов
                      // для этой ноты

PLS     0             // Играть первую ноту
                      // для каждого канала

PLS     1             // Играть другую ноту

RET                   // Возврат
```

// Подпрограмма 2 queues the вторую ноту для каждого канал. Индекс ноты канала 0
// и соответствующее количество единиц время загружается и передается в импульсный
// выход 0. Та же самая процедура выполняется для канала 1, так что обе ноты
// проигрываются в то же самое время.




```
SBR     2

LD      SM0.0
MOVD    &VB4,VD1500  // Указатель на
                     // таблицу нот

INCD    VD1504        // Указатель на вторую
                      // ноту для канала 0
```



MOVB *VD1504,VB1503 // Получить
// вторую ноту для
// канала 0

SLW VW1502,2 // Индекс в таблице
// (умноженный на 4)

MOVW *VD1500,SMW68 // Загрузить
// время цикла для
// канала 0

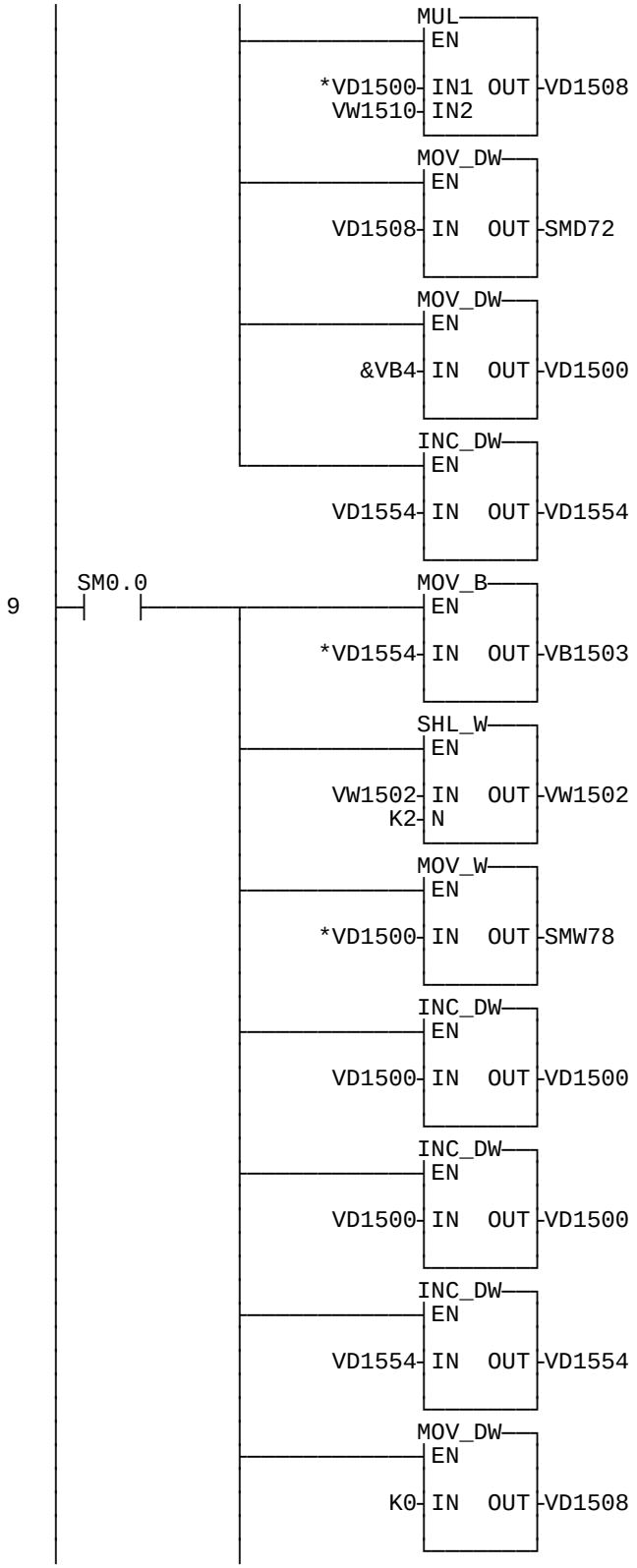
INCD VD1500 // Указатель на
// количество импульсов
// за единицу времени

INCD VD1500 // Другой указатель

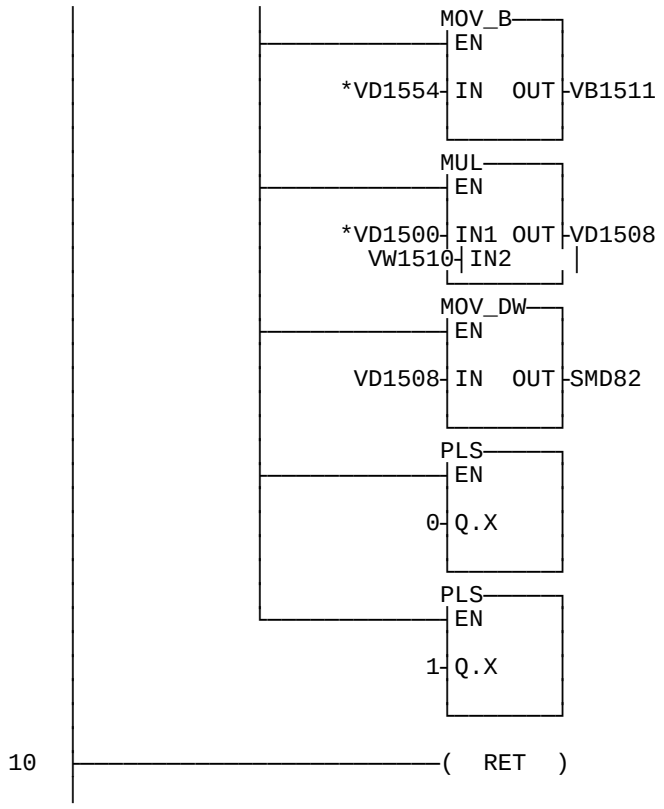
INCD VD1504 // Указатель на
// количество единиц
// времени

MOVD 0,VD1508 // Очистить V1508

MOVB *VD1504,VB1511 // Получить
// количество единиц
// времени для этой
// ноты

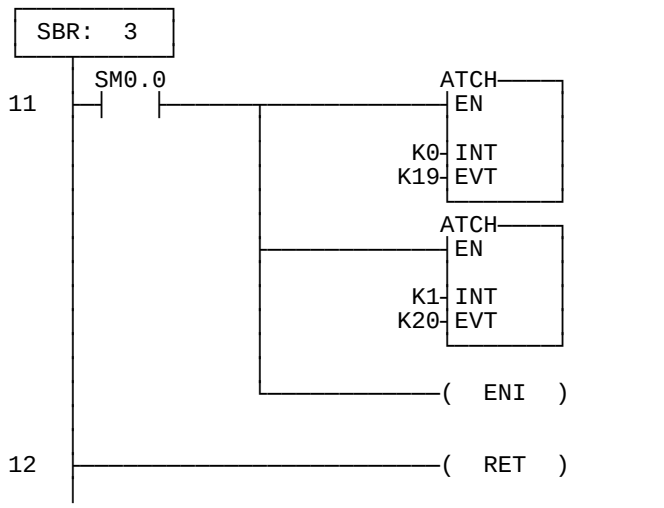



```
MUL     *VD1500,VD1508  // Вычислить
                        // общее импульсов
                        // для этой ноты

MOVD    VD1508,SMD72    // Загрузить
                        // количество импульсов
                        // для этой ноты

MOVD    &VB4,VD1500     // Указатель на
                        // таблицу нот

INCD    VD1554          // Указатель на вторую
                        // ноту для канала 1

LD      SM0.0
MOVB    *VD1554,VB1503  // Получить
                        // вторую нота для
                        // канала 1

SLW     VW1502,2        // Индекс в таблице
                        // (умноженный на 4)

MOVW    *VD1500,SMW78   // Загрузить
                        // время цикла для
                        // канала 1

INCD    VD1500          // Указатель на
                        // количество импульсов
                        // за единицу времени

INCD    VD1500          // Другой указатель

INCD    VD1554          // Указатель на
                        // количество единиц
                        // времени

MOVD    0,VD1508        // Очистить V1508
```




```
MOVB    *VD1554,VB1511  // Получить
                        // количество единиц
                        // времени для этой
                        // ноты

MUL     *VD1500,VD1508  // Вычислить
                        // общее импульсов
                        // для этой ноты

MOVD    VD1508,SMD82  // Загрузить
                        // количество импульсов
                        // для этой ноты

PLS      0    // Очередь второй
                        // для каждого канала

PLS      1    // Очередь другой ноты

RET                     // Возврат
```

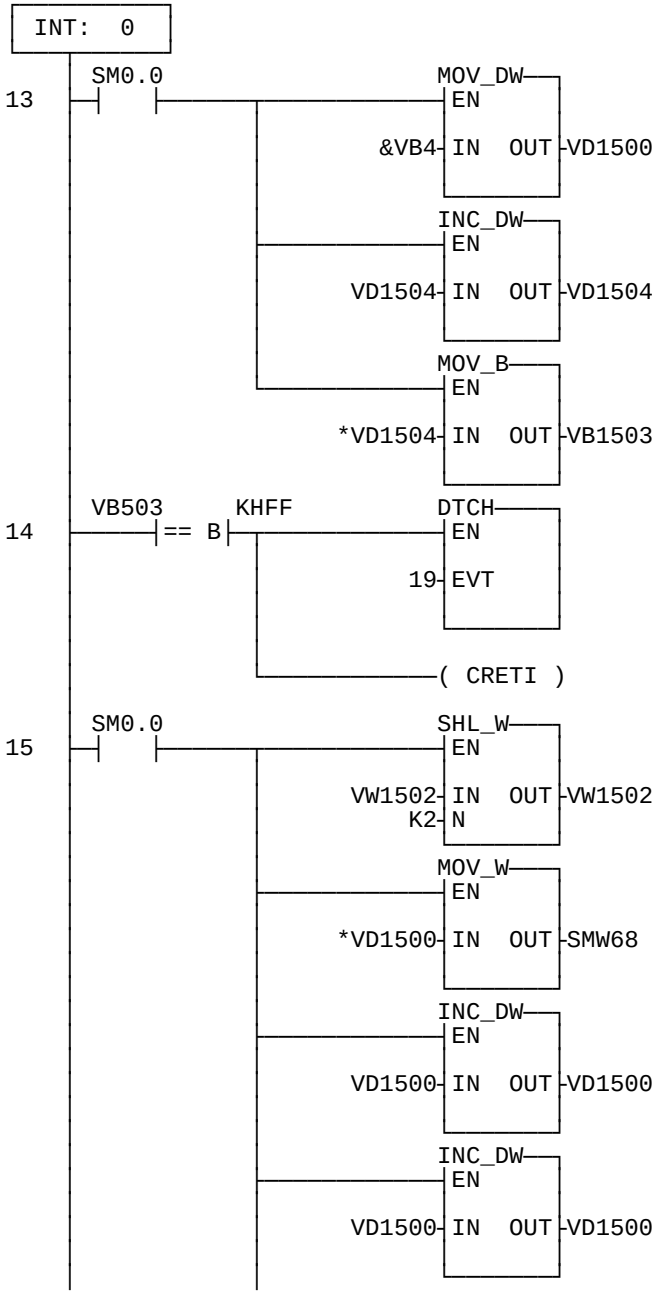
// Подпрограмма 3 задает прерывания, необходимые для продолжения музыки.
// В данной программе присваиваются прерывания. Прерывание 0 - подсчитывает
// импульсы для канала 0 - событие (19), второе - импульсы для канала 1 - событие (20).
// Если нота закончена, возникает прерывание.



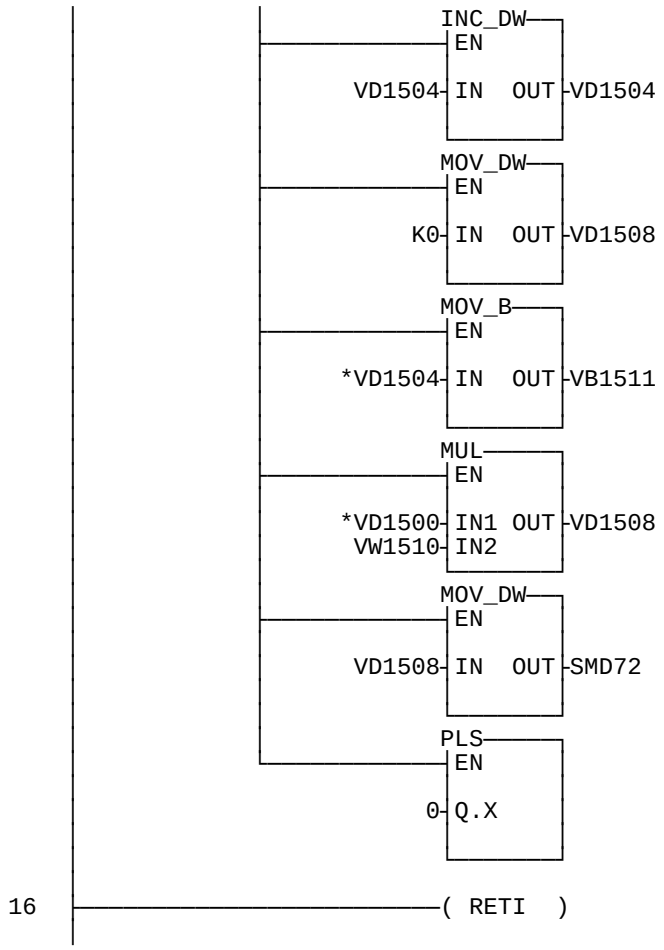

```
SBR     3

LD      SM0.0
ATCH   0, 19    // Присоединить
                        // прерывание 0 к PLS0
                        // завершить событие
                        // (Событие 19)

ATCH   1, 20     // Присоединить
                        // прерывание 1 к PLS1
                        // завершить событие
                        // (Событие 20)

ENI                     // Разрешить
                                // прерывания

RET                     // Возврат
```

Программы обработки прерываний

// Прерывание 0 активируется, если заканчивает звучать нота канала 0. Индекс следующей
// ноты и количество время единиц подсчитывается и передается на импульсный выход
// канала 0. Если звучит последняя нота данного канала, прерывание снимается и программа
// заканчивается.

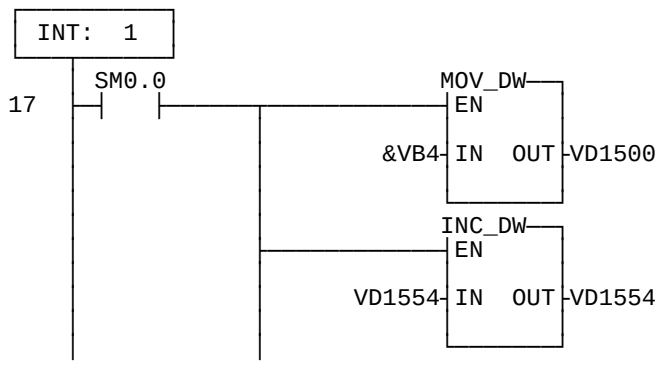



```
INT     0                  // Играть следующую
                           // ноту для канала 0

LD      SM0.0
MOVD    &VB4,VD1500        // Указатель на
                           // таблицу нот

INCD    VD1504             // Указатель на
                           // следующую ноту
                           // для канала 0

MOVB    *VD1504,VB1503     // Получить
                           // следующую нота для
                           // канала 0

LDB=    VB1503,16#FF       // Если это конец
                           // то останов
DTCH    19
CRETI

LD      SM0.0              // Бит всегда "1"
SLW     VW1502,2           // Индекс в таблице
                           // (умноженный на 4)

MOVW    *VD1500,SMW68      // Загрузить
                           // время цикла для
                           // канала 0

INCD    VD1500             // Указатель на
                           // количество импульсов
                           // на единицу времени

INCD    VD1500             // Другой указатель
```

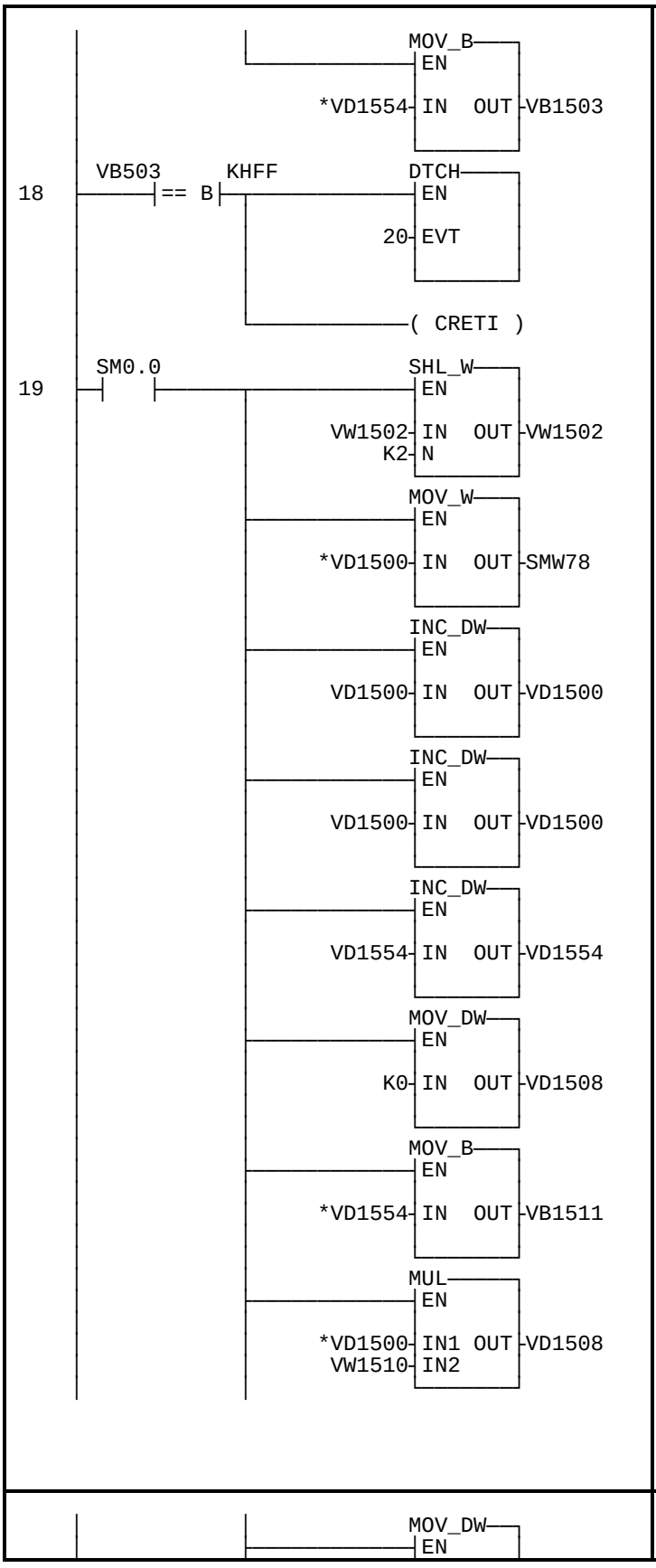



```
INCD    VD1504          // Указатель на
                        // количество единиц
                        // времени

MOVD    0,VD1508  // Очистить V508

MOVB    *VD1504,VB1511  // Получить
                        // количество единиц
                        // времени для этой
                        // ноты

MUL     *VD1500,VD1508  // Вычислить
                        // общее импульсов
                        // для этой ноты

MOVD    VD1508,SMD72  // Загрузить
                        // количество импульсов
                        // для этой ноты

PLS     0     // Запрос следующей
                        // ноты для канала 0

RETI                    // Возврат
```

```
// Прерывание 1 активируется, если заканчивает звучать нота канала 1. Индекс следующей
// ноты и количество время единиц подсчитывается и передается на импульсный выход
// канала 1. Если звучит последняя нота данного канала, прерывание снимается и программа
// заканчивается.
```




```
INT     1               // Играть следующую
                        // нота для канала 1

LD      SM0.0
MOVD    &VB4,VD1500  // Указатель на
                        // таблицу нот

INCD    VD1554          // Указатель на
                        // следующую ноту
                        // для канала 1
```




```
MOVB   *VD1554,VB1503  // Получить
                       // следующую нота для
                       // канала 1

LDB=   VB1503,16#FF    // Если это конец
                       // то останов
DTCH   20

CRETI

LD     SM0.0           // Бит всегда "1"
SLW    VW1502,2        // Индекс в таблице
                       // (умноженный на 4)

MOVW   *VD1500,SMW78   // Загрузить
                       // время цикладля
                       // канала 1

INCD   VD1500          // Указатель на
                       // количество импульсов
                       // за единицу времени

INCD   VD1500          // Другой указатель

INCD   VD1554          // Указатель на
                       // количество единиц
                       // времени

MOVD   0,VD1508        // Очистить V508

MOVB   *VD1554,VB1511  // Получить
                       // количество единиц
                       // времени для этой
                       // ноты

MUL    *VD1500,VD1508  // Вычислить
                       // общее импульсов
                       // для этой ноты
```



```
MOVD   VD1508,SMD82    // Загрузить
```

```
   |                         |
   |                         |  VD1508-|IN   OUT|-SMD82
   |                         |          |________|
   |                         |
   |                         |             PLS---+
   |                         |____________|EN    |
   |                                      |      |
   |                                    1-|Q.X   |
   |                                      |______|
20 |-----------------------------------( RETI  )
   |
```

```
                    // количество импульсов
                    // для этой ноты

PLS        1        // Запрос следующей
                    // ноты для канала 1

RETI                // Возврат
```

Блок Данных Db1 (V Память)

```
// Данный раздел блока данных инициализирует ноты, используемые в
// программе.
// Команды Пересылки над Двойными Словами обрабатывают ноты как
// шестнадцатиричные числа в памяти CPU 214. Первые
// четыре знака кода соответствуют частоте ноты.
// Последние четыре знака соответствуют количеству импульсов, для
// поддержания тона длительностью 0.125 секунды.
// Например, команда
//                    MOVD 16#08E00037,VD4
// означает, что нота 08E0 (шестнадатиричное) загружается в память VW4, а
// количество импульсов 0037 (шестнадатиричное) загружается в память VW6.
//
VD4       16#08E00037
VD8       16#0850003B
VD12      16#07E4003E
VD16      16#07620042
VD20      16#07030046
VD24      16#0691004A
VD28      16#063D004E
VD32      16#05EB0053
VD36      16#058A0058
VD40      16#0542005D
VD44      16#04ED0063
VD48      16#04AE0068
VD52      16#04610070
VD56      16#04280075
VD60      16#03F2007C
VD64      16#03B20084
VD68      16#0382008B
VD72      16#03490095
VD76      16#031E009D
VD80      16#02F600A5
VD84      16#02C500B0
VD88      16#02A100BA
VD92      16#027700C6
VD96      16#025600D1
VD100 16#023100DF
```

```
//
// Т.к. мелодия содержит двойные ноты, то определены два различных канала.
// Следующий раздел блока данных инициализирует мелодию для канала 0.
// Каждая команда Пересылки над Двойными Словами содержит четыре байта.
// Первый байт содержит индекс ноты, второй байт задает количество 0.125-
// секундных интервалов времени. Третий байт - индекс следующей ноты,
// а четвертый байт - количество интервалов для этой ноты.
// Команда              MOVD 16#05040104,VD104
// означает, что первая проигрываемая нота имеет номер 5, а звучание
// первой ноты - четыре временных интервала. Следующая нота имеет номер 1,
// а звучание второй ноты будет опять 4 временных интервала.
//
VD104 16#05040104
VD108 16#03040504
VD112 16#03040104
VD116 16#03040504
VD120 16#03040304
VD124          16#05040604
VD128 16#05040304
VD132 16#05040602
VD136 16#07020804
VD140 16#01040304
VD144 16#05040304
VD148 16#01040304
VD152 16#05040304
VD156 16#03040504
VD160 16#06040504
VD164 16#03040504
VD168 16#06080608
VD172 16#08080A08
VD176 16#08080604
VD180 16#08040A04
VD184 16#0B040D04
VD188 16#0F041204
VD192 16#11040104
VD196 16#03040504
VD200 16#03040104
VD204 16#03040504
VD208 16#03040304
VD212 16#05040604
VD216 16#05040304
VD220 16#05040608
VD224 16#03040504
VD228 16#06080304
VD232 16#05040608
VD236 16#03040504
VD240 16#06020102
VD244 16#0602FFFF
```

```
// Т.к. мелодия содержит двойные ноты, определены два различных канала.
// Следующий раздел блока данных инициализирует мелодию для канала 0.
// Для расшифровки структуры блока, смотри объяснение в предыдущем
// разделе блока данных.
//
VD248 16#0D040D02
VD252 16#0B020A02
VD256 16#0B020D06
VD260 16#0D010F01
VD264 16#0D020B02
VD268 16#0A020B02
VD272 16#0D060D02
VD276 16#0D030B01
VD280 16#0A020802
VD284 16#0D020B02
VD288 16#0A020802
VD292 16#0D020B02
VD296 16#0A020802
VD300 16#0D060D01
VD304 16#0F010D02
VD308 16#0B020A02
VD312 16#0B020D06
VD316 16#0D010F01
VD320 16#0D020B02
VD324 16#0A020B02
VD328 16#0D060D02
VD332 16#0D020B02
VD336 16#0A020802
VD340 16#0D020B02
VD344 16#0A020802
VD348 16#0D020B02
VD352 16#0A020802
VD356 16#06060602
VD360 16#12021002
VD364 16#0F021002
VD368 16#12060602
VD372 16#12021002
VD376 16#0F021002
VD380 16#12060602
VD384 16#12021002
VD388 16#0E021002
VD392 16#12021202
VD396 16#10020E02
VD400 16#0D030F01
VD404 16#0D020F02
VD408 16#0D010F01
VD412 16#0D040D01
VD416 16#0F010D02
VD420 16#0B020A02
VD424 16#0B020D06
VD428 16#0D010F01
```

```
VD432 16#0D020B02
VD436 16#0A020B02
VD440 16#0D060D02
VD444 16#0D020B02
VD448         16#0A020802
VD452         16#0D020B02
VD456         16#0A020802
VD460         16#0D020B02
VD464         16#0A020802
VD468         16#06040F04
VD472         16#0D020B02
VD476 16#0A020802
VD480         16#06040F04
VD484         16#0D020B02
VD488         16#0A020802
VD492 16#06040F04
VD496         16#0D020D02
VD500         16#0F021102
VD504         16#1206FFFF
//
```

Указания по преобразованию

Для того, чтобы преобразовать IEC STL в S7-Micro/DOS STL:

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD- формукаждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Каждый сегмент в этом приере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR, и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные ‘//’, не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов.

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чобы дать пользователямS7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP 7 Micro. Для соблюдения соответствующих технически безопасных предписаний при применениии необходимо предпринять дополнительные меры.
Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применения примеров программ исключается, равно как и при возникновении ущерба личным вещам, персональном ущербе или при намеренных или неосторожных действиях.

Все права защищены. Любая форма копирования и дальнейшего распространения, в том числе и частично, допустимо только с письменного разрешения SIEMENS.

SIMATIC

S7-200 Примеры

Группа	Тема
3	**Подключение S7-200 к системе TI505 через FIM (Field Interface Module - Периферийный Интерфейсный Модуль)**

Краткое описание

В данном примере описывается подключение S7-200 CPU 212 (или CPU 214) к ПЛК SIMATIC TI505. Активная станция - "master" - (TI505) передает данные к пассивной станции - "slave" - (CPU 212) через FIM (Периферийный Интерфейсный Модуль). Для предачи данных используется 4-Словный простой протокол. Таким образом, TI 505 может управлять новыми SIMATIC PLC и другими устройствами, которые поддерживают данный протокол (например, некоторые приводы SE&A).
CPU 212 принимает и передает данные от активной станции по свободнопрограммируемому интерфейсу. Несколько программ обработки прерываний манипулируют данными в пассивной станции.

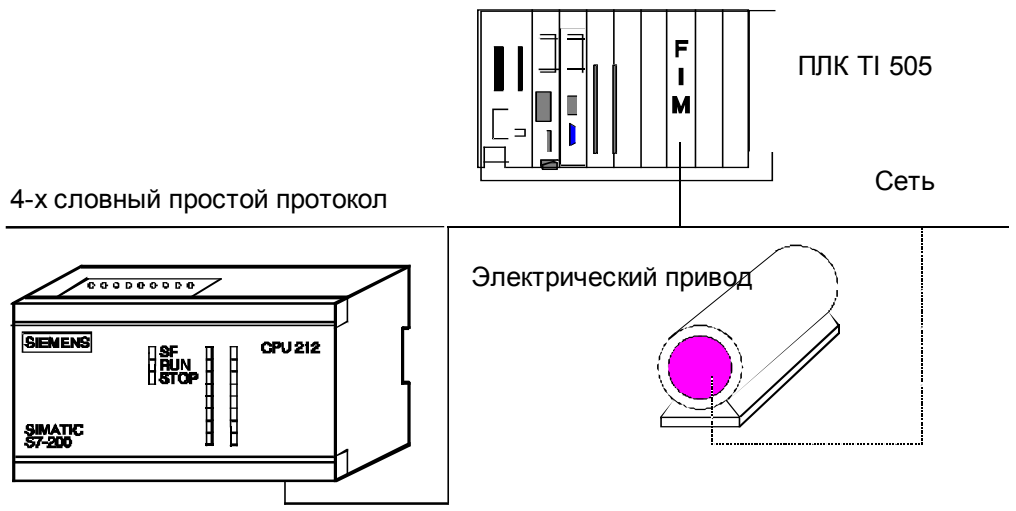


Рисунок 25.1

Аппаратные требования

SIMATIC S7-200 CPU 212 или CPU 214
Кабель RS 485 с двумя разъемами "мама"
SIMATIC TI505 (например, TI 545 или TI 555)
FIM (Периферийный Интерфейсный Модуль) для SIMATIC TI505

Структура программы

MAIN	**Инициализация программы**
SBR 0	**Настройка запросов о приеме FIM.**
INT 0	**Превышение таймера свободной линии.**
INT 1	**Обнаружение свободной линии.**
INT 2	**Прием первого знака.**
INT 3	**Прием адреса.**
INT 4	**Прием данных.**
INT 5	**Прием контрольной суммы.**
INT 6	**Сообщения таймера о тайм-ауте.**
INT 7	**Передача телеграммы и установка таймера задержки.**
INT 8	**Передача завершена или прервана.**

Описание программы

В данном примере описывается подключение S7-200 к ПЛК SIMATIC TI505 через FIM (Периферийный Интерфейсный Модуль). CPU 212 или CPU 214 является пассивной станцией по отношению к SIMATIC TI505.

Для настройки FIM переключатели следует установить в следующие положения:

Переключатель 1	№ FIM
Переключатель 2	Тип протокола: 4 словный простой протокол: 0
Переключатель 3	Число микропереключателей
Микропереключатель 1	закрыт
Микропереключатель 2	закрыт
Микропереключатель 3	закрыт
Микропереключатель 4	закрыт
Микропереключатель 5	закрыт
Микропереключатель 6	открыт
Микропереключатель 7	закрыт
Микропереключатель 8	закрыт

4 словный простой протокол имеет следующую структуру

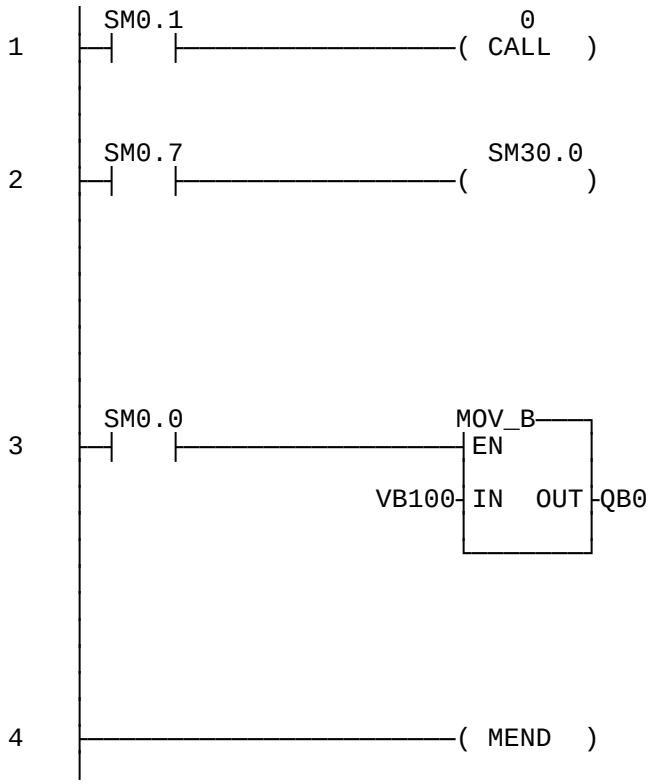
02	Начальная комбинация	1 байт
AA	Адрес	1 байт
MM	Телеграмма	
...	...	8 байт
MM	Телеграмма	
CC	Контрольная сумма	1 байт

Размер программы - 189 слов.
Более подробную информацию о программах обработки прерываний Вы найдете в Главе 6.2 “Команды прерываний” *в Руководстве по Программированию S7-200*.

LAD (S7-MicroDOS)	STL (IEC)

Главная программа

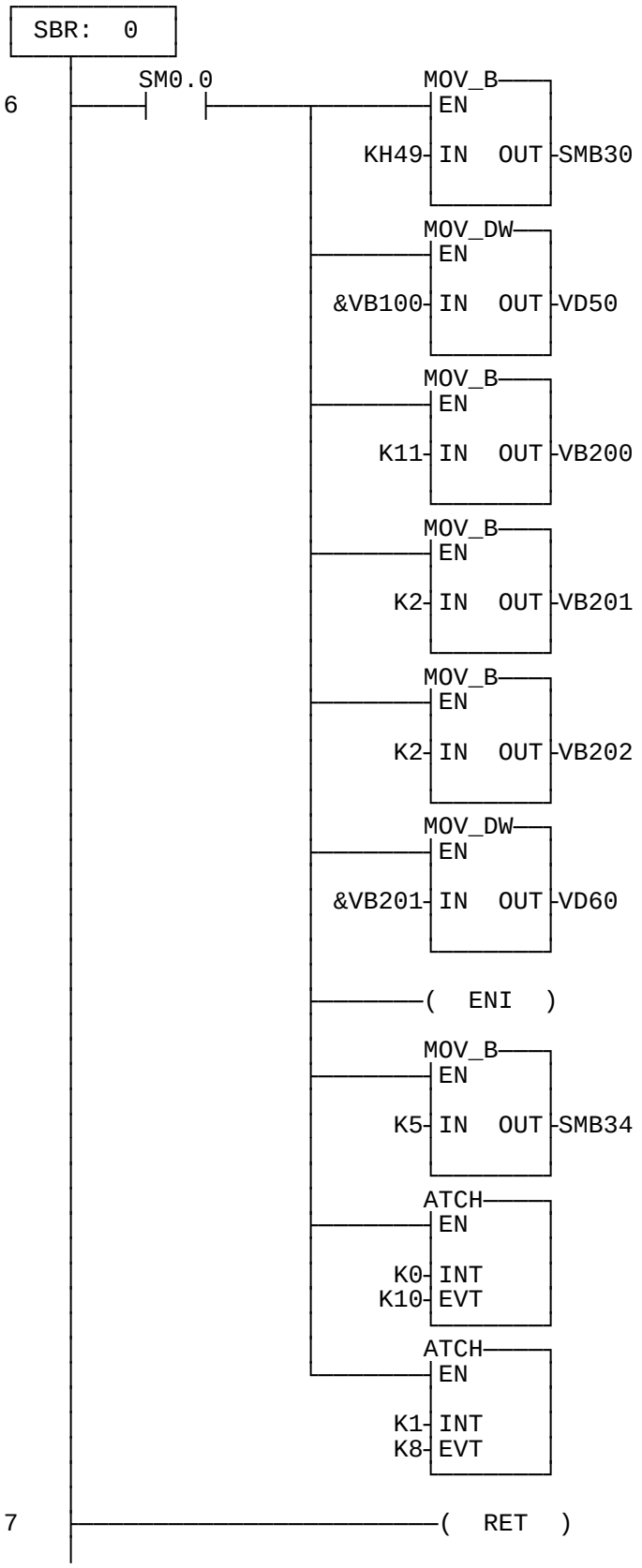
```
// TITLE=FIM

// Главная программа вызывает программу настройки и устанавливает режим.
// Если переключатель режима SIMATIC CPU 212 находится в положении TERM,
// то используется протокол PPI. Если же этот прерключатель - в положении RUN,
// то устанавливается режим свободнопрограммируемого интерфейса.
// Певый байт входов (IB 0) копируется в буфер передачи  V памяти, а
// первый байт из буфера приеме V памяти корируется в байт выходов (QB 0).
```




```
LD      SM0.1   // Бит первого цикла
CALL    0       // Вызов программы
                // настройки 0

LD      SM0.7   // Если режим term,
                // то установить
                // протокол PPI
=       SM30.0
                // Если режим run,
                // то установить
                // протокол
                // свободнопрограмми
                // руемого интерфейса

LD      SM0.0   // Бит всегда в "1"
MOVB    VB100, QB0
                // Скопировать
                // первый байт
                // первого принятого
                // слова в выходы

                // Скопировать входы
                // в первй байт
                // первого слова,
                // переданного в
                // INT 5
MEND
```

Подпрограммы

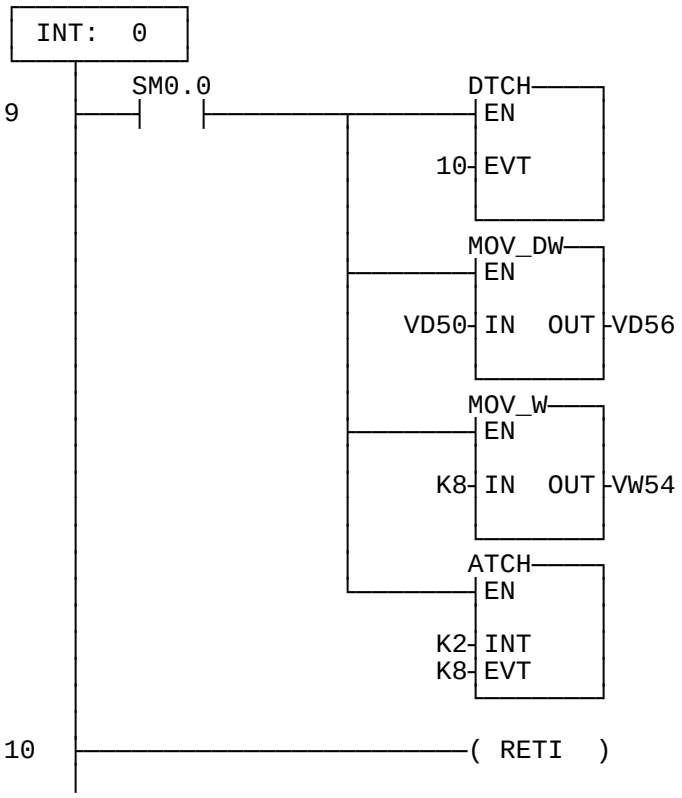
// Подпрограмма 0 устанавливает запросы о приеме FIM; она устанавливает режим
// свободнопрограммируемого интерфейса (скорость передачи 9600, 8 бит/знак, четность)




```
SBR     0            // Установка
                     // запросов о приеме
                     // FIM
LD      SM0.0        // Бит всегда "1"
MOVB    16#49, SMB30
                     // 9600 Бод,
                     // 8 бит/знак,
                     // с четностью
MOVD    &VB100, VD50
                     // Загрузить
                     // указатель приема
MOVB    11, VB200    // Загрузить
                     // количество
                     // переданных знаков
MOVB    2, VB201     // Загрузить знак STX
MOVB    2, VB202     // Загрузить адрес
MOVD    &VB201, VD60
                     // Загрузить
                     // указатель на
                     // передаваемые
                     // данные
ENI                  // Разрешить
                     // прерывания
MOVB    5, SMB34     // Установить тайм-
                     // аут свободной
                     // линии = 5 ms
ATCH    0, 10        // Запустить таймер
                     // свободной линии
ATCH    1, 8         // Присоединить
                     // программу
                     // прерываний 1 для
                     // обнаружения
                     // свободной линии
RET                  // Прервать
                     // подпрограмму
```

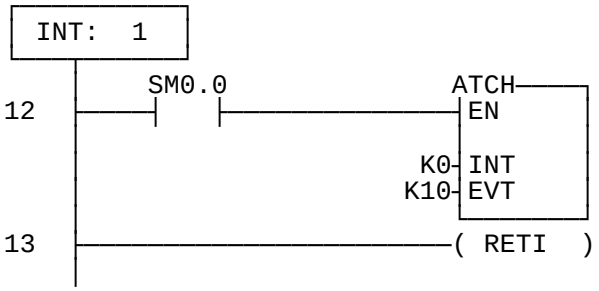
Программы обработки прерываний

// **********
// Программа обработки прерывания 0 активизируется, если превышено время свободной
// линии. Разрешается программа STX (старт). Время свободной линии - это время между
// двумя телеграммами.



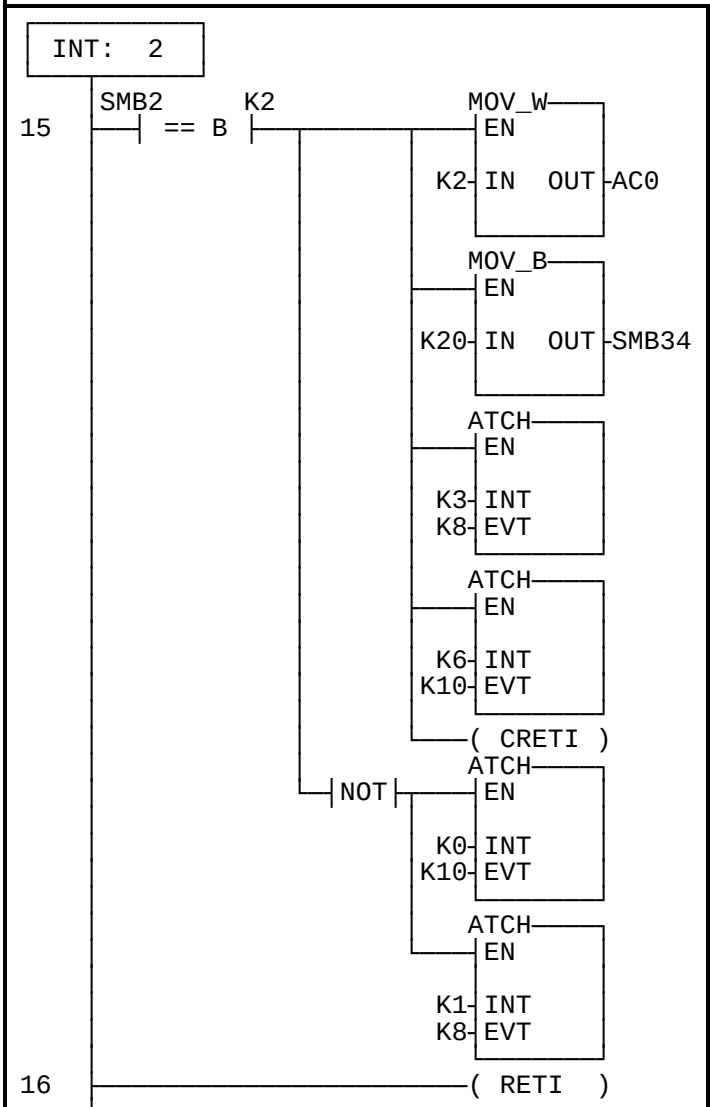
INT	0	// Превышено время // свободной линии
LD	SM0.0	// Бит всегда в "1"
DTCH	10	// Остановить таймер // свободной линии
MOVD	VD50, VD56	// Загрузить V56 с // указателем на // буфер приема
MOVW	8, VW54	// Установить прием 8 // знаков данных
ATCH	2, 8	// Разрешить // программу приема // STX
RETI		

// **********
// Программа обработки прерываний 1 контролирует свободна ли линия,
// если произошло прерывание.



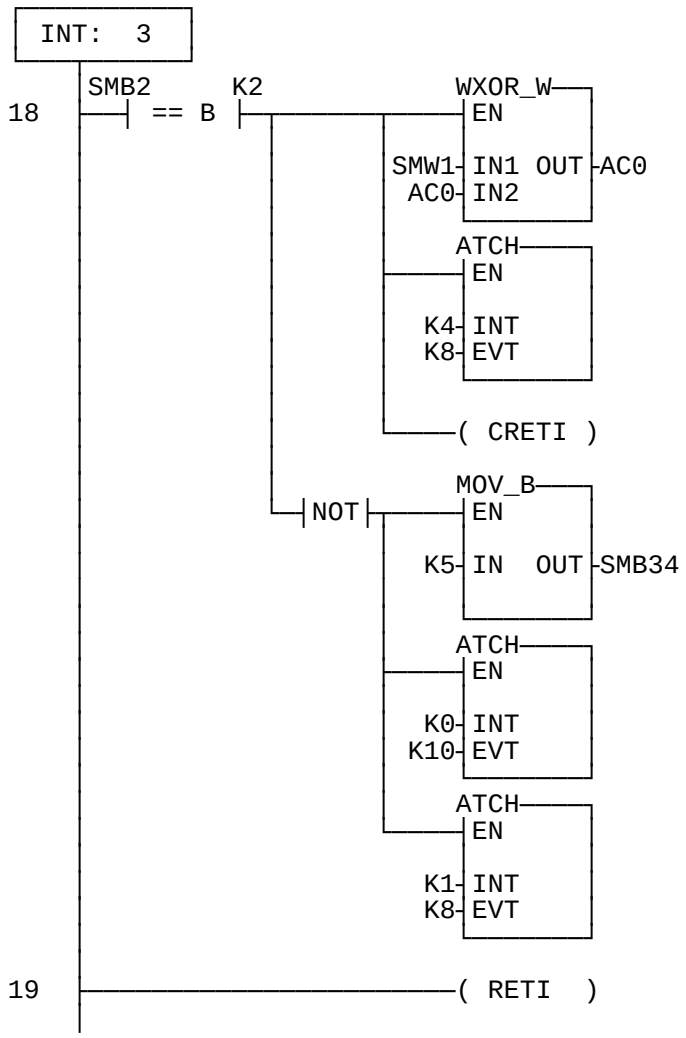
INT	1	// Ожидание // свободной линии
LD	SM0.0	// Бит всегда в "1"
ATCH	0, 10	// Перезапустить // таймер свободной // линии
RETI		

```
// ****************************************************************
// Программа обработки прерываний 2 активизируется, если принят знак STX (стартовый).
```



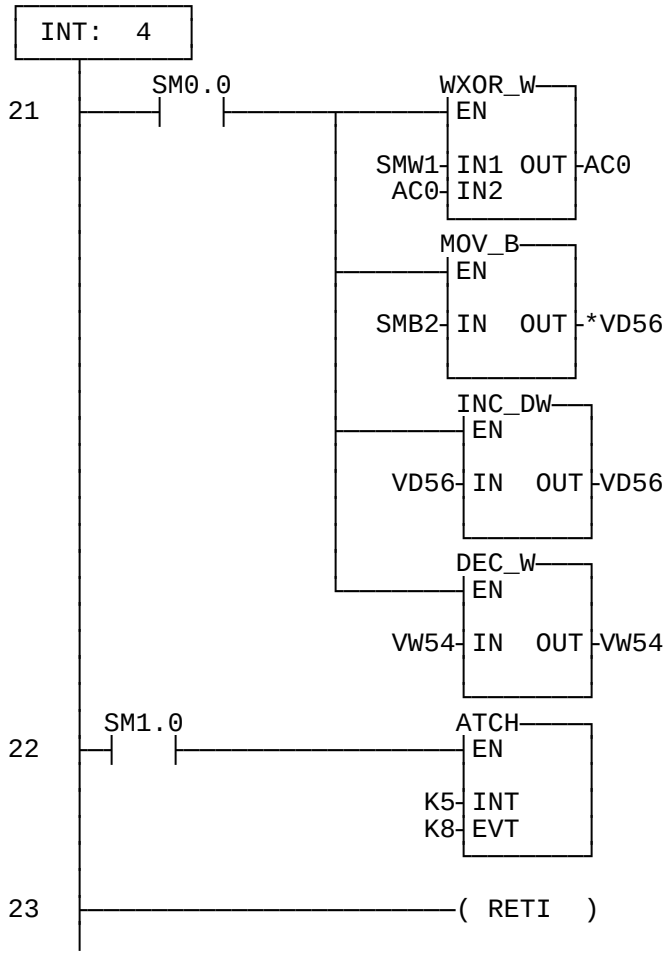

```
INT      2          // Прием знака STX
LDB=     SMB2, 2
                    // Это знак STX?
MOVW     2, AC0     // Запуск контрольной
                    // суммы

MOVB     20, SMB34
                    // Установить таймер
                    // тайм-аута = 20ms

ATCH     3, 8       // Присоединить
                    // программу
                    // прерываний 3 для
                    // получения адреса

ATCH     6, 10      //   Присоединить
                    // программу
                    // прерываний 6 для
                    // тайм-аута
                    // телеграммы

CRETI
NOT                 // Взять логическое НЕ
                    // от значения
                    // вершины стека
ATCH     0, 10      // Перезапустить
                    // таймер свободной
                    // линии

ATCH     1, 8       // Присоединить
                    // программу
                    // прерываний 1 для
                    // обнаружения
                    // свободной линии
RETI
```

```
// ****************************************************************
// Программа обработки прерываний 3 активизируется при получении знака адреса.
```



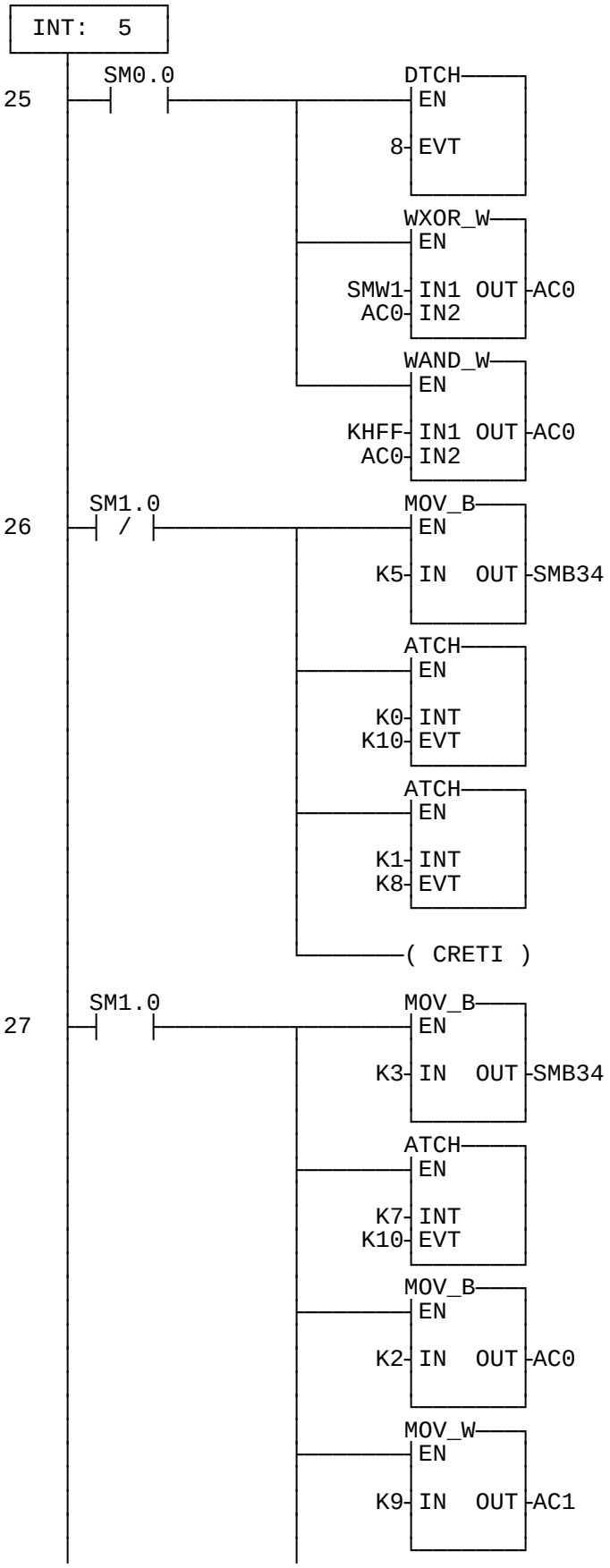

```
INT      3          // Прием адреса
LDB=     SMB2, 2
                    // Телеграмма для этой
                    // станции?
XORW     SMW1, AC0
                    // Вычислить
                    // контрольную сумму

ATCH     4, 8       // Присоединить
                    // программу
                    // прерывания 4 для
                    // приема данных

CRETI

NOT                 // Взять логическое НЕ
                    // от значения
                    // вершины стека
MOVB     5, SMB34
                    // Установить тайм-
                    // аут свободной
                    // линии = 5 ms

ATCH     0, 10      // Запуск таймера
                    // свободной линии

ATCH     1, 8       // Присоединить
                    // программу
                    // прерывания 1 для
                    // обнаружения
                    // свободной линии

RETI
```

// Программа обработки прерываний 4 активизируется при получении данных.

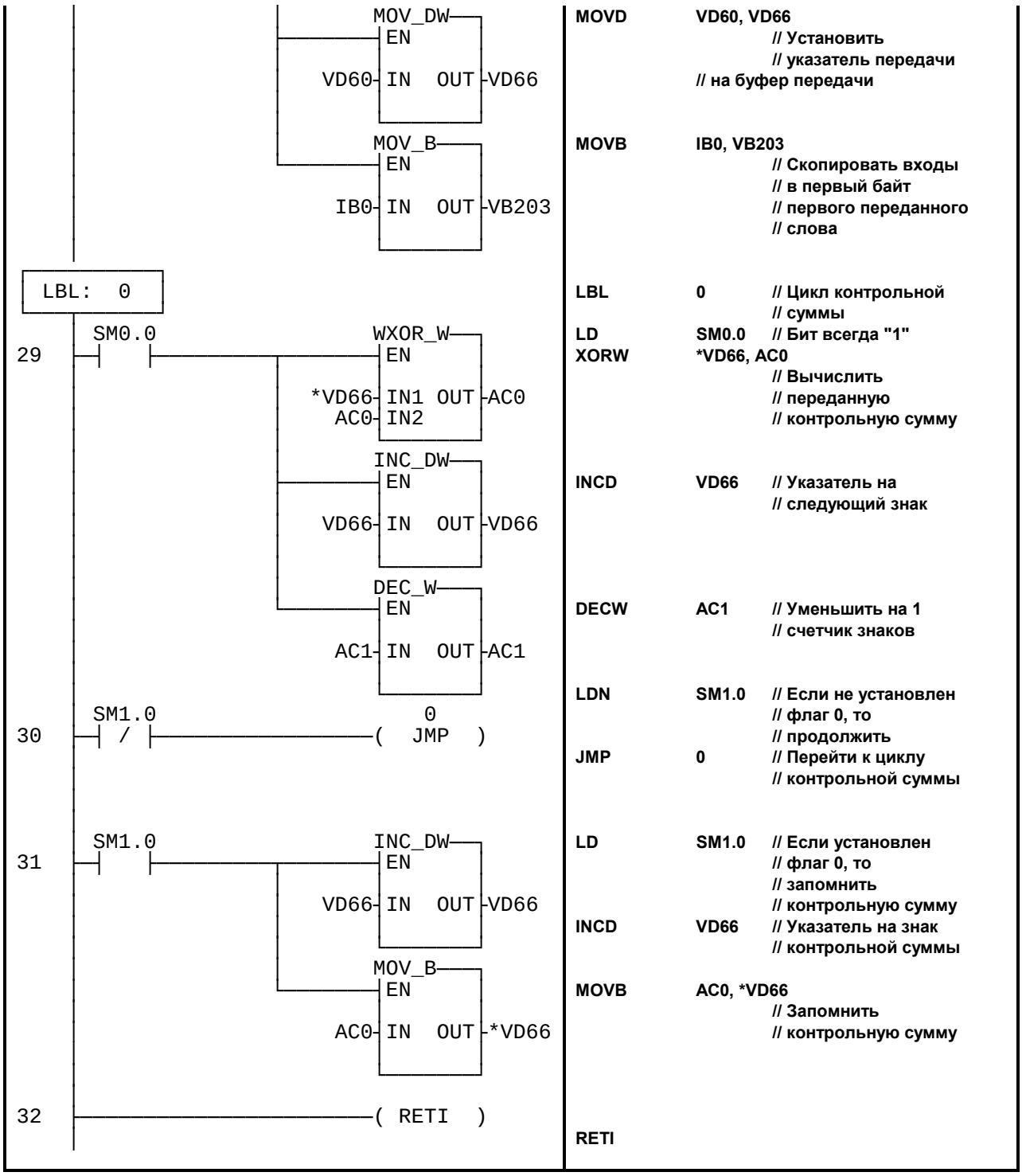


INT	**4**	**// Прием данных**
LD	**SM0.0**	**// Бит всегда "1"**
XORW	**SMW1, AC0**	
		// Вычислить
		// контрольную сумму
MOVB	**SMB2, *VD56**	
		// Сохранить байт
		// данных
INCD	**VD56**	**// Увеличить на 1**
		// указатель на
		// данные
DECW	**VW54**	**// Уменьшить на 1**
		// счетчик знаков для
		// каждого принятого
		// знака
LD	**SM1.0**	**// Если приняты все**
		// данные, то получить
		// контрольную сумму
ATCH	**5, 8**	**// Присоединить**
		// программу
		// прерывания 5 для
		// приема
		// контрольной суммы
RETI		

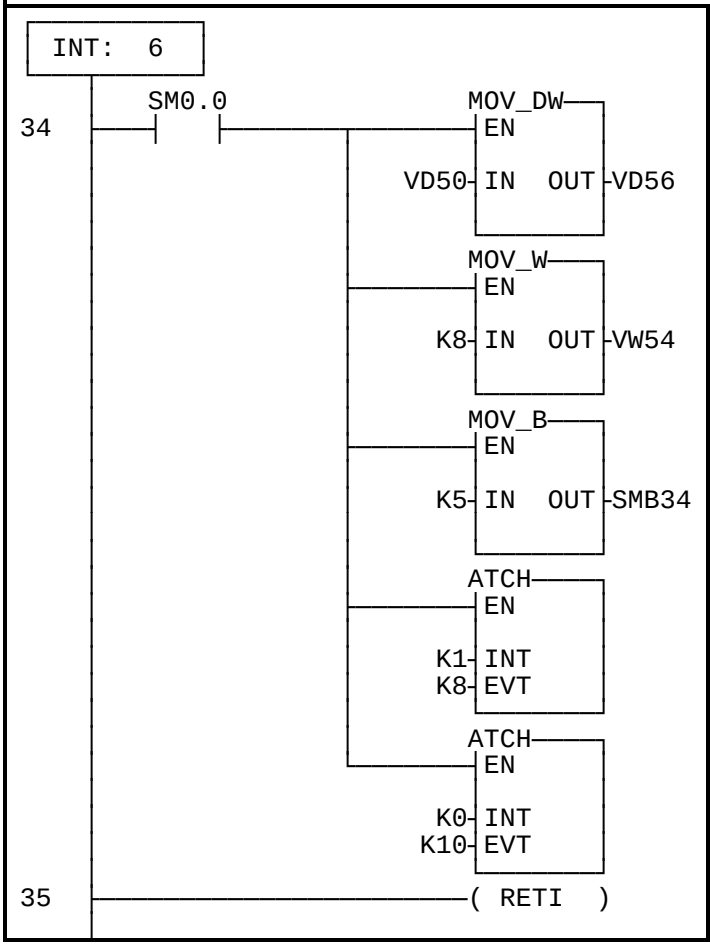
// **********
// Программа обработки прерываний 5 активизируется при получении контрольной суммы.




```
INT      5        // Прием контрольной
                  // суммы
LD       SM0.0    // Бит всегда "1"
DTCH     8        // Прервать прием

XORW     SMW1, AC0
                  // Вычислить
                  // контрольную сумму

ANDW     16#FF, AC0
                  // Маскировать
                  // младший байт
         // контрольной суммы

LDN      SM1.0    // Если не установлен
                  // флаг 0, то отбросить
         // телеграмму
MOVB     5, SMB34
                  // Установить тайм-
                  // аут свободной
                  // линии = 5 ms

ATCH     0, 10    // Запуск таймера
                  // свободной линии

ATCH     1, 8     // Присоединить
                  // программу
                  // прерывания 1 для
                  // обнаружения
                  // свободной линии

CRETI

LD       SM1.0    // Если установлен
                  // флаг 0, то
                  // контрольная сумма
                  // верная
MOVB     3, SMB34
                  // Установить время
                  // задержки передачи
                  // = 3ms

ATCH     7, 10    // Присоединить
                  // программу
                  // прерывания 7 для
                  // передачи задержки
                  // тайм-аута

MOVB     2, AC0   // Инициализация AC0

MOVW     9, AC1   // № знака в
                  // вычислении
                  // контрольной суммы
```

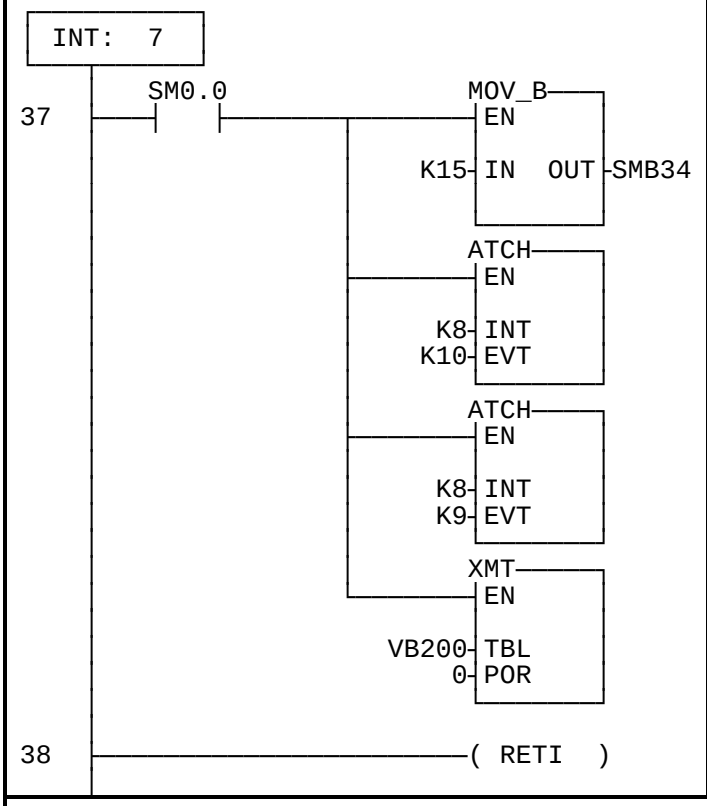

MOV_DW
EN
VD60 IN OUT VD66
MOV_B
EN
IB0 IN OUT VB203
LBL: 0
SM0.0
29
WXOR_W
EN
*VD66 IN1 OUT AC0
AC0 IN2
INC_DW
EN
VD66 IN OUT VD66
DEC_W
EN
AC1 IN OUT AC1
SM1.0
30
0
JMP
SM1.0
31
INC_DW
EN
VD66 IN OUT VD66
MOV_B
EN
AC0 IN OUT *VD66
32
RETI
MOVD VD60, VD66
// Установить
// указатель передачи
// на буфер передачи
MOVB IB0, VB203
// Скопировать входы
// в первый байт
// первого переданного
// слова
LBL 0 // Цикл контрольной
// суммы
LD SM0.0 // Бит всегда "1"
XORW *VD66, AC0
// Вычислить
// переданную
// контрольную сумму
INCD VD66 // Указатель на
// следующий знак
DECW AC1 // Уменьшить на 1
// счетчик знаков
LDN SM1.0 // Если не установлен
// флаг 0, то
// продолжить
JMP 0 // Перейти к циклу
// контрольной суммы
LD SM1.0 // Если установлен
// флаг 0, то
// запомнить
// контрольную сумму
INCD VD66 // Указатель на знак
// контрольной суммы
MOVB AC0, *VD66
// Запомнить
// контрольную сумму
RETI

```
// ****************************************************************
// Программа обработки прерываний 6 активизируется,
// если превышено установленное время телеграммы.
```



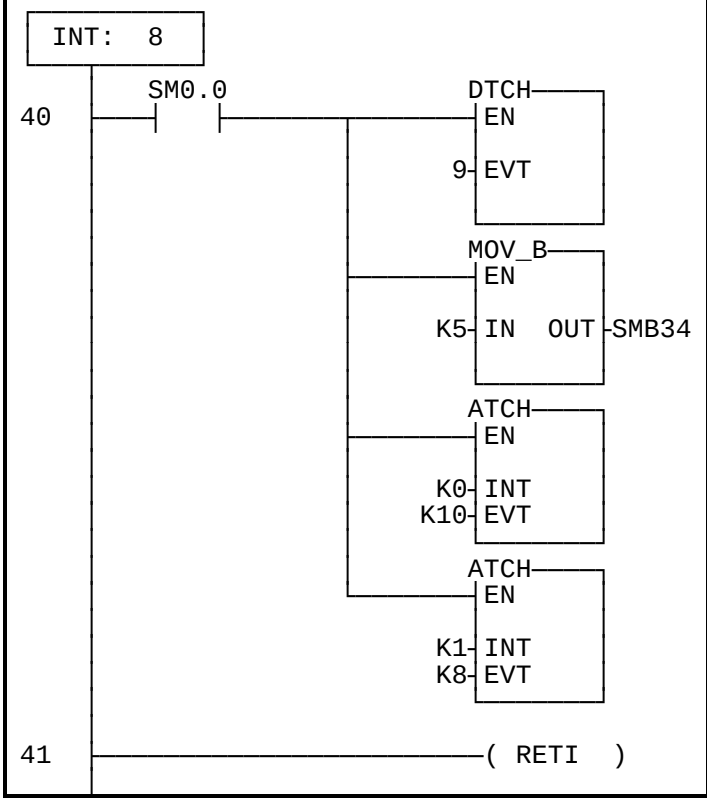
INT	**6**	**// Таймер тайм-аута // телеграммы**
LD	**SM0.0**	**// Бит всегда "1"**
MOVD	**VD50, VD56**	**// Загрузить V56 с // указателем на // буфер приема**
MOVW	**8, VW54**	**// Установить прием 8 // знаков данных**
MOVB	**5, SMB34**	**// Установить таймер // свободной линии // = 5ms**
ATCH	**1, 8**	**// Присоединить // программу // прерывания для // обнаружения // свободной линии**
ATCH	**0, 10**	**// Запуск таймера // свободной линии**
RETI		

// **********
// Программа обработки прерываний 7 устанавливает время задержки передачи и
// передает телеграмму.



INT	7	// Передача таймера // задержки
LD	SM0.0	// Бит всегда "1"
MOVB	15, SMB34	// Установить тайм-аут // передачи = 15ms
ATCH	8, 10	// Запуск тайм-аута // передаваемой // телеграммы
ATCH	8, 9	// Разрешить // прерывания при // передачи // телеграммы
XMT	VB200, 0	// Передать // телеграмму
RETI		

// **********
// Программа обработки прерываний 8 переводит ПЛК в первоначальное сотсояние после
// передачи телеграммы для приема новой телеграммы.



INT	8	// Передача телеграммы
LD	SM0.0	// Бит всегда "1"
DTCH	9	// Запретить // прерывания // передачи
MOVB	5, SMB34	// Установить тайм- // аут свободной // линии = 5 ms
ATCH	0, 10	// Запуск таймера // свободной линии
ATCH	1, 8	// Присоединить // программу // прерывания 1 для // обнаружения // свободной линии
RETI		

Copyright ©1995 by SIEMENS

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//", не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все прав защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC

S7-200 Примеры

Группа	Тема
3	**Использование Hayes Модема с S7-200**

Краткое описание

В данном примере описывается использование стандартного Hayes совместимого модема с SIMATIC S7-200 CPU 212 (или CPU 214) в режиме свободнопрограммируемого интерфейса. Модем подсоединенный к S7-200 вызывает другой модем и передает строку сообщений. Т.к. Hayes модемы поддерживают использование только семи бит данных с четностью, то режим PPI не может быть использован. Поэтому информация передается используя свободнопрограммируемый интерфейс.

S7-200 м.б. также использован при передачи данных как пассивное (slave) устройство.

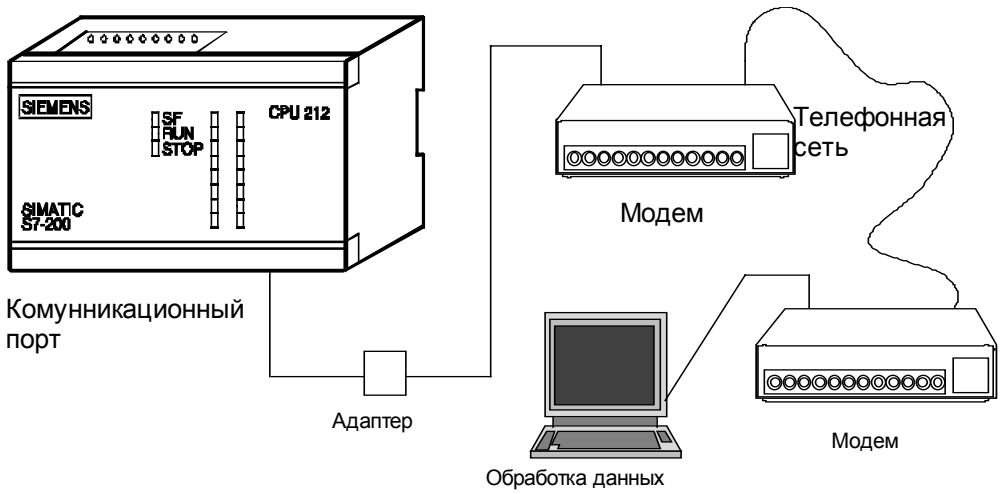


Рисунок 26.1

Аппаратные требования

Для функционирования данной программы Вам необходимо:

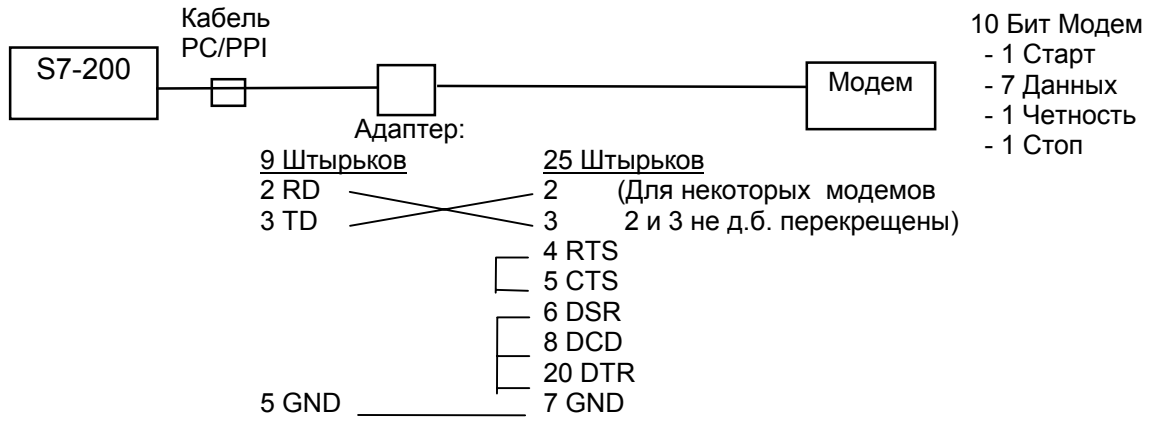
1 SIMATIC S7-200 CPU 212 или CPU 214

1 Кабель PC/PPI

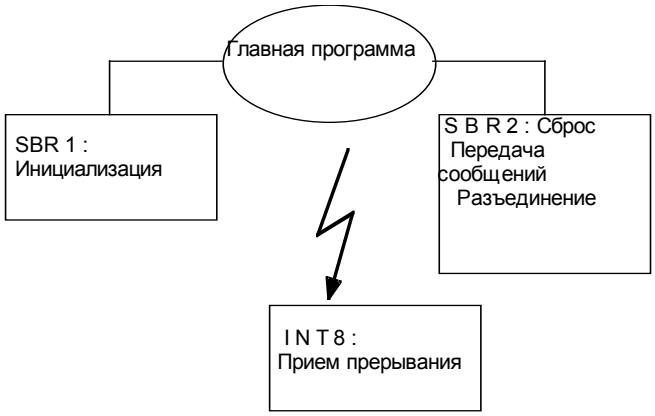
1 соответствующий адаптер (в зависимости от выхода модема, обычно это 9 штырьковый "мама" 25 штырьковый "мама" и и нуль-модемный кабель с перекрестием передающей и приемной линий)

1 Hayes модем (модем должен иметь скорость передачи 9600 Бод, но так же м.б. использован и более медленный модем.)

Интерфейс адаптера



Структура программы



Описание программы

Этот пример инициализирует вызов модема главной ЭВМ.

S7-200 CPU 212 или CPU 214 передает информацию к Hayes модему, если у него есть сообщение для передачи. Данные содержат вызываемый телефонный номер и передаваемое сообщение. Сообщение храниться в памяти ПЛК.

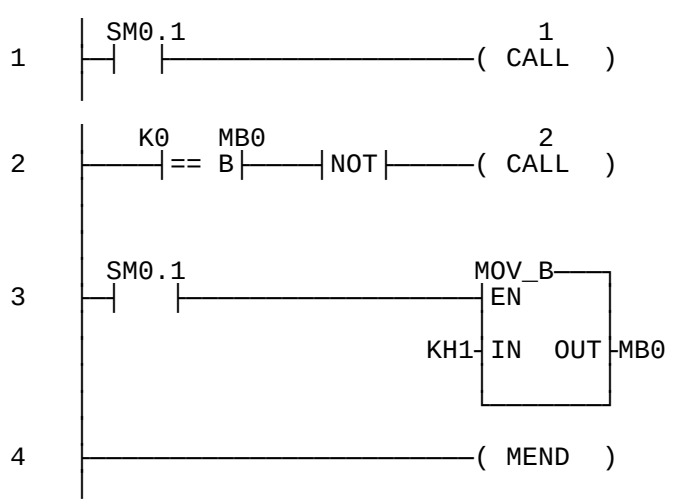
Размер программы составляет 190 слов.

Более подробную информацию о передающей функции XMT Вы найдете в Главе 6.4 “Команды передачи” в *Руководстве по Программированию S7-200*.

LAD (S7-MicroDOS)	STL (IEC)

Главная программа

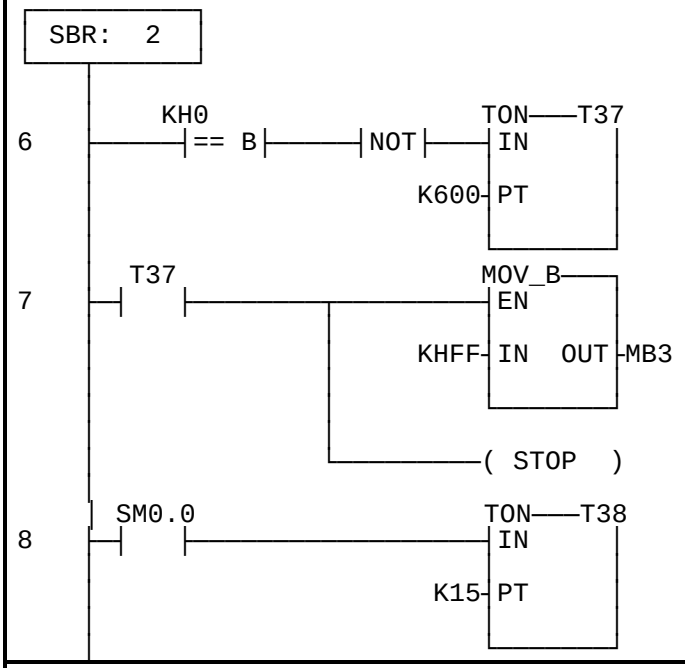
// Главная программа




```
LD      SM0.1   // Если первый цикл, то
CALL    1       // инициализация
                // сообщений

LDB=    0,MB0   // Если есть сообщение NOT
                // для передачи, то
CALL    2       // Вызывается
                // подпрограмма
                // передачи

LD      SM0.1   // Если первый цикл, то

MEND
```

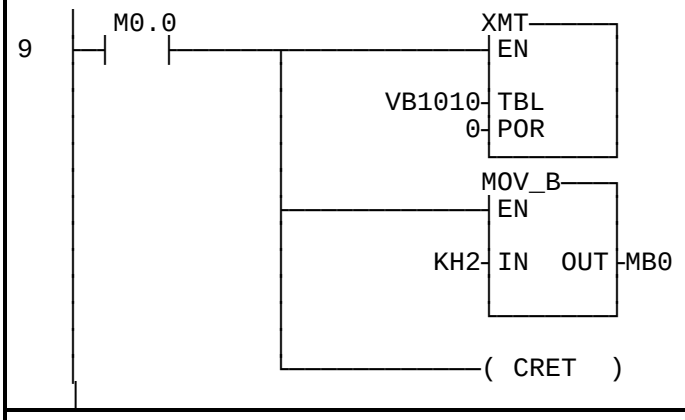
Подпрограммы

// **********
// Подпрограмма 2 набирает, передает сообщение и разъединяет.



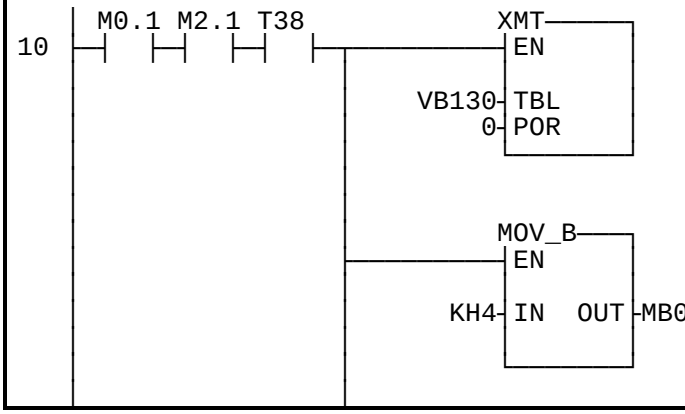

```
SBR     2
LDB=    16#0,MB0
                // Если передача
                // активна,то
NOT
TON     T37,600
                // Разрешить таймер
                // тайм-аута
LD      T37     // Если таймер тайм-аута
                // превышен, то
MOVB    16#FF,MB3
                // Выдать ошибку
STOP            // Прервать остальное
LD      SM0.0   // Запустить общий
TON     T38,15  // 1.5 сек. таймер
```

// Состояние 1 - Вызов модема.




```
LD      M0.0    // Если состояние вызова
XMT     VB1010,0
                // то сообщить модему,
                // чтобы он набирал
MOVB    16#2,MB0
                // Переход в состояние
                // ожидания
CRET            // Прервать
```

// Состояние 2 - Ожидание соединения и передача сообщения




```
LD      M0.1    // Если ожидание
                // состояния соединения
A       M2.1    // И мы получили ответ
                // о соединении
A       T38     // И таймер ожидания
                // превышен, то
XMT     VB130,0
                // Послать телеграмму
MOVB    16#4,MB0
                // Перейти в состояние
                // ожидания
```

```
                                  ( CRET   )
```

CRET	// Конец если

// Состояние 3 - Ожидание передачи сообщения

```
11   M0.2   SM4.5          MOV_B
 ─┤    ├──┤    ├──────────┤EN
                      KH8-IN   OUT-MB0
```

LD	M0.2	// Если ожидаем
		// конца передачи
A	SM4.5	// И передача
		// закончена, то
MOVB	16#8,MB0	
		// Перейти в состояние
		// отбоя

// Состояния 4 - 8 - Повесить телефонную трубку
// Включает пять состояний
// 4) Ожидание 1.5 сек.
// 5) Передача ESCAPE последовательности (+++)
// 6) Ожидание 1.5 сек.
// 7) Передача команды
// 8) Ожидание перехода в автономный режим

// Состояние 4 - Первая пауза при отбое

```
12   M0.3                  MOV_B
 ─┤    ├──────────────────┤EN
                     KH10-IN   OUT-MB0

                      T38       K1
                   ──────────(   R   )

                          MOV_W
                   ──────┤EN
                     K0-IN   OUT-VW1008

                   ──────────( CRET  )
```

LD	M0.3	// Если первая пауза, то
MOVB	16#10,MB0	
		// Показать состояние
		// передачи
R	T38,1	// Сбросить таймер
		// ожидания
MOVW	0,VW1008	
		// Очистить счетчик "+"
CRET		// Пропустить остальное

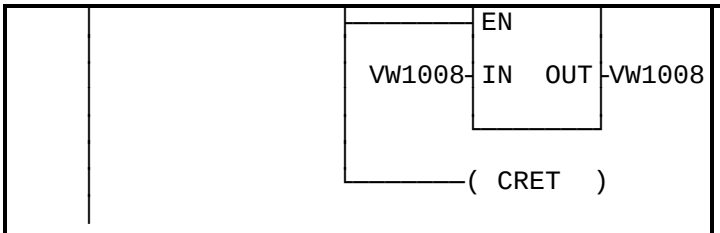
// Состояние 5 - Передача ESCAPE последовательности. Д.б. передано как последовательность
// 3 телеграмм. Вариант передачи одной телеграммы с 3 знаками не подходит.

```
13   M0.4   K3 VW1008      MOV_B
 ─┤    ├──┤== W ├──────┤EN
                     KH20-IN   OUT-MB0

                      T38       K1
                   ──────(   R   )
                   ──────( CRET  )
```

LD	M0.4	// Если состояние
		// передачи escape
		// последовательности
AW=	3,VW1008	
		// И счетчик = 3, то
MOVB	16#20,MB0	
		// Перейти во второе
		// состояние ожидания
R	T38,1	// Сброс таймера
		// ожидания
CRET		// Возврат

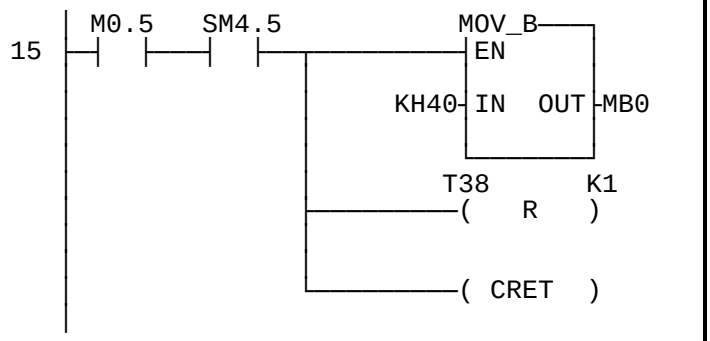
```
14   M0.4    T38           XMT
 ─┤    ├──┤    ├──────────┤EN
                     VB1000-TBL
                          0-POR

                          INC_W
```

LD	M0.4	// Если состояние
A	T38	// передачи и превышен
		// таймер ожидания, то
XMT	VB1000,0	
		// Передать escape
		// последовательность



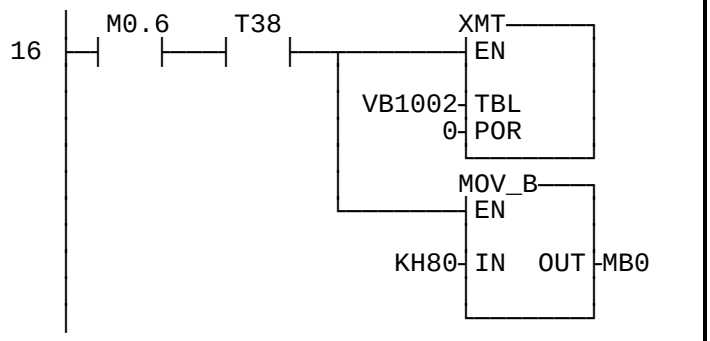

```
INCW    VW1008
                // Увеличить счетчик "+"

CRET            // Возврат
```

// Состояние 6 - Вторая пауза отбоя



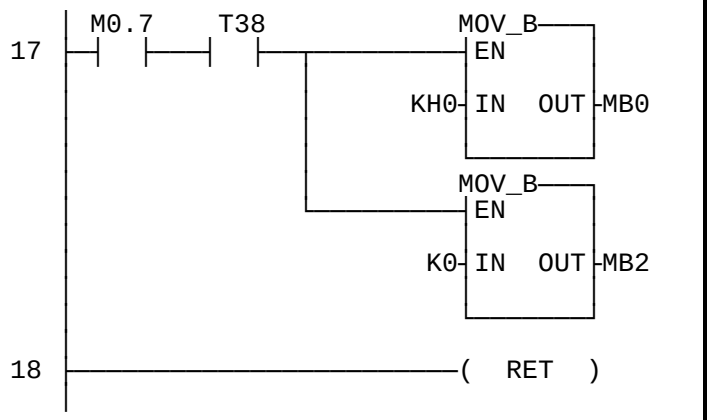

```
LD      M0.5    // Если второе состояние AND
                // ожидания и
A       SM4.5   // Передача завершена
MOVB    16#40,MB0
                // Показать состояние
                // отбоя
R       T38,1   // Сбросить таймер
                // ожидания
CRET            // Возврат
```

// Состояние 7 - Передача команды отбоя



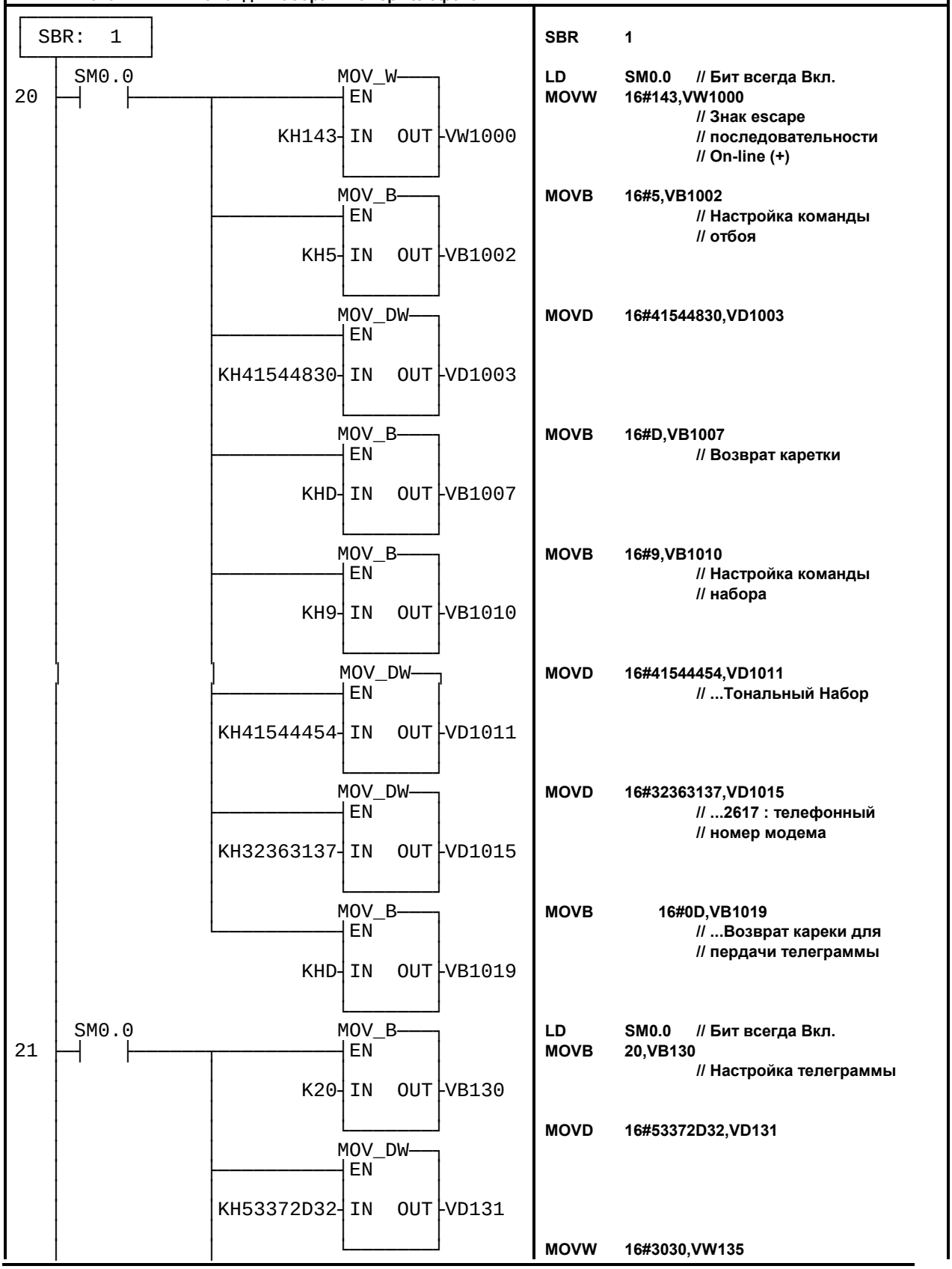

```
LD      M0.6    // Если состояние отбоя
A       T38     // И таймер ожидания
                // превышен, то
XMT     VB1002,0
                // Передать команду
                // отбоя
MOVB    16#80,MB0
                // Перейти в третье
                // состояние ожидания
```

// Состояние 8 - Ожидание ACK от модема




```
LD      M0.7    // Если третье состояние AND
                // ожидания и превышен
A       T38     // таймер ожидания, то
MOVB    16#0,MB0
                // Разорвать передачу
MOVB    0,MB2   // Очистить ошибку
                // ответа
RET             // Возврат
```

```
// ****************************************************************
// Попрограмма 1 выполняет инициализацию телеграмм и режима
// свободнопрограммируемого интерфейса.
// Области памяти :
//      V1000 - V1001      Знаки escape последовательности для режима On-line
//      V1002 - V1007      Команды отбоя
//      V1008 - V1009      Счетчик знаков Escape последовательности
//  V1010 - V1???   Команды набора и номер телефона
```




```
SBR     1
LD      SM0.0     // Бит всегда Вкл.
MOVW    16#143,VW1000
                  // Знак escape
                  // последовательности
                  // On-line (+)
MOVB    16#5,VB1002
                  // Настройка команды
                  // отбоя
MOVD    16#41544830,VD1003
MOVB    16#D,VB1007
                  // Возврат каретки
MOVB    16#9,VB1010
                  // Настройка команды
                  // набора
MOVD    16#41544454,VD1011
                  // ...Тональный Набор
MOVD    16#32363137,VD1015
                  // ...2617 : телефонный
                  // номер модема
MOVB        16#0D,VB1019
                  // ...Возврат кареки для
                  // пердачи телеграммы
LD      SM0.0     // Бит всегда Вкл.
MOVB    20,VB130
                  // Настройка телеграммы
MOVD    16#53372D32,VD131
MOVW    16#3030,VW135
```

```
MOV_W
EN
KH3030- IN   OUT -VW135
```

// Передаваемая
// телеграмма

```
MOV_W
EN
KH2020- IN   OUT -VW137
```

MOVW 16#2020,VW137

```
MOV_DW
EN
KH50484F4E- IN   OUT -VD139
```

MOVD 16#50484F4E,VD139
// номер S7-200

```
MOV_B
EN
KH45- IN   OUT -VB143
```

MOVB 16#45,VB143

```
MOV_B
EN
KH20- IN   OUT -VB144
```

MOVB 16#20,VB144

```
MOV_DW
EN
KH484F4D45- IN   OUT -VD145
```

MOVD 16#484F4D45,VD145

```
MOV_W
EN
KHD0A- IN   OUT -VW149
```

MOVW 16#D0A,VW149

```
MOV_B
EN
KH9- IN   OUT -SMB30
```

MOVB 16#9,SMB30
// Установить
// свободнопрограммир.
// режим - 9600,8,N

```
MOV_B
EN
K0- IN   OUT -MB0
```

MOVB 0,MB0 // Показать нулевое
// состояние

```
MOV_B
EN
K0- IN   OUT -MB2
```

MOVB 0,MB2 // Очистить ответ об
// ошибке

```
ATCH
EN
K8- INT
K8- EVT
```

ATCH 8,8 // Назначить прерывание
// приема

```
( ENI )
```

ENI; // Разрешить прерыван.
// пользователя

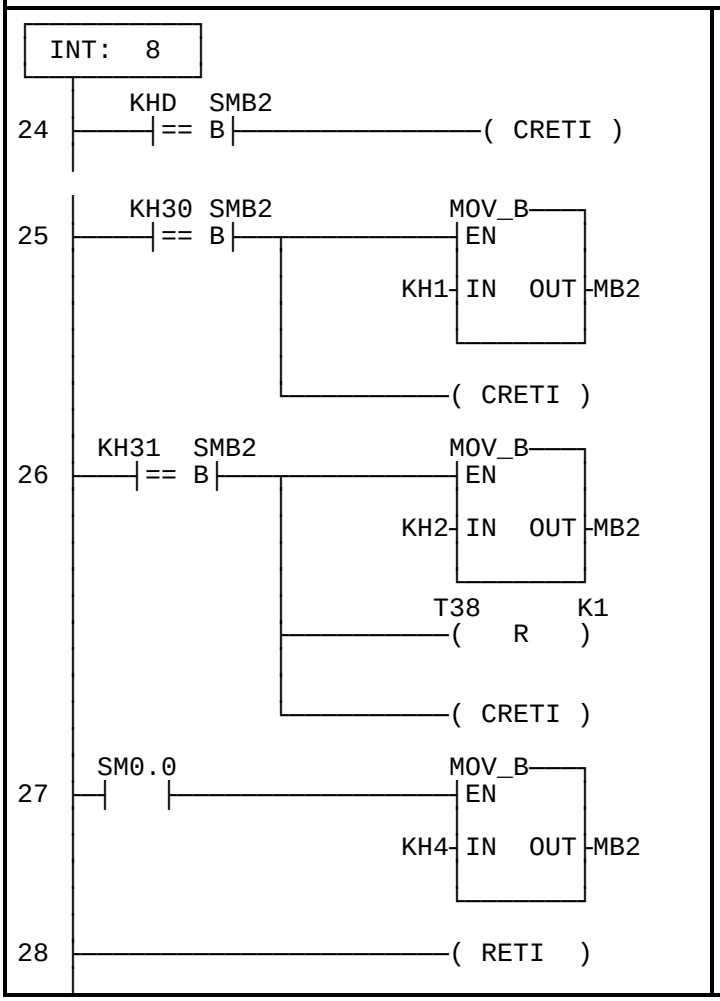
```
T37 K1
( R )
```

R T37,1 // Сброс таймера 33

22	(RET)	RET

Программы прерываний

```
// ******************************************************************
// Эта программа обработки прерывания выделяет ответы от модема и обрабатывает их.
// Коды:
//      0       OK
//      1       соединение
//      2       звонок
//      3       нет несущей
//      4       ошибка
//      5       соединение 1200
//      6       нет тона набора
//      7       занято
//      8       нет ответа
//      10      соединение 2400
//      11      соединение 4800
//      12      соединение 9600

// Если выбран параметр X0, то будут только коды 0, 1, 2, 3 и 4.
// Коды 0 и 1 - положительные; коды 2, 3 и 4 - ошибки.

// Принимаемый кадр состоит из кода плюс CR.
```




```
INT          8

LDB=     16#D,SMB2
                 // Если  CR, то
CRETI            // аварийное завершение

LDB=     16#30,SMB2
                 // Если "0" (OK), то
MOVB     16#1,MB2
                 // Установить состояние
                 // OK

CRETI            // Возврат

LDB=     16#31,SMB2
                 // Если "1" (соединение),
MOVB     16#2,MB2
                 // установить состояние
                 // соединения

R        T38,1   // Перезапуск таймера
                 // ожидания

CRETI            // Возврат

LD       SM0.0
MOVB     16#4,MB2
                 // Иначе показать ошибки

RETI
```

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все прав защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC
S7-200 Примеры

Группа	Тема
3	**Подключение нескольких S7-200 CPU в децентрализованную сеть в режиме свободнопрограммируемого интерфейса**

Краткое описание

В данном примере соединены между собой 3 SIMATIC S7-200 CPU 214. Станция 0 (активная станция) соединена со станциями 1 и 2 (пассивными станциями). Активная станция посылает четыре байта выходных данных по очереди каждой станции. А каждая станция отвечает четырьма байтами входных данных. Для обмена данными используется режим свободнопрограммируемого интерфейса.

Зарезервированы два буфера памяти, один для удаленных входов и один для выходов. Передаваемые выходные данные копируются в виде двойного слова из буфера выходов в буфер передачи. После передачи, активная станция получает ответ от пассивных станций и сохраняет данные в буфере приема.

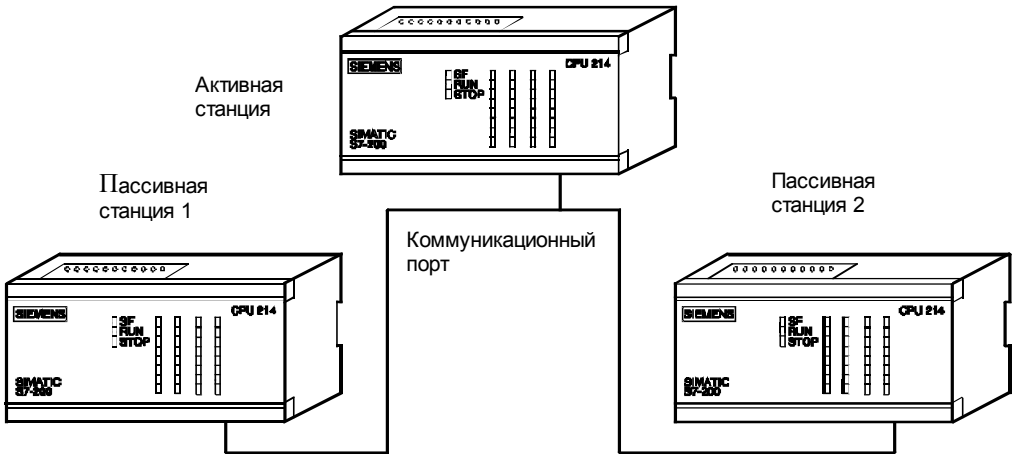


Рисунок 27.1

Аппаратные требования

Для работы программы необходимо:
(2 SIMATIC S7-200 CPU 212 или CPU 214
1 Соединительный кабель 9 штырьковый
Если используется больше, чем два ПЛК (больше, чем одна пассивная станция), то необходимы сетевые штеккеры (Siemens - Зак. №: 6ES7-972-0BB00-0XA0 или 6ES5-762-2AA21).

Структура программы: Активная станция

Main Главная программа

SBR 0 Выбор протокола PPI или свободнопрограммируемого протокола

INT 0 Приемный таймер тайм-аута
INT 1 Передача тайм-аута
INT 10 Передача после передачи данных выходов
INT 11 Прием первого символа телеграммы
INT 12 Прием данных входов
INT 13 Прием FCS знака
INT 14 Обнаружение свободной линии

Описание программы: Активная станция

В данной программе объединяются два, три или более ПЛК SIMATIC S7-200 CPU 214 или CPU 212 в информационную сеть.

Инициализация UART: Контроль четности
8 бит/знак
38,400 Бод (CPU 214)
19,200 Бод (CPU 212)

Если используются только ПЛК SIMATIC CPU 214, то скорость передачи 38,400 Бод.
Если используются только ПЛК SIMATIC CPU 212, то автоматически скорость передачи устанавливается на 19,200 Бод.
Если используются ПЛК SIMATIC CPU 214 и CPU 212, то программа д.б. изменена: Значение байта SMB30 надо изменить с 16#C0 на 16#C4 и с 16#C1 на 16#C5. Скорость передачи будет 19,200 Бод.

Функционирование программы:

Каждая телеграмма проверяется при помощи контрольного символа последовательности, который формируется как исключающего ИЛИ для байт данных телеграммы.
Активная станция удаленной периферии - это станция 0 - м.б. подключена к одной, двум или трем пассивным станциям. Если подсоединяется одна пассивная станция, то она д.б. станцией 1. Если подсоединяется две пассивные станции, то это д.б. станции 1 и 2. Если подсоединяется три пассивные станции, то это д.б. станции 1, 2 и 3.
Количество подключенных пассивных станций д.б. указано как параметр в V0 активной станции. Для пассивных станций в V0 храниться адрес станции.
Активная станция посылает четыре байта выходных данных по очереди каждой станции, а каждая станция отвечает четырьма байтами входных данных.
Два буфера V памяти зарезервировано, один для удаленных входов и один для удаленных выходов.
Области буферов в активной станции показаны на следующей диаграмме.

Станция 1		Станция 2		Станция 3	
Буфер входов		Буфер входов		Буфер входов	
VB 500	Байт 0	VB 504	Байт 0	VB 508	Байт 0
VB 501	Байт 1	VB 505	Байт 1	VB 509	Байт 1
:	:	:	:	:	:
VB 503	Байт 3	VB 507	Байт 3	VB 511	Байт 3

Станция 1		Станция 2		Станция 3	
Буфер выходов		Буфер выходов		Буфер выходов	
VB 540	Байт 0	VB 544	Байт 0	VB 548	Байт 0
VB 541	Байт 1	VB 545	Байт 1	VB 549	Байт 1
:	:	:	:	:	:
VB 543	Байт 3	VB 547	Байт 3	VB 551	Байт 3

Передаваемые выходные данные копируются в виде двойного слова из буфера выходов в буфер передачи. После передачи, активная станция получает ответ от пассивных станций и сохраняет данные в буфере приема. Формат буферов передачи и приема приведен ниже.
VB607 - вспомогательная область, используемая при генерации передачи контрольной суммы (FCS означает Frame Check Sequence - Контрольная Последовательность Пакета).

	Буфер передачи	Буфер передачи	
VB 600	Длина	VB 608	Байт 0
VB 601	Адрес	VB 609	Байт 1
VB 602	Байт 0	VB 610	Байт 2
VB 603	Байт 1	VB 611	Байт 3
:	:		
VB 605	Байт 3		
VB 606	FCS		
VB 607	xx		

Формат телеграммы приведен ниже:

Адрес	B0	B1	B2	B3	FCS

Размер программы активной станции для удаленной периферии = 191 слов.

Более подробную информацию о ...
команде SWAP Вы найдете в Главе 4.6 “Команды перемещения”,
командах сдвига Вы найдете в Главе 5.1 “Команды сдвига”,
программах обработки прерываний Вы найдете в Главе 6.2 “Команды прерываний” и XMT Вы найдете в Главе 6.4 “Команды передачи”
в Руководстве по Программированию SIMATIC S7-200.

LAD (S7-MicroDOS)	STL (IEC)

Главная программа

// Программа инициализируется в главном меню.

```
1   SM0.0                    0
   ─┤   ├──────────────( CALL   )

2  ──────────────────────( MEND   )
```

```
LD      SM0.0
CALL    0    //Вызов подпрограммы
// Начало программы пользователя

// Конец программы пользователя

MEND
```

Подпрограммы

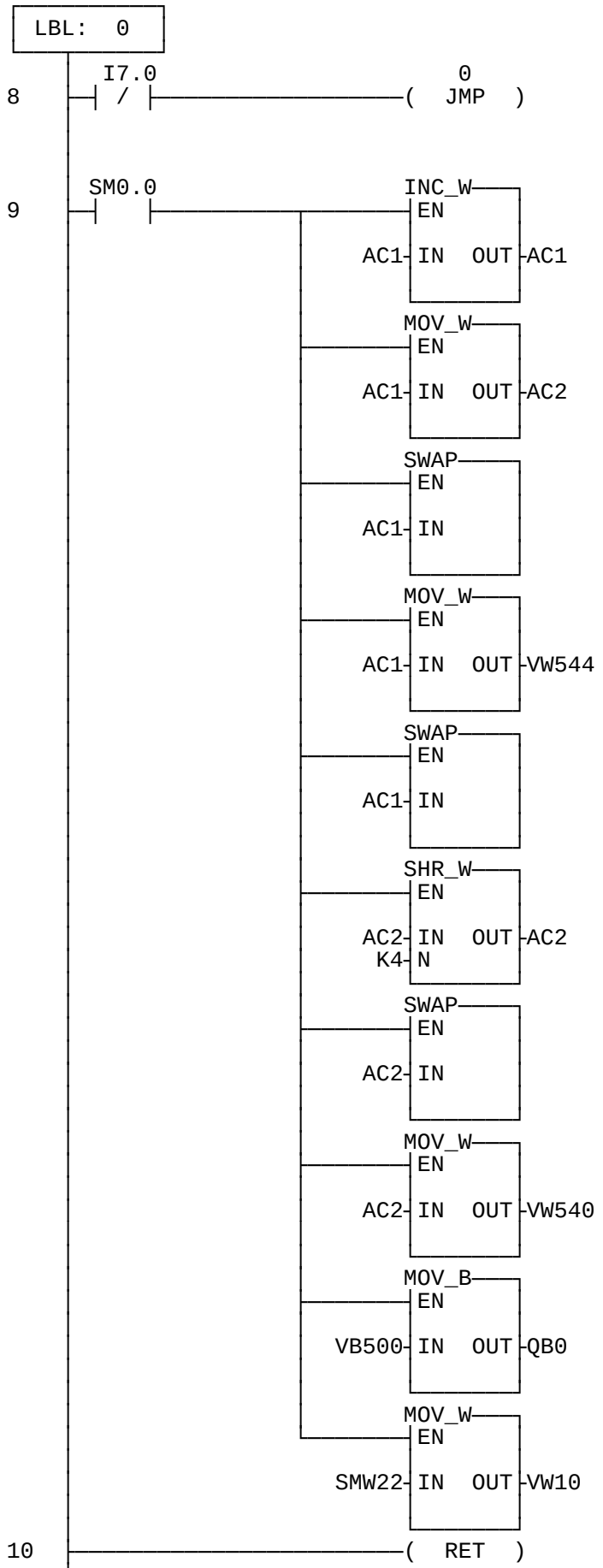
// **********
// Подпрограмма 0 инициализирует режим свободнопрограммируемого интерфейса
// и манипулирует указателями.

```
SBR:  0

4   SM0.7                    MOV_B
   ─┤ / ├──────────┬──────┬─┤EN
                    │      │
              KHC0─┤IN  OUT├─SMB30
                    │      │
                    │      │  DTCH
                    │      │─┤EN
                    │      │
                    │      │ 8─┤EVT
                    │      │
                    │      │  DTCH
                    │      │─┤EN
                    │      │
                    │      │ 9─┤EVT
                    │      │
                    │      │  DTCH
                    │      │─┤EN
                    │      │
                    │      │10─┤EVT
                    │      │
                    │      │
                    │      │( CRET   )

5   SM30.0                   MOV_B
   ─┤ / ├──────────┬──────┬─┤EN
                    │      │
              KHC1─┤IN  OUT├─SMB30
                    │      │
                    │      │
                    │      │(   ENI   )
                    │      │
                    │      │  MOV_B
                    │      │─┤EN
                    │      │
                    │      │
              VB1─┤IN  OUT├─SMB34
                    │      │
```

```
SBR     0

LDN     SM0.7  // Если переключатель
               // в положении TERM
MOVB    16#C0,SMB30
               // Блокировать режим
               // свободнопрограми-
               // руемого интерфейса

DTCH    8      // Блокировать
               // прерывания приема

DTCH    9      // Блокировать
               // прерывания
               // передачи

DTCH    10     // Прерывания
               // таймера

CRET

LDN     SM30.0 // Если не режим
               // свободнопрограми-
               // руемого интерфейса
MOVB    16#C1,SMB30
               // Разрешить режим
               // свободнопрограми-
               // руемого интерфейса
               // с четностью, 8 бит/
               // знак, 38.4 КБод
ENI            // Разрешить
               // прерывания

MOVB    VB1,SMB34
               // Установить значение
               //тайм-аута
```

```
                    └──────( CRET  )
                                                  CRET

      SM0.0                I7.0     K1
6   ─┤   ├──────────────────( R   )
                    ┌──────MOV_DW──┐
                    ├──────┤EN      │
                    │ &VB540─┤IN  OUT├─VD630
                    │       └──────┘
                    ├──────MOV_DW──┐
                    ├──────┤EN      │
                    │ &VB500─┤IN  OUT├─VD634
                    │       └──────┘
                    ├──────MOV_B───┐
                    ├──────┤EN      │
                    │     K6─┤IN  OUT├─VB600
                    │       └──────┘
                    ├──────MOV_B───┐
                    ├──────┤EN      │
                    │     K1─┤IN  OUT├─VB601
                    │       └──────┘
                    ├──────MOV_DW──┐
                    ├──────┤EN      │
                    │ *VD630─┤IN  OUT├─VD602
                    │       └──────┘
                    ├──────WXOR_W──┐
                    ├──────┤EN      │
                    │ VW602─┤IN1 OUT├─AC0
                    │ VW604─┤IN2    │
                    │       └──────┘
                    ├──────MOV_B───┐
                    ├──────┤EN      │
                    │    AC0─┤IN  OUT├─VB606
                    │       └──────┘
                    ├──────WXOR_W──┐
                    ├──────┤EN      │
                    │    AC0─┤IN1 OUT├─VW606
                    │ VW606─┤IN2    │
                    │       └──────┘
                    ├──────ATCH────┐
                    ├──────┤EN      │
                    │     K1─┤INT    │
                    │    K10─┤EVT    │
                    │       └──────┘
                    ├──────ATCH────┐
                    ├──────┤EN      │
                    │    K10─┤INT    │
                    │     K9─┤EVT    │
                    │       └──────┘
                    └──────XMT─────┐
                    ├──────┤EN      │
                    │ VB600─┤TBL    │
                    │     0─┤POR    │
                    │       └──────┘
```

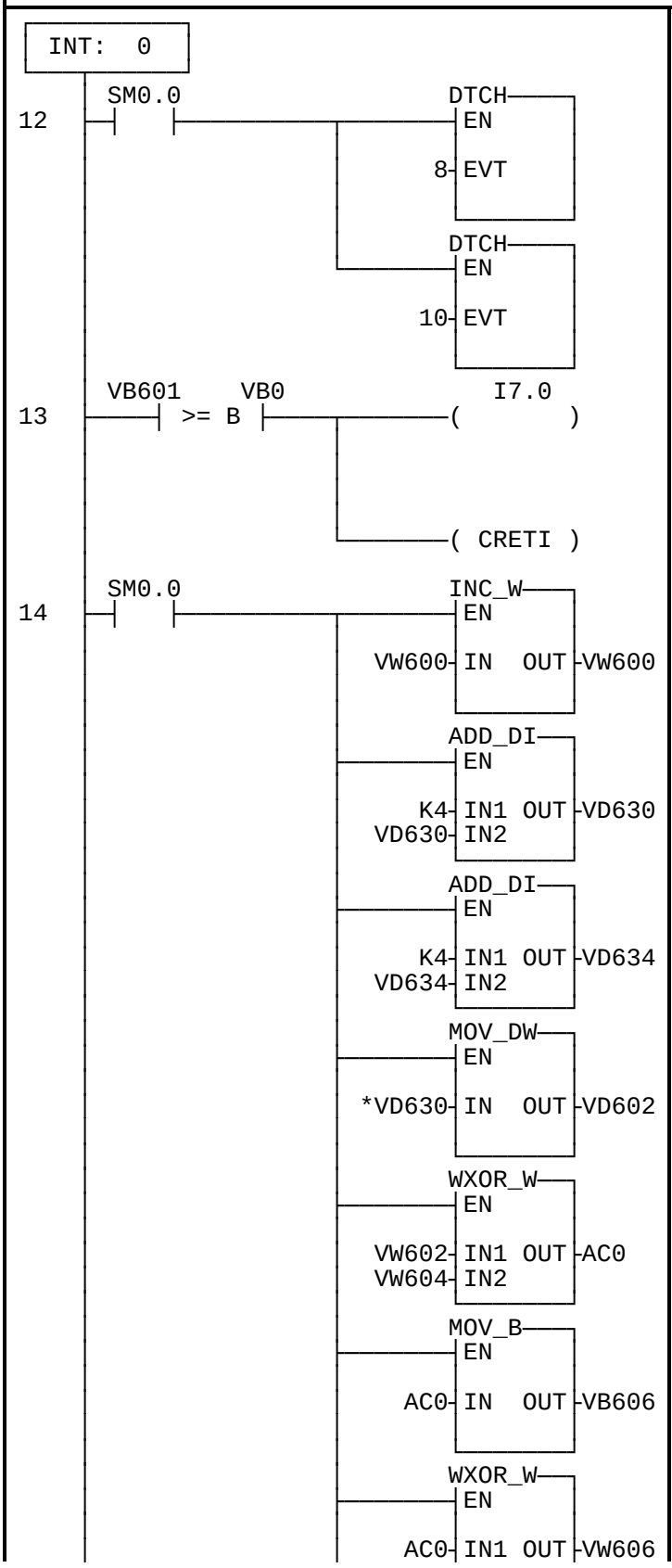
LD	SM0.0	// Бит всегда „1“
R	I7.0,1	// Сбросить индикатор // удаленной // периферии
MOVD	&VB540,VD630	// Загрузить указатель // на буфер данных // выходов
MOVD	&VB500,VD634	// Загрузить указатель // на буфер данных // входов
MOVB	6,VB600	// Загрузить длину // буфера передачи
MOVB	1,VB601	// Адрес станции = 1
MOVD	*VD630,VD602	// Загрузить данные // для передачи
MOVW	VW602,AC0	// Вычислить FCS
XORW	VW604,AC0	
MOVB	AC0,VB606	
XORW	AC0,VW606	// Сохранить FCS
ATCH	1,10	// Разрешить таймер // тайм-аута // передатчика
ATCH	10,9	// Разрешить // прерывание // передатчика
XMT	VB600,0	// Передача данных



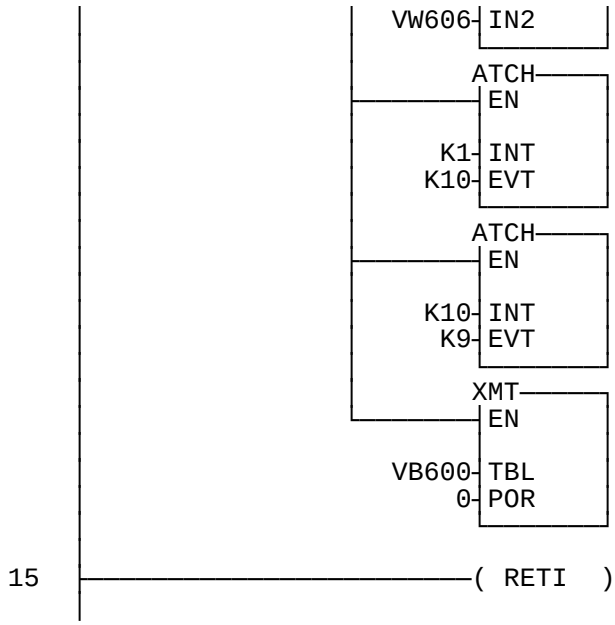

```
LBL   0
LDN   I7.0     // Если актуализация
               // удаленной
               // периферии не
               // завершена
JMP   0        // То ожидание
               // завершения
LD    SM0.0    // Бит всегда „1“
INCW  AC1
MOVW  AC1,AC2
SWAP  AC1
MOVW  AC1,VW544
SWAP  AC1
SRW   AC2,4
SWAP  AC2
MOVW  AC2,VW540
MOVB  VB500,QB0
MOVW  SMW22,VW10
RET
```

Программы обработки прерываний

```
// ***********************************************************************
// Программа обработки прерывания 0 отделяет прерывание приема и прерывание
// таймера при передачи или приеме данных, если таймер перезапускается.
```




```
INT      0        // Прием таймера
                  // тайм-аута

LD       SM0.0
DTCH     8     // Блокировать
                  // прерывания приема

DTCH     10       // Блокировать
                  // прерывания по
                  // времени

LDB>=    VB601,VB0
                  // Если это последняя
                  // пассивная станция
                  // в сети,
=        I7.0     // Индицировать, что
                  // цикл удаленной
                  // периферии
         // завершен
CRETI

LD       SM0.0    // Бит всегда „1“
INCW     VW600    // Иначе, увеличить на
                  // 1 адрес станции

+D       4,VD630
         // Увеличить на 1
         // указатель на                    //
выходы следующей
                  // станции

+D       4,VD634
                  // Увеличить на 1
                  // указатель на входы
                  // следующей станции

MOVD     *VD630,VD602
                  // Загрузить данные
         // для передачи

MOVW     VW602,AC0
                  // Вычислить FCS
XORW     VW604,AC0

MOVB     AC0,VB606

XORW     AC0,VW606
                  // Сохранить FCS
```

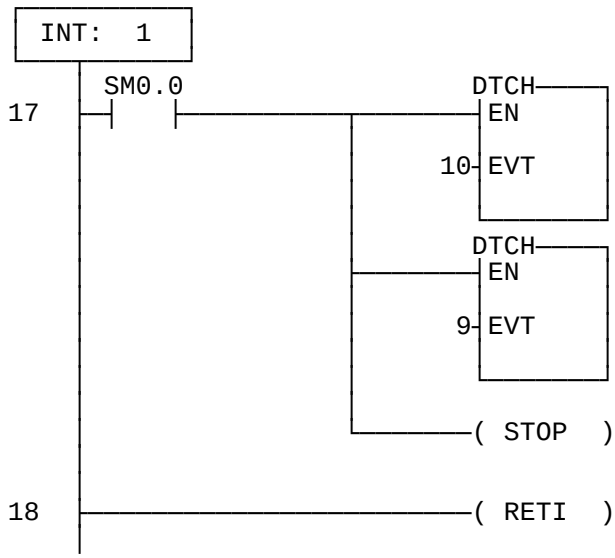


ATCH	1,10	// Разрешить таймер
		// тайм-аута
		// передатчика

ATCH	10,9	// Разрешить
		// прерывания
		// передатчика

XMT	VB600,0	// Передача данных
RETI		

// **********
// Программа обработки прерывания 1 обрабаттывакт XMT тайм-аут таймера



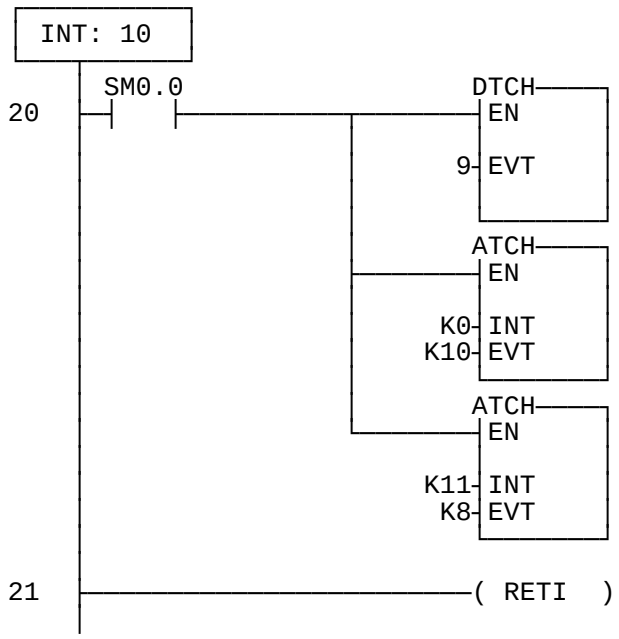
INT	1	// Передача
		// телеграммы
		// тайм-аута таймера

LD	SM0.0	// Бит всегда „1“
DTCH	10	// Останов таймера
DTCH	9	// Останов // передатчика

STOP		// Переход в режим
		// основной
		// программы

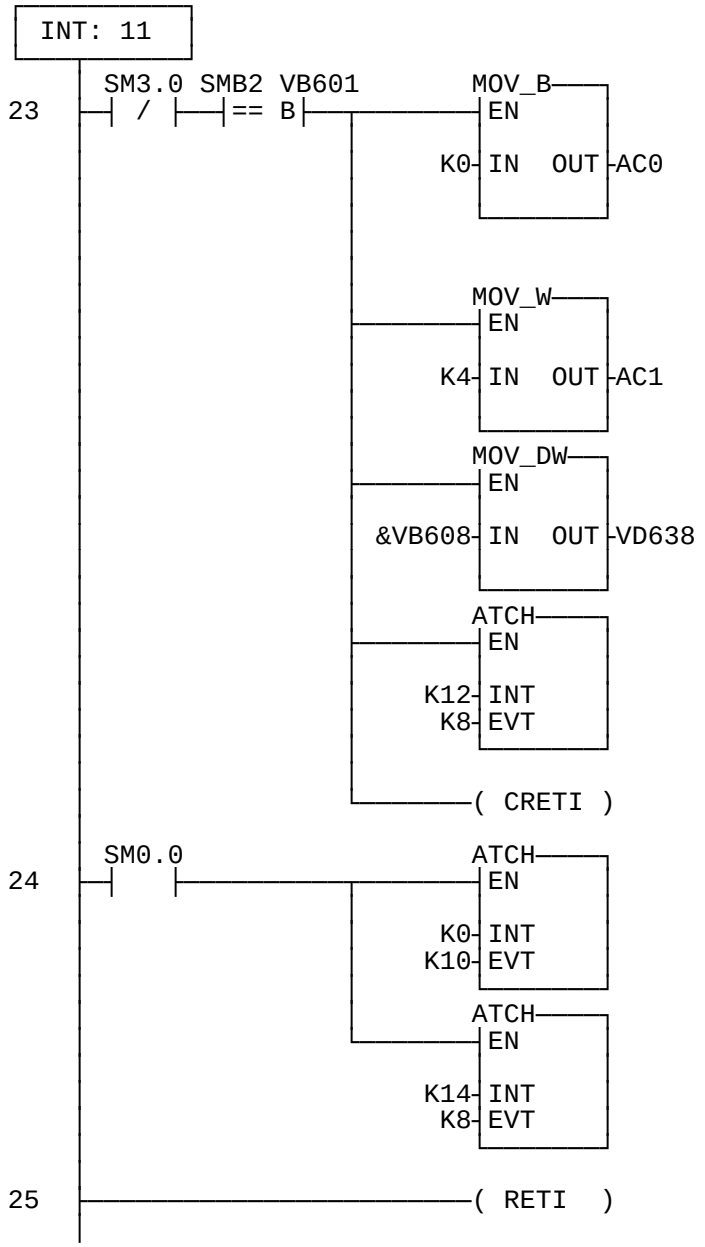
RETI		

```
// ************************************************************************
// Программа обработки прерывания 10 прерывает программу для прием данных.
```




```
INT     10      // Прерывание
                // передатчика

LD      SM0.0   // Бит всегда „1“
DTCH    9       // Блокировать
                // прерывания
                // передачи

ATCH    0,10    // Запуск таймера
                // тайм-аута

ATCH    11,8    // Разрешить прием
                // входных данных

RETI
```

```
// ************************************************************************
// Программа обработки прерывания 11 прерывает программу, если принят
// первый знак телеграммы.
```




```
INT     11          // Прием первого
                    // знак телеграммы
LDN     SM3.0       // Если нет ошибки
                    // четности,
AB=     SMB2,VB601
                    // И если отвечает
                    // соответствующая
                    // станция, то
MOVB    0,AC0       // Инициализировать
                    // регистр контрольной
                    // суммы
MOVW    4,AC1       // Загрузить счетчик
                    // приема знаков
MOVD    &VB608,VD638
                    // Загрузить указатель
                    // на буфер приема
ATCH    12,8        // Разрешить прием
                    // данных
CRETI
LD      SM0.0       // Иначе, отвечает
                    // не та станция
                    // или есть ошибка
ATCH    0,10        // Разрешить таймер
                    // приема тайм-аута
ATCH    14,8        // Разрешить
                    // обнаружение
                    // свободной линии
RETI
```

```
// ************************************************************************
// Программа обработки прерывания 12 прерывает программу, если приняты данные.
```

```
INT: 12
27  SM3.0 / ──┬──────────────────────────────────────────────────────
              │                          MOV_B
              │                          EN
              │                   SMB2-  IN   OUT -*VD638
              │
              │                          INC_DW
              │                          EN
              │                  VD638-  IN   OUT -VD638
              │
              │                          WXOR_W
              │                          EN
              │                   SMW1-  IN1  OUT -AC0
              │                    AC0-  IN2
              │
              │                          DEC_W
              │                          EN
              │                    AC1-  IN   OUT -AC1
              │
28  SM1.0 ────┼──────────────────────────────────────────────────────
              │                          ATCH
              │                          EN
              │                    K13-  INT
              │                     K8-  EVT
              │
              │                          ( CRETI )
29  SM3.0 ────┼──────────────────────────────────────────────────────
              │                          ATCH
              │                          EN
              │                     K0-  INT
              │                    K10-  EVT
              │
              │                          ATCH
              │                          EN
              │                    K14-  INT
              │                     K8-  EVT
30            │                          ( RETI  )
```

INT	**12**	**// Прием данных**
LDN	**SM3.0**	**// Если нет ошибки // четности,**
MOVB	**SMB2,*VD638**	**// Сохранить данные в // регистре приема**
INCD	**VD638**	**// Указатель на // следующий буфер // приема**
XORW	**SMW1,AC0**	**// Вычислить // контрольную сумму**
DECW	**AC1**	**// Уменьшить на 1 // счетчиу принятых // знаков**
LD	**SM1.0**	**// Если было принято // четыре знака,**
ATCH	**13,8**	**// Разрешить прием // FCS**
CRETI		
LD	**SM3.0**	**// Если есть ошибка // четности,**
ATCH	**0, 10**	**// Разрешить таймер // приема тайм-аута**
ATCH	**14,8**	**// Разрешить // обнаружение // свободной линии**
RETI		

```
INT: 13
32  SM0.0                    ATCH
    ─┤   ├──────────────────┤EN
                          K0┤INT
                         K10┤EVT
33  SM3.0 SMB2 AC0
    ─┤   ├──┤== B├──────────( CRETI )
34  SM0.0                    MOV_DW
    ─┤   ├──────────────────┤EN
                     VD608┤IN  OUT├*VD634
35  ──────────────────────────( RETI  )
```

```
INT     13      // Прием знака FCS

LD      SM0.0   // Бит всегда „1“
ATCH    0,10    // Разрешить таймер
                // приема тайм-аута

LD      SM3.0   // Если есть ошибка
                // четности,
AB=     SMB2,AC0
                // Или, если  FCS не
        // соответствует
CRETI

LD      SM0.0   // Иначе,
MOVD    VD608,*VD634
                // Сохранить принятые
                // данные

RETI
```

// **********
// Программа обработки прерывания 14 перезапускает таймер приема тайм-аута.

```
INT: 14
37  SM0.0                    ATCH
    ─┤   ├──────────────────┤EN
                          K0┤INT
                         K10┤EVT
38  ──────────────────────────( RETI  )
```

```
INT     14      // Обнаружение
                // свободной линии

LD      SM0.0   // Бит всегда „1“
ATCH    0,10    // Новый запуск
                // таймера приема
                // тайм-аута

RETI
```

Структура программы: Пассивная станция

Main **Главная программа**

SBR 0 **Програма инициализации станции при включении питания**

INT 0 Таймер свободной линии
INT 1 Передача тайм-аута
INT 10 Передача после передачи данных выходов
INT 11 Прием первого символа телеграммы
INT 12 Прием данных входов
INT 13 Прием FCS знака
INT 14 Обнаружение свободной линии

Описание программы: Пассивная станция

В данной программе объединяются два, три или более ПЛК SIMATIC S7-200 CPU 214 или CPU 212 в информационную сеть.

Инициализация UART: Контроль четности
8 бит/знак
38,400 Бод (CPU 214)
19,200 Бод (CPU 212)

Если используются только ПЛК SIMATIC CPU 214, то скорость передачи 38,400 Бод.
Если используются только ПЛК SIMATIC CPU 212, то автоматически скорость передачи устанавливается на 19,200 Бод.
Если используются ПЛК SIMATIC CPU 214 и CPU 212, то программа д.б. изменена:
Значение байта SMB30 надо изменить с 16#C0 на 16#C4 и с 16#C1 на 16#C5.
Скорость передачи будет 19,200 Бод.

Функционирование программы:

Каждая телеграмма проверяется при помощи контрольного символа последовательности, который формируется как исключающего ИЛИ для байт данных телеграммы.
Активная станция удаленной периферии - это станция 0 - м.б. подключена к одной, двум или трем пассивным станциям. Если подсоединяется одна пассивная станция, то она д.б. станцией 1. Если подсоединяется две пассивные станции, то это д.б. станции 1 и 2. Если подсоединяется три пассивные станции, то это д.б. станции 1, 2 и 3.
Количество подключенных пассивных станций д.б. указано как параметр в V0 активной станции. Для пассивных станций в V0 храниться адрес станции..

Каждая пассивная станция имеет входной и выходной буфер, как показано ниже.

	Буфер входов
IB0	Байт 1
IB1	Байт 2
:	:
IB3	Байт 3

	Буфер выходов
QB0	Байт 1
QB1	Байт 2
:	:
QB3	Байт 3

Когда активная станция передает телеграмму, пассивная станция, к которой адресуется активная станция, сохраняет выходные данные в буфере приема и отвечает на телеграмму актиной станции путем передачи входных данных в буфер передачи. Этот буфер показан ниже. VB607 - вспомогательная область, используемая при генерации передачи контрольной суммы.

	Буфер передачи		Буфер передачи
VB 600	Длина	VB 608	Байт 0
VB 601	Адрес	VB 609	Байт 1
VB 602	Байт 0	VB 610	Байт 2
VB 603	Байт 1	VB 611	Байт 3
:	:		
VB 605	Байт 3		
VB 606	FCS		
VB 607	xx		

Формат телеграммы показан ниже:

Адрес	B0	B1	B2	B3	FCS

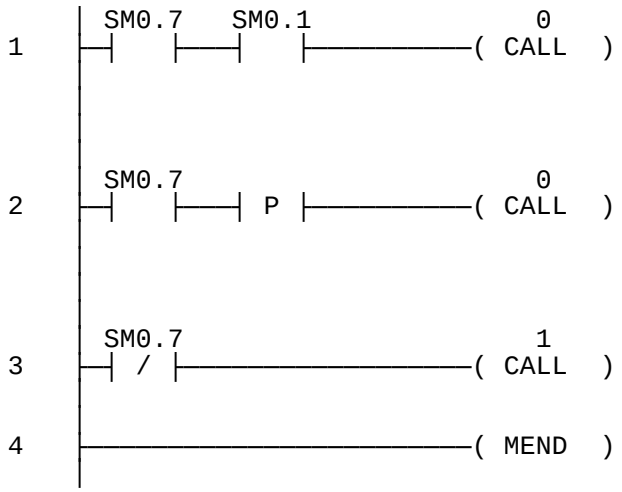
Размер программы пассивной станции для удаленной периферии = 148 слов.

Более подробную информацию о ...
команде SWAP Вы найдете в Главе 4.6 “Команды перемещения”,
командах сдвига Вы найдете в Главе 5.1 “Команды сдвига”,
программах обработки прерываний Вы найдете в Главе 6.2 “Команды прерываний” и
XMT Вы найдете в Главе 6.4 “Команды передачи”
в Руководстве по Программированию SIMATIC S7-200.

LAD (S7-MicroDOS)	STL (IEC)

Главная программа

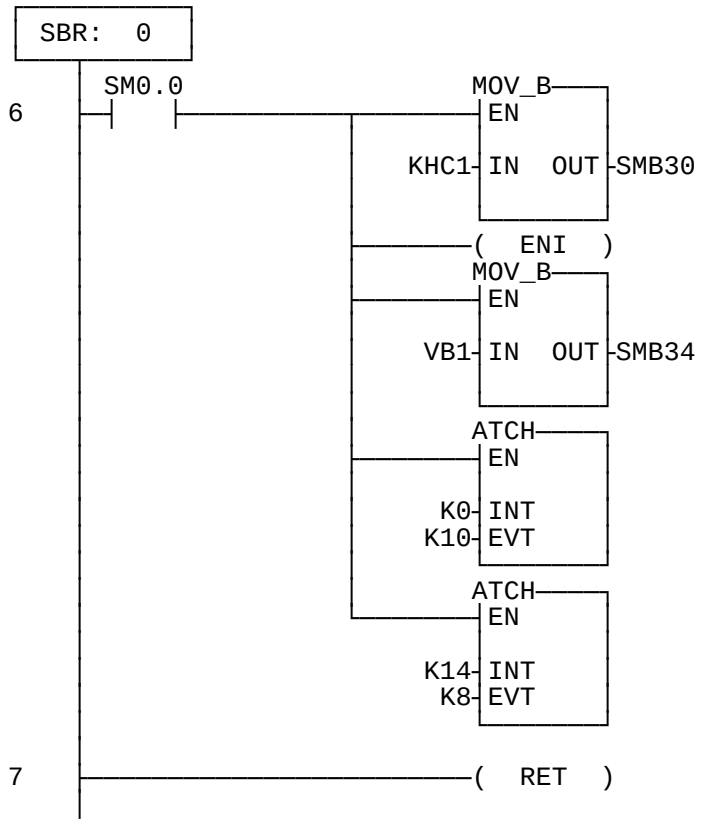
// Главное меню инициализации программы.




```
LD      SM0.7   // Если переключатель
                // в позиции RUN,
A       SM0.1   // И это первый цикл

CALL    0       // Запуск обмена

LD      SM0.7   // Если переключатель
                // переводится в
                // положение RUN,
EU
CALL    0       // Запуск обмена

LDN     SM0.7   // Если переключатель
                // в позиции TERM,
CALL    1       // Блокировать обмен

MEND
```

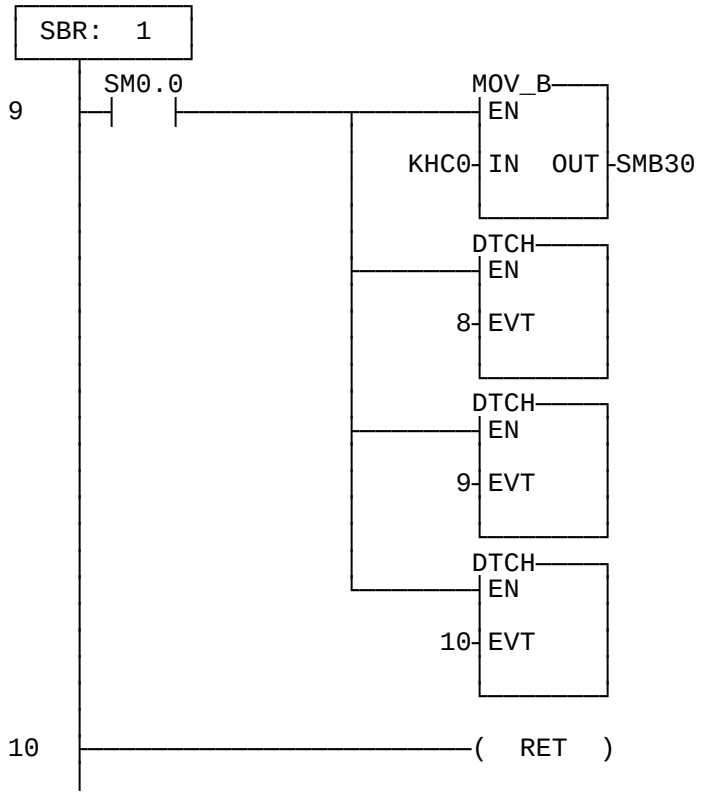
Подпрограммы

// **********
// Подпрограмма 0 устанавливает режим свободнопрограммируемого интерфейса и
// разрешает прерывания для таймера свободной линии и обнаружения свободной линии.




```
SBR     0           // Инициализация
                    // пассивной станции
LD      SM0.0       // Бит всегда „1“
MOVB    16#C1,SMB30
                    // Разрешить режим
                    // свободнопрограм
                    // мируемого
                    // интерфейса с
                    // четностью, 8 бит/
                    // знак, 38.4 КБод
ENI                 // Разрешить
                    // прерывания
MOVB    VB1,SMB34
                    // Установить значение
                    // тайм-аута = 5 ms

ATCH    0,10        // Разрешить таймер
                    // свободной линии

ATCH    14,8        // Разрешить
                    // обнаружение
                    // свободной линии

RET                 // Конец подпрограммы
```

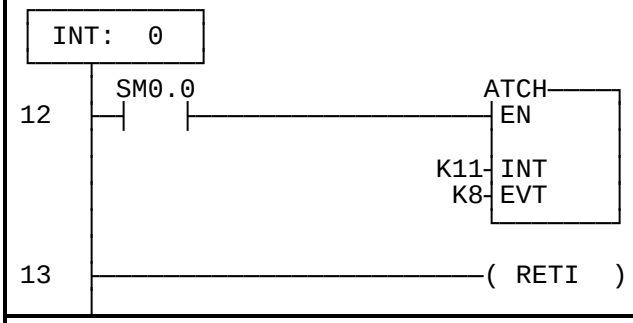
```
// ************************************************************************
// Подпрограмма 1 блокирует режим свободнопрограммируемого интерфейса и
// отделяет прерывания приема, передачи и времени.
```




```
SBR     1          // Блокировать обмен
LD      SM0.0      // Бит всегда „1“
MOVB    16#C0,SMB30
                   // Блокировать режим
                   // свободнопрограмми-
                   // руемого интерфейса
DTCH    8          // Блокировать
                   // приемник
DTCH    9          // Блокировать
                   // передатчик
DTCH    10         // Блокировать
                   // таймер
RET                // Конец подпрограммы
```

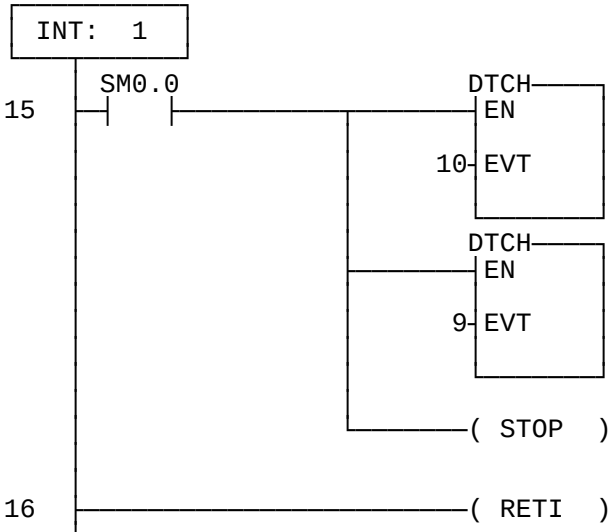
Программы обработки прерываний

// **********
// Программа обработки прерывания 0 разрешает прием входных данных.



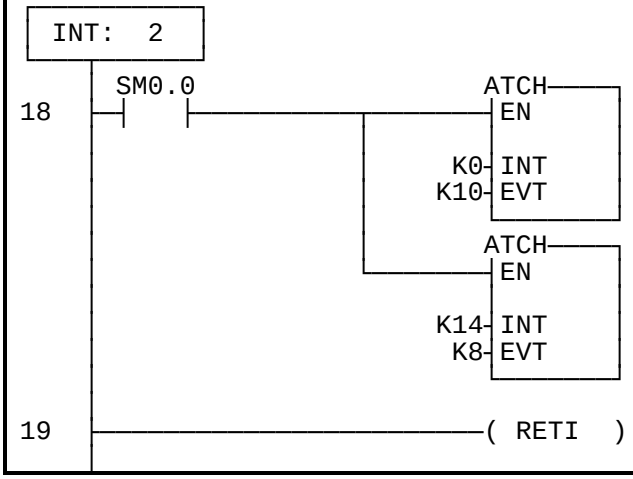

```
INT     0       // Таймер тайм-аута
                // свободной линии
LD      SM0.0   // Бит всегда „1“
ATCH    11,8    // Разрешить прием
                // входных данных

RETI
```

// **********
// Программа обработки прерывания 1 отделяет прерывания таймера и передачи при
// передаче или приеме данных или если перезапущен таймер.



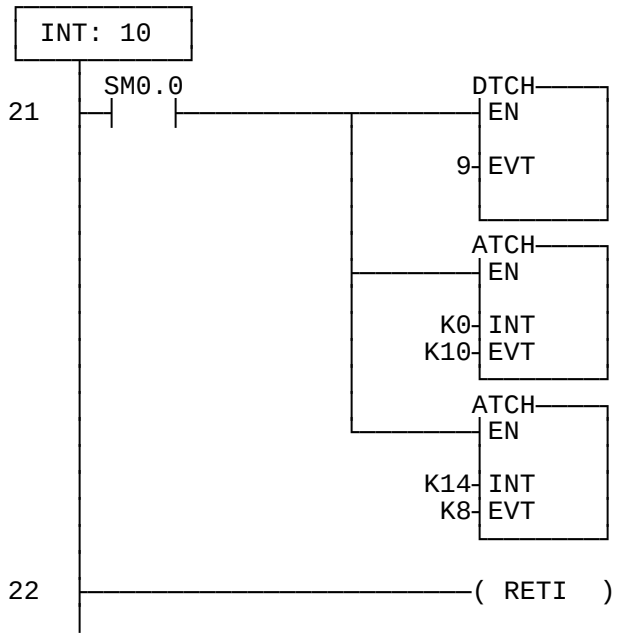

```
INT     1       // Передача таймера
                // тайм-аута
                // телеграммы
LD      SM0.0   // Бит всегда „1“
DTCH    10      // Останов таймера

DTCH    9       // Останов
                // передатчика

STOP            // Переход в обычный
                // режим программы

RETI
```

// **********
// Программа обработки прерывания 2 прерывает программму, если принят
// первый знак телеграммы.



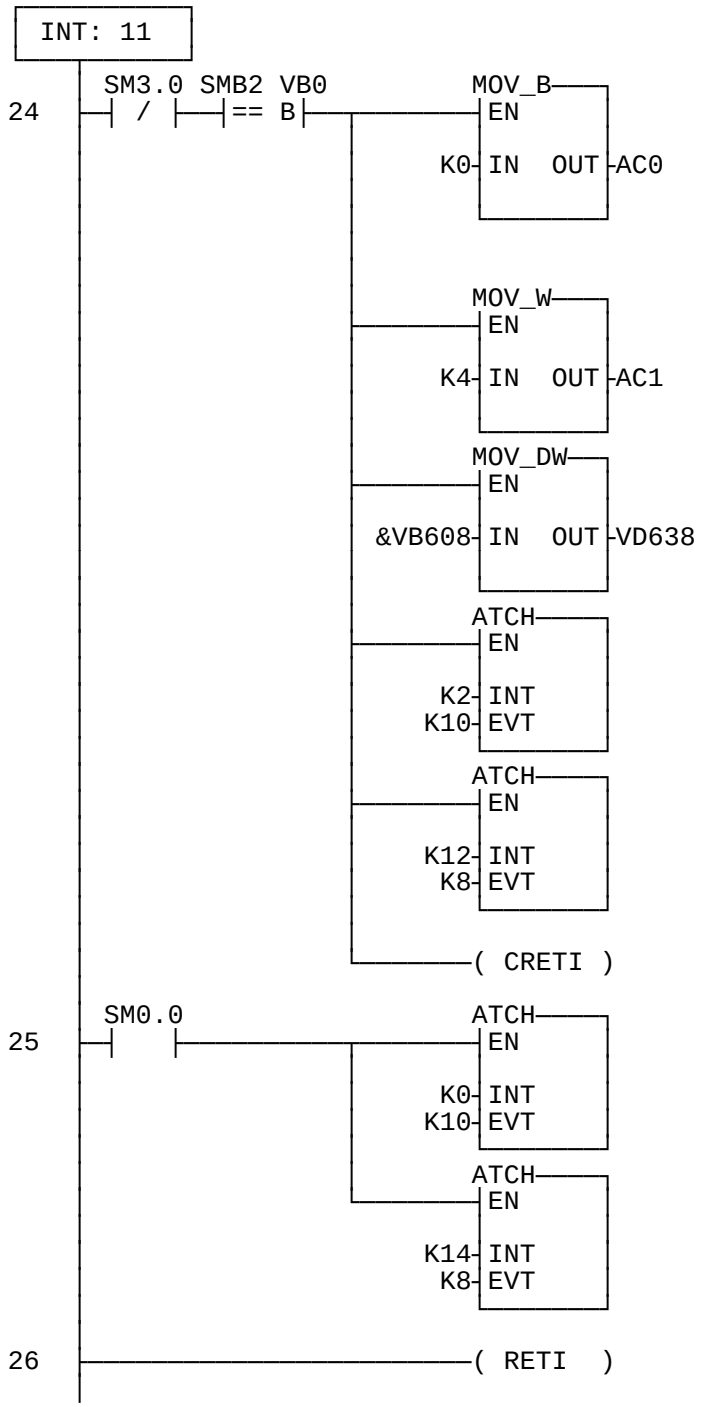

```
INT     2       // Таймер тайм-аута
                // Телеграммы
LD      SM0.0   // Бит всегда „1“
ATCH    0,10    // Разрешить таймер
                // тайм-аута
                // свободной линии

ATCH    14,8    // Разрешить
                // обнаружение
                // свободной линии

RETI
```

```
// ************************************************************************
// Программа обработки прерывания 10 прерывает программу, если передана
// телеграмма. Прерывание передачи блокировано. Запускается таймер
// свободной линии и разрешается обнаружение свободной линии.
```



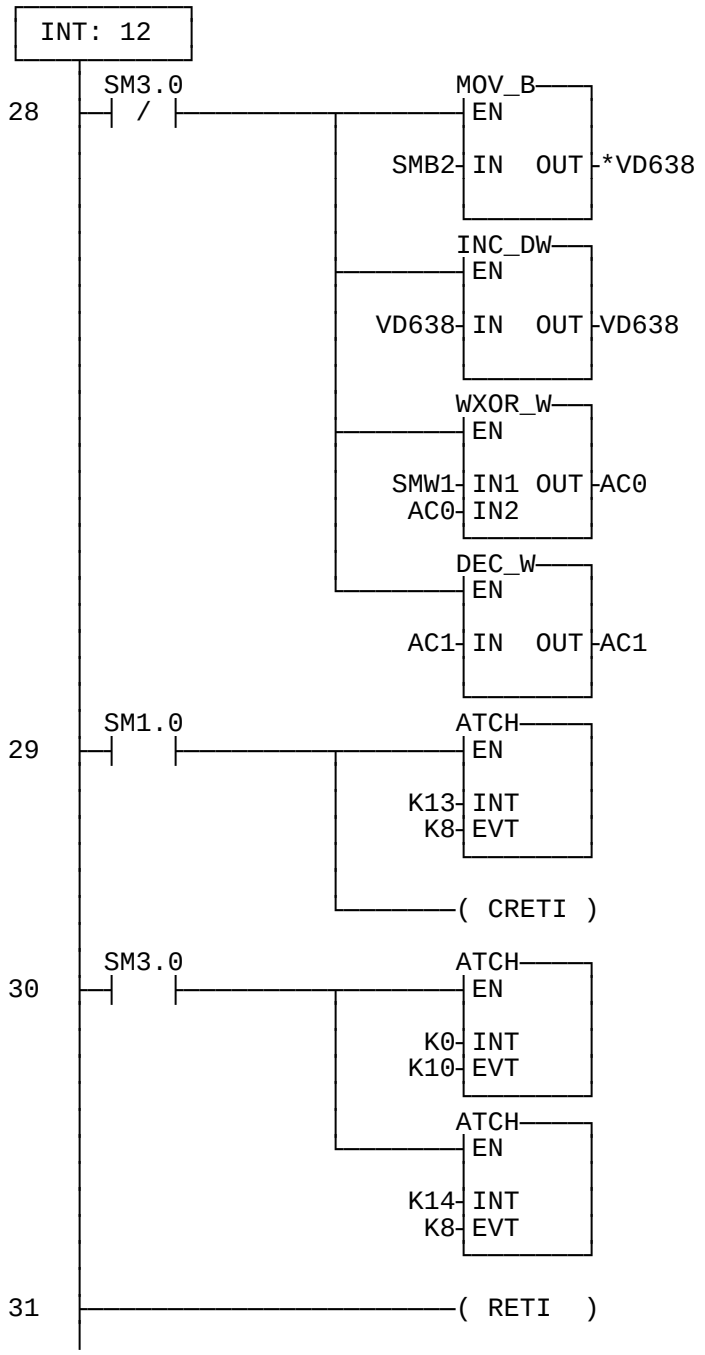

```
INT     10      // Прерывание
                // передатчика
LD      SM0.0   // Бит всегда „1“
DTCH    9       // Блокировать
                // прерывания
                // передачи

ATCH    0,10    // Запуск таймера
                // тайм-аута
                // свободной линии

ATCH    14,8    // Разрешить
        // обнаружение
                // свободной линии

RETI
```

```
// ************************************************************************
// Программа обработки прерывания 11 активируется, если принят первый знак
// телеграммы. Устанавливается таймер тайм-аута телеграммы. Данные м.б. приняты.
// Если обнаружена ошибка, то разрешается таймер тайм-аута свободной линии и
// обнаружение свободной линии.
```



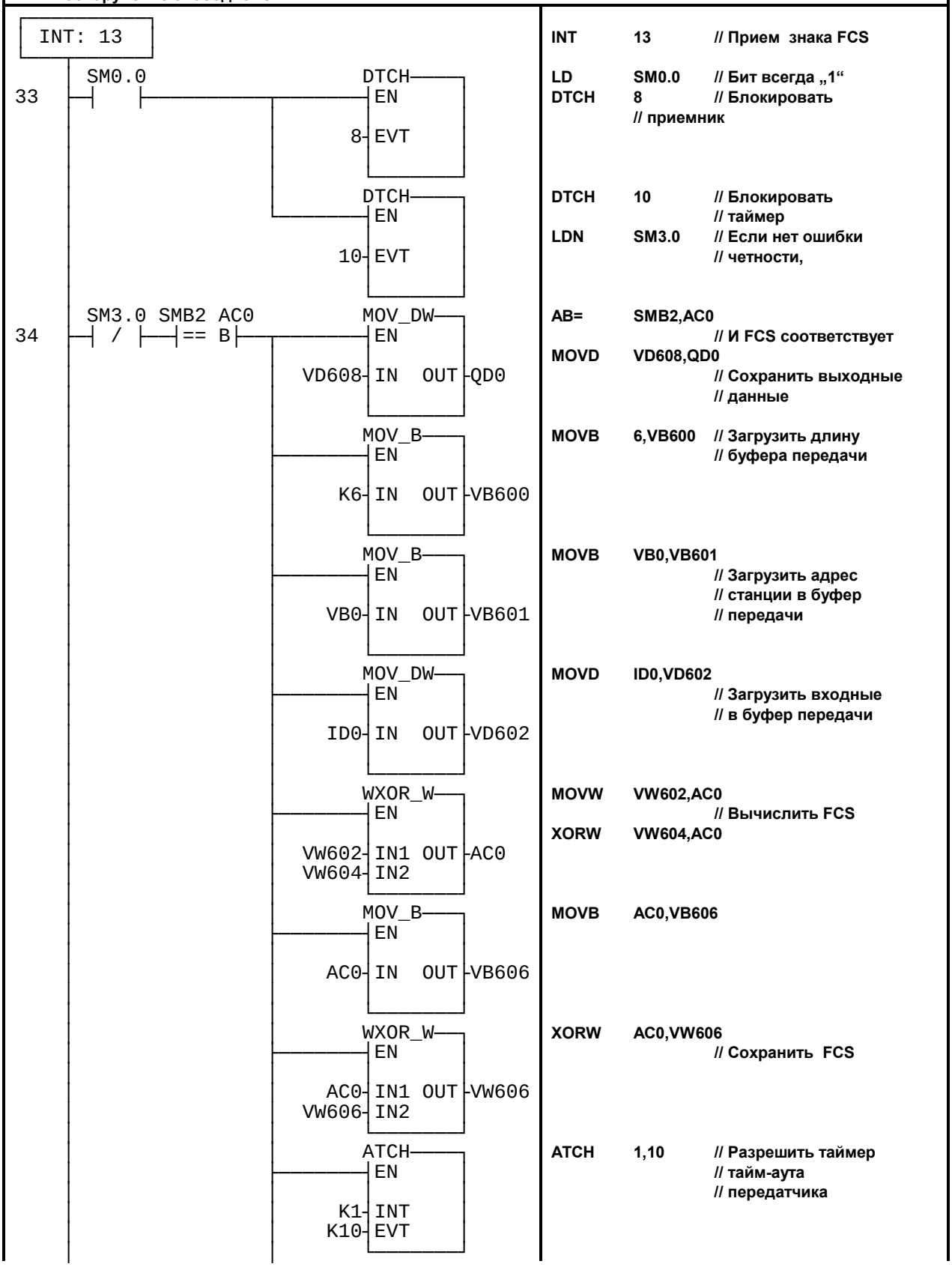

```
INT     11          // Прием первого
                    // знака телеграммы
LDN     SM3.0       // Если нет ошибки
                    // четности,
AB=     SMB2,VB0
                    // И если телеграмма
                    // для этой станции,
MOVB    0,AC0       // То инициализация
                    // регистра
                    // контрольной суммы

MOVW    4,AC1       // Загрузить счетчик
                    // принятых знаков

MOVD    &VB608,VD638
                    // Загрузить указатель
                    // на буфер приема

ATCH    2,10        // Таймер тайм-аута
                    // телеграммы

ATCH    12,8        // Разрешить прием
                    // данных

CRETI

LD      SM0.0       // Иначе, если есть
                    // ошибкачетности,
ATCH    0,10        // Разрешить таймер
                    // свободной линии

ATCH    14,8        // Разрешить
                    // обнаружение
                    // свободной линии

RETI
```

```
// ************************************************************************
// Программа обработки прерывания 12 прерывает программу, если данные приняты без
// ошибки. Информация сохраняется в буфере.
// Контрольная последовательность кадра FCS м.б. принята следующим прерыванием.
// Если обнаружена ошибка, то разрешается таймер тайм-аута свободной линии и
// обнаружение свободной линии.
```



INT	12	// Прием данных
LDN	SM3.0	// Если нет ошибки // четности,
MOVB	SMB2,*VD638	// Сохранить данные в // регистре приема
INCD	VD638	// Указатель на // следующий буфер // приема
XORW	SMW1,AC0	// Вычислить // контрольную сумму
DECW	AC1	// Уменьшить на 1 // количество // принятых знаков
LD	SM1.0	// Если было принято // четыре знака,
ATCH	13,8	// Разрешить прием // FCS
CRETI		
LD	SM3.0	// Если есть ошибка // четности то,
ATCH	0,10	// Разрешить таймер // свободной линии
ATCH	14,8	// Разрешить // обнаружение // свободной линии
RETI		

```
// ************************************************************************
// Программа обработки прерывания 13 принимает контрольную последовательность
// кадра (FCS). Прерывания таймера и приема блокированы. После вычисления FCS
// разрешаются прерывания тайм-аута и передатчика.
// Если обнаружена ошибка, то разрешается таймер тайм-аута свободной линии и
// обнаружение свободной линии.
```




```
INT     13          // Прием  знака FCS

LD      SM0.0       // Бит всегда „1“
DTCH    8           // Блокировать
        // приемник

DTCH    10          // Блокировать
                    // таймер
LDN     SM3.0       // Если нет ошибки
                    // четности,

AB=     SMB2,AC0
                    // И FCS соответствует
MOVD    VD608,QD0
                    // Сохранить выходные
                    // данные

MOVB    6,VB600     // Загрузить длину
                    // буфера передачи

MOVB    VB0,VB601
                    // Загрузить адрес
                    // станции в буфер
                    // передачи

MOVD    ID0,VD602
                    // Загрузить входные
                    // в буфер передачи

MOVW    VW602,AC0
                    // Вычислить FCS
XORW    VW604,AC0

MOVB    AC0,VB606

XORW    AC0,VW606
                    // Сохранить  FCS

ATCH    1,10        // Разрешить таймер
                    // тайм-аута
                    // передатчика
```

```
                          ATCH
                        EN
                    K10 INT
                     K9 EVT

                          XMT
                        EN
                  VB600 TBL
                      0 POR

                        ( CRETI )

   SM0.0                  ATCH
35 |  |                  EN
                      K0 INT
                     K10 EVT

                          ATCH
                        EN
                     K14 INT
                      K8 EVT

36                      ( RETI  )
```

ATCH	10,9	// Разрешить
		// прерывание
		// передатчика

XMT	VB600,0	// Передача данных
CRETI		
LD	SM0.0	// Иначе, если есть // ошибка четности,
ATCH	0,10	// Разрешить таймер // свободной линии

ATCH	14,8	// Разрешить
		// обнаружение
		// свободной линии

RETI		

// **********
// Программа обработки прерывания 14 разрешает прерывание тайм-аута.

```
INT: 14
   SM0.0                  ATCH
38 |  |                  EN
                      K0 INT
                     K10 EVT

39                      ( RETI  )
```

INT	14	// обнаружение // свободной линии
LD	SM0.0	// Бит всегда „1“

ATCH	0,10	// Перезапустить
		// таймер тайм-аута
		// свободной линии

RETI		

Copyright ©1995 by SIEMENS

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все прав защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC
S7-200 Tips

Group	Topic
4	Freeport Communication Interface to SIMOVERT Motor Drive

Overview

In this example, the CPU 214 communicates with a SIMOVERT MicroMaster motor drive to start, stop, and vary the output frequency to the motor. Communication occurs in the S7-200 Freeport mode using the USS 5-Word Protocol. An input simulator is used to initiate commands to the motor drive.

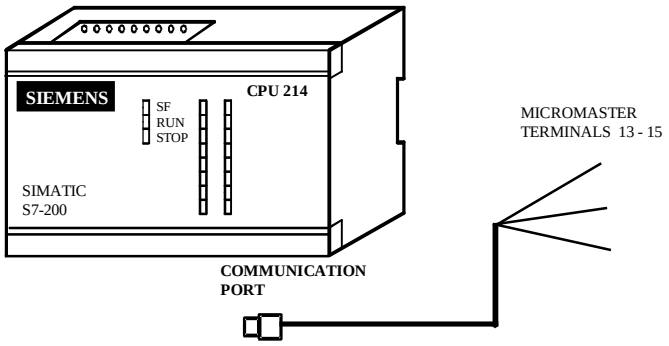


Figure 28.1

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

Hardware Requirements

This program assumes that the user has appropriately wired a motor and the MicroMaster motor drive, and that all motor and MicroMaster parameters have been set up manually. The MicroMaster must also be set to operate in remote mode (P910 = 1) with its baud rate set to 19.2Kb (P92 = 7) and MicroMaster address set to 1 (P91 = 1).

A communication cable (with a 9-pin male connector to the CPU 214, having pins 1, 3 and 8 connected) is required. The other end of the cable should be open-ended so that the 3 wires can be connected to terminals 13 - 15 on the MicroMaster (pin 1 to terminal 15, pin 3 to terminal 13, pin 8 to terminal 14).

Program Structure

MAIN	*The main program* monitors /TERM/STOP switch for Freeport/PPI communication & looks for input point rising edges as motor commands.
SBR 0	*Setup for Freeport communication* sets up Freeport mode parameters on first scan.
SBR 1	*RUN subroutine* sets up motor to run at constant speed.
SBR 2	*RAMP subroutine* sets up motor to run at variable speed.
SBR 3	*INC frequency multiplier subroutine* increases MicroMaster output frequency.
SBR 4	*DEC frequency multiplier subroutine* decreases MicroMaster output frequency.
SBR 5	*STOP subroutine* stops the motor.
SBR 6	*Outgoing message BCC calculation*.
SBR 7	*Transmit message & initiate transmit timeout timing*.
INT 0	*Transmit complete interrupt handler* turns on the receiver.
INT 1	*Transmit timeout interrupt handler* retries transmit for up to 3 timeouts.
INT 2	*Receive characters interrupt routine* performs validity checks after completion.
INT 3	*Receive timeout interrupt handler* retries transmit for up to 3 timeouts.

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

Program Description

The CPU 214 communicates with a SIMOVERT MicroMaster motor drive to start, stop, and vary the output frequency to the motor. Communication is via the S7-200 Freeport mode using the USS 5-Word Protocol. An input simulator is used to initiate commands to the motor drive. The program monitors the RUN/TERM switch and sets the Freeport mode control byte(SMB30) protocol selection field accordingly. It also monitors input points for motor commands:

I0.0 rising edge	Run motor at constant frequency as of last command.
I0.2 rising edge	Run motor at variable frequency starting at frequency of last command. Frequency can be increased/decreased using I0.6 & I0.7.
I0.4 rising edge	Stop the motor.
I0.5 level	1x or 2x multiplier for amount of change in frequency. I0.5: 0 1x, 1 2x.
I0.6 rising edge	Increase frequency of motor by 1x or 2x the frequency delta(50 in this example).
I0.7 rising edge	Decrease frequency of motor by 1x or 2x the frequency delta(50 in this example).
I1.0 level	Motor direction. I1.0: 0 CW, 1 CCW

The program detects and reports communication errors. Transmission to the MicroMaster is timed, and it is attempted up to 3 times before aborting the specified operation. The subsequent receive operation from the MicroMaster is timed, and the transmit/receive operation is also attempted up to 3 times before aborting the specified operation. The response message from the MicroMaster is checked for validity (STX, LEN, ADR and BCC). Any error detected is shown in QB0 until completion of the next operation:

0	No error.
1	Illegal response(other than bad BCC).
2	Bad BCC.
3	Transmit timeout.
4	Receive timeout.

Although this sample program communicates with only one MicroMaster, it could be extended to use additional input points to select the address of a particular MicroMaster on a multidrop link to which a command is sent. Additionally, the basic communication structure of this program may be used to send other messages to a MicroMaster, i.e., monitoring current, torque, etc.

The program size is 342 words.

LAD (S7-MicroDOS)	STL (IEC)

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

Main Program

```
// This program is an example of S7-200 Freeport communication interfacing with
// a SIMOVERT MicroMaster motor drive.  The communication between the two
// uses the USS 5-Word Protocol.  The S7-200 is acting as the master and, in this
// example, is acting as USS Protocol address 0, while the MicroMaster is at address 1.

// For the described application you need:
//        1 SIMATIC S7-200 CPU 212 or CPU 214 with similator for inputs.
//        1 RS 485 cable with one male interface and an open end to connect to the
//        MicroMaster.
//        1 MicroMaster Drive & motor.
//        ***This example used a demo unit(drive & motor) from Herb Taylor, SE&A.***
//
// Ensure that the MicroMaster parameters are set appropriately for the motor you
// are using.  Also ensure that the MicroMaster is set for remote mode (P910 = 1), baud
// rate select is appropriately set (P092) and that the MicroMaster slave address is set
// (P091).  For this example, baud rate is 19.2Kb and MicroMaster address is 1.

//  The USS Serial Communications 5-Word Protocol as used by the MicroMaster
//  has the following 14-byte structure:
//
//        02          Start of message (STX).
//        12          Length of message (12 bytes)with the MicroMaster (from AA to
//                    BCC).
//        AA          Device number address (In this Demo the Drive is 1)
//        PKE         High byte used for parameterisation control.
//        PKE         Low byte
//        IND         High byte used for Array index word
//        IND         Low byte
//        PWE         High byte used for param. values and error codes.
//        PWE         Low byte
//        PZD1        High byte used for control/status word
//        PZD1        Low byte
//        PZD2        High byte used for main set-point and return values.
//        PZD2        Low byte
//        BCC         Block Checksum
//
//        ***Note that this example uses only data words
//        ***PZD1 and PZD2 for ON/OFF/Speed changes.

// Structure of the program:
//        MAIN    The main program.
//        SBR 0   Setup for Freeport communication.
//        SBR 1   RUN subroutine.
//        SBR 2   RAMP subroutine.
//        SBR 3   INC frequency multiplier subroutine.
//        SBR 4   DEC frequency multiplier subroutine.
//        SBR 5   STOP subroutine.
//        SBR 6   Outgoing message BCC calculation.
//        SBR 7   Transmit message & initiate transmit timeout timing.
//        INT 0   Transmit complete interrupt handler.  Turns on receiver.
//        INT 1   Transmit timeout interrupt handler.
//        INT 2   Receive characters interrupt routine.
//        INT 3   Receive timeout interrupt handler.
```


SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

```
// Memory Usage:

//      VB99            Transmit msg. length
//      VB100-VB113     Transmit buffer.
//      VB114-VB127     Receive buffer.
//      VW200           Default  frequency multiplier.  Initially = 5461.
//                      Frequency = (5461/16384) x P094
//      VW202           Frequency multiplier increment/decrement amount.  Initially = 50.
//      VW204           Transmit retry count.  Set to 3 at init. & on successful XMT.
//      VW206           Receive retry count.  Set to 3 at init. & on successful resp. receive.
//      VW208           No. chars. to receive in a message.  For USS 10 = 14.
//      VB210           Status of last attempted operation (also shown in QB0):
//                              0       Operation OK.
//                              1       Illegal response from MicroMaster
//                                      (other than bad BCC).
//                              2       Bad BCC.
//                              3       Transmit timeout.
//                              4       Receive timeout.
//      VD211           Receive buffer address pointer
//      VW215           Accumulating BCC of msg. being received in lower byte.
//      VB217-VB218     Scratch area

// I/O Usage:

//      I0.0            RUN.  Turn motor ON at current value of frequency multiplier.
//                      Direction is taken from I1.0.  Affects operation only after a STOP.
//                      Rising edge operation.
//      I0.2            RAMP.  Turn motor ON at current value of frequency multiplier.
//                      Direction is taken from I1.0. Affects operation only after a STOP.
//                      Rising edge operation.
//      I0.4            STOP.  Stops motor and allows subsequent RUN/RAMP commands.
//                      Rising edge operation.
//      I0.5            Specifies multiplier(1x or 2x) for frequency multiplier increment/
//                      decrement amount.  Level operation:  0     1x, 1     2x.
//      I0.6            Increment frequency multiplier by value in VW202x multiplier (I0.5).
//                      If motor is in RAMP, speed is immediately increased.  Rising edge
//                      operation.
//      I0.7            Decrement frequency multiplier by value in VW202x multiplier (I0.5).
//                      If motor is in RAMP, speed is immediately decreased.  Rising edge
//                      operation.
//      I1.0            Specifies motor direction.  Level operation:  0     CW, 1     CCW.

// First scan maintenance activities
```

```
    | SM0.1                                                  0
1   |--|  |-------------------------------------------------( CALL   )
```

```
LD      SM0.1
CALL    0               // Set up Freeport comm. buffers, settings, etc.
```


SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

// Follow STOP-TERM- mode switch for Freeport or PPI mode selection

```
    | SM0.7                                                  SM30.0
2   |-|   |-------------------------------------------------(       )
    |
```

LD	**SM0.7**
=	**SM30.0**

// Check for any commands

```
    | I0.4                                                     5
3   |-|   |---| P |-------------------------------------------( CALL  )
    |                                                 |
    |                                                 |        M0.0     K1
    |                                                 |-------------(   S   )
    |                                                 |
    |                                                 |        M0.1     K1
    |                                                 |-------------(   R   )
    |
    | I0.0            M0.0                                     1
4   |-|   |---| P |---|   |-------------------------------------( CALL  )
    |                                                 |
    |                                                 |        M0.0     K2
    |                                                 |-------------(   R   )
    |
```

LD	**I0.4**	**// STOP**
EU		**// Edge?**
CALL	**5**	**// Send msg to MicroMaster**
S	**M0.0,1**	**// Enable subsequent RUN or RAMP cmd.**
R	**M0.1,1**	**// No longer in**
LD	**I0.0**	**// RUN**
EU		**// Edge?**
A	**M0.0**	**// Enabled?**
CALL	**1**	**// Send msg to MicroMaster**
R	**M0.0,2**	**// Disable subsequent RUN or cmd. & no**

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

```
     I0.2          M0.0                          M0.0     K1
5  ─┤   ├──┤ P ├──┤   ├──────────────┬──────────(   R    )
                                     │
                                     │                2
                                     ├──────────( CALL   )
                                     │
                                     │           M0.1     K1
                                     └──────────(   S    )

     I0.6                                          3
6  ─┤   ├──┤ P ├──────────────────────────────( CALL   )

     I0.7                                          4
7  ─┤   ├──┤ P ├──────────────────────────────( CALL   )

8  ──────────────────────────────────────────( MEND   )
```

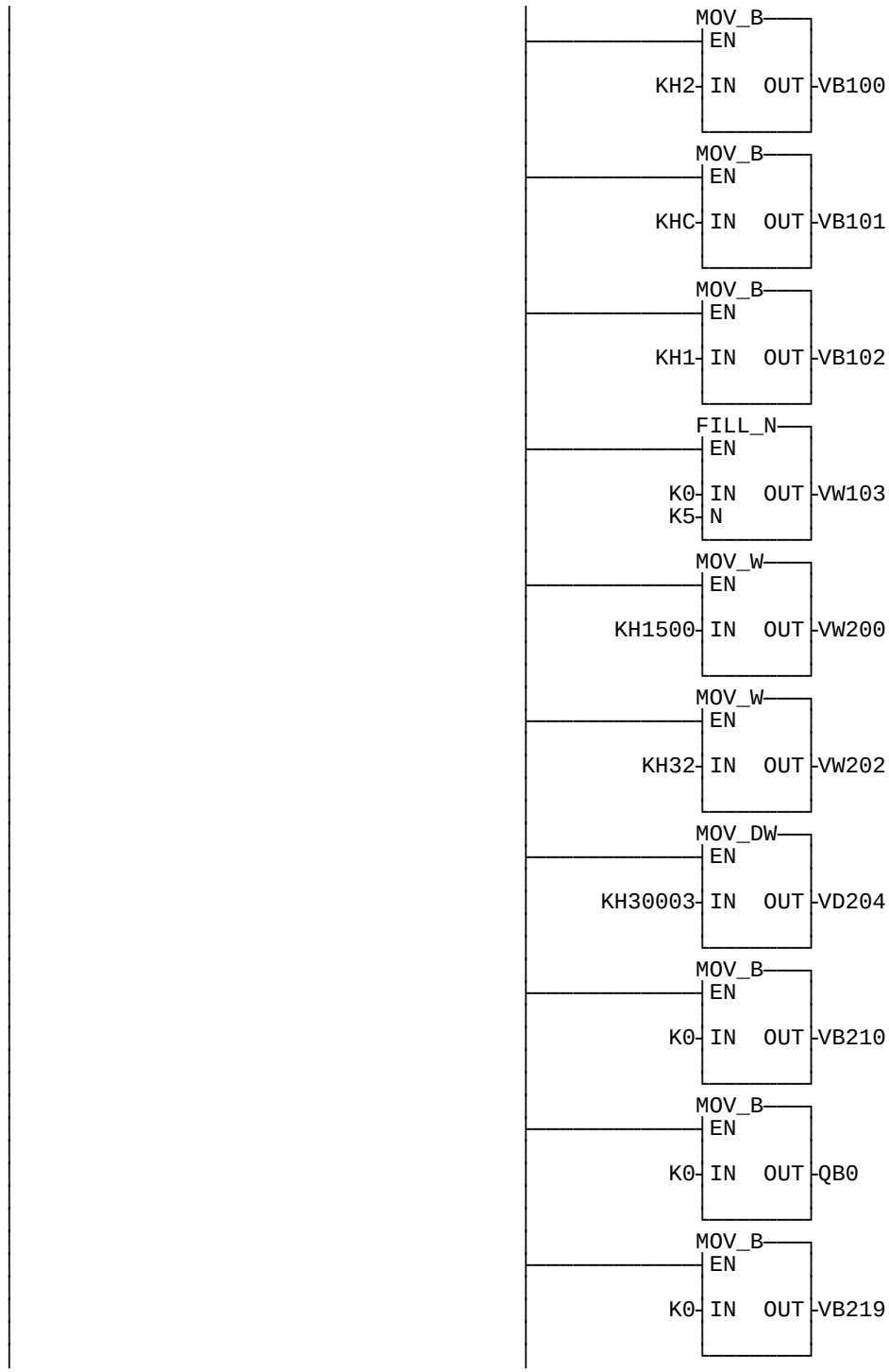
```
LD      I0.2      //
EU                // Edge?
A       M0.0      // Enabled?
R       M0.0,1    // Disable subsequent RUN or cmd.
CALL    2         // Send msg to MicroMaster
S       M0.1,1    // Show now in state

LD      I0.6      // Increment freq.?
EU
CALL    3         // ++ speed

LD      I0.7      // Decrement freq.?
EU
CALL    4         // -- speed

MEND
```

Subroutines

// Initialization subroutine - executed on 1st scan only

```
 SBR:  0
     SM0.0                                    MOV_B
10  ─┤   ├──────────────────────────┬──────────┤EN
                                     │
                                     │     KH44─IN   OUT─SMB30
                                     │
                                     │           MOV_B
                                     ├──────────┤EN
                                     │
                                     │      KHE─IN   OUT─VB99
```


SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.



SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

```
                                                  M0.0      K1
                                                 (    S    )

                                                 (   ENI   )

11                                               (   RET   )
```

```
SBR     0              // Initializes XMT buffer and sets up comm. settings
LD      SM0.0          // Always On Bit
MOVB    16#44,SMB30    //  19.2kb, 8, Even, 1 when Freeport
MOVB    16#0E,VB99     //  XMT length
MOVB    16#02,VB100    //  STX
MOVB    16#0C,VB101    //  LEN
MOVB    16#01,VB102    //  ADR(MicroMaster is at address 1)
FILL    0,VW103,5      // Clear ALL 5 data words; can change later
MOVW    16#1500,VW200  // 1/3 max. frequency multiplier
MOVW    16#32,VW202    // Freq. multiplier inc/dec of 50
MOVD    16#00030003,VD204 // Set xmit & rcv retry counts to 3
MOVB    0,VB210// Clear operation status indicators.
MOVB    0,QB0
MOVB    0,VB219// S7-200 address
S       M0.0,1         // Allow RUN or  RAMP or RAMP( & direction change)
ENI                    // Enable user interrupts
RET
```

// Subroutine to handle running the motor; direction taken from I1.0.

```
SBR:  1
      SM0.0
13                                                MOV_B
                                                 EN
                                          KH5 IN   OUT VB109

                                                  MOV_B
                                                 EN
                                          KH7F IN   OUT VB110

                                                  MOV_W
                                                 EN
                                          VW200 IN   OUT VW111
```

```
SBR     1              // Run motor
LD      SM0.0          // Always On Bit
MOVB    16#05,VB109    // Set up STW for
MOVB    16#7F,VB110    // cw inching
MOVW    VW200,VW111    // Set frequency
```


SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

```
     I1.0                                                  V109.3
14  ─┤   ├──┬──────────────────────────────────────────────(       )
     │      │                                              V109.4
     │      └─┤NOT├──────────────────────────────────────(       )
     SM0.0                                                   6
15  ─┤   ├──────────────────────────────────┬──────────────( CALL  )
     │                                      │                7
     │                                      └──────────────( CALL  )
16  ─────────────────────────────────────────────────────(  RET  )
```

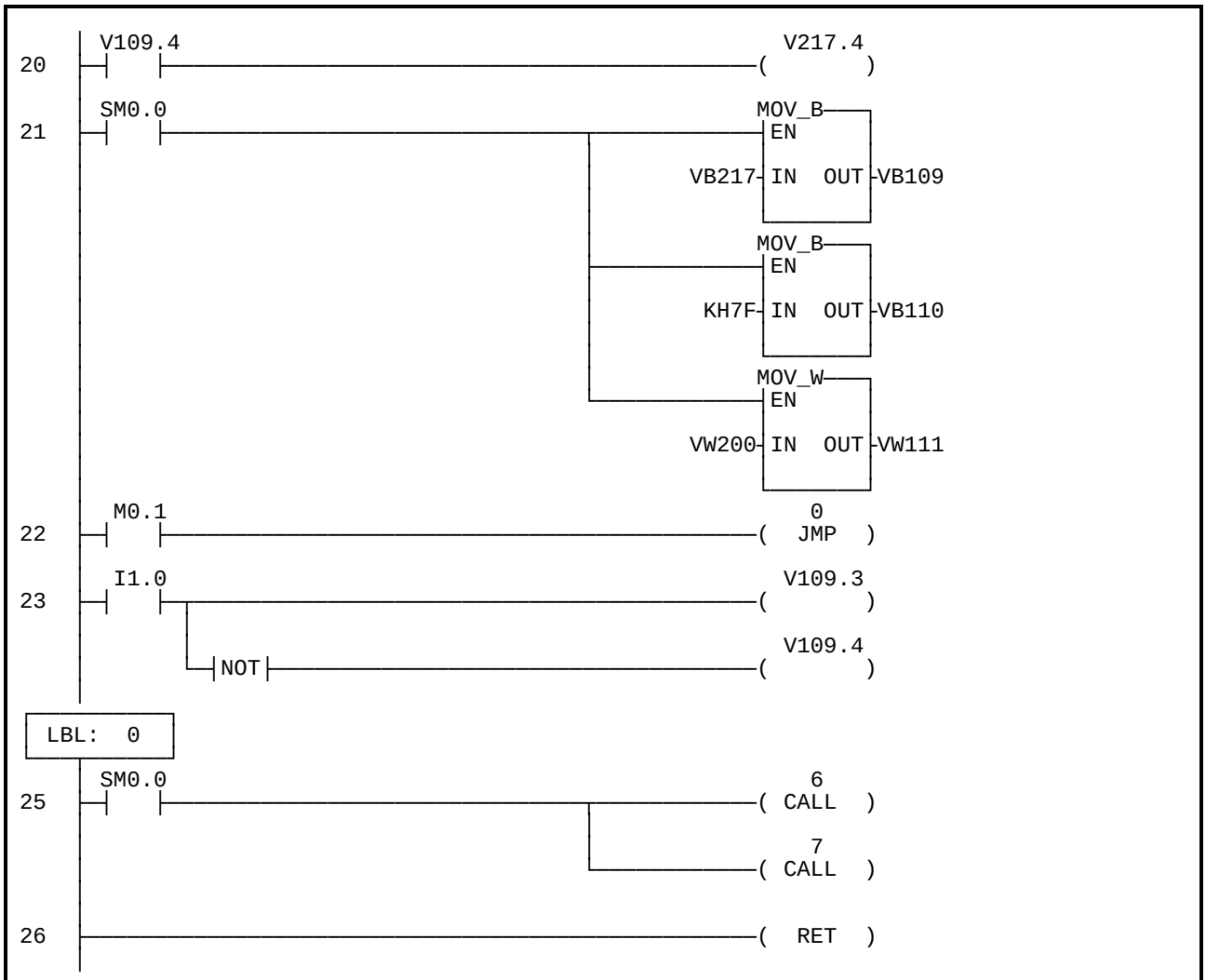
LD	**I1.0**	**// Set up direction**
=	**V109.3**	**// for start**
NOT		
=	**V109.4**	
LD	**SM0.0**	
CALL	**6**	**// Calculate BCC**
CALL	**7**	**// Initiate/Time XMT**
RET		

// Subroutine to handle running/ramping the motor. Called from seeing I0.2 edge and also
// from seeing +/- ramp modification to motor frequency when motor is running. Direction
// taken from I1.0.

```
 SBR:  2
     SM0.0                                                 MOV_B──┐
18  ─┤   ├──────────────────────────────────────────────────┤EN     │
     │                                                     KH4─┤IN  OUT├─VB217
     │                                                     └──────┘
     V109.3                                                V217.3
19  ─┤   ├──────────────────────────────────────────────────(       )
```

SBR	**2**	**// Run motor**
LD	**SM0.0**	**// Always On Bit**
MOVB	**16#04,VB217**	**// Set up command byte**
LD	**V109.3**	**// Preserve any**
=	**V217.3**	**// previous motor**

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.




```
LD      V109.4        // direction
=       V217.4        // indicators

LD      SM0.0
MOVB    VB217,VB109   //  Set up STW for
MOVB    16#7F,VB110   //  run.
MOVW    VW200,VW111   // Set frequency

LD      M0.1          // Check state to see if direction specification allowed
JMP     0             // Only allowed on  after STOP; not in
                      // If in, previous direction is preserved
LD      I1.0          // Set up direction
=       V109.3        // for run
NOT
=       V109.4

LBL     0
LD      SM0.0
CALL    6             // Calculate BCC
CALL    7             //  Initiate/Time XMT
RET
```


SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

// Subroutine to handle increasing motor frequency. Freq. incremented by value
// in VW202. If I0.5 is ON, it is doubly incremented. On overflow, freq. set to
// 32767.

```
 SBR:  3
    |
    |  SM0.0                                                  ADD_I
28  |-|   |-------------------------------------------------|EN
    |                                                       |
    |                                                 VW202-|IN1  OUT|-VW200
    |                                                 VW200-|IN2
    |                                                       |
    |  I0.5                                                   ADD_I
29  |-|   |-------------------------------------------------|EN
    |                                                       |
    |                                                 VW202-|IN1  OUT|-VW200
    |                                                 VW200-|IN2
    |                                                       |
    |   VW200       K16384                                    MOV_W
30  |---------| >= W |-------------------------------------|EN
    |                                                       |
    |                                                K16384-|IN   OUT|-VW200
    |                                                       |
    |                                                       |
```

SBR	**3**	**// Increase freq.**
LD	**SM0.0**	**// Always On Bit**
+I	**VW202,VW200**	**// by +factor**
LD	**I0.5**	**// Double it?**
+I	**VW202,VW200**	

LDW>= VW200,16384 // Did we overflow max. +?

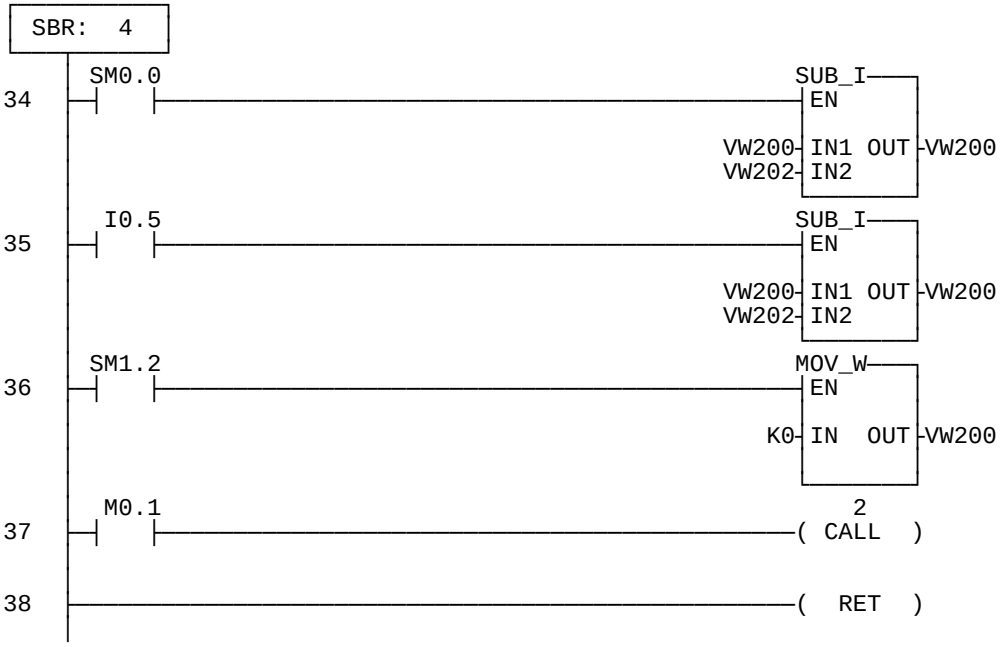
MOVW	**16384,VW200**	**// If so, set to max. +**

```
    |  M0.1                                                   2
31  |-|   |-------------------------------------------------( CALL  )
    |
32  |-------------------------------------------------------(  RET  )
    |
```

LD	**M0.1**	**// If in,**
CALL	**2**	**// send msg. w/ increased freq.**
RET		

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

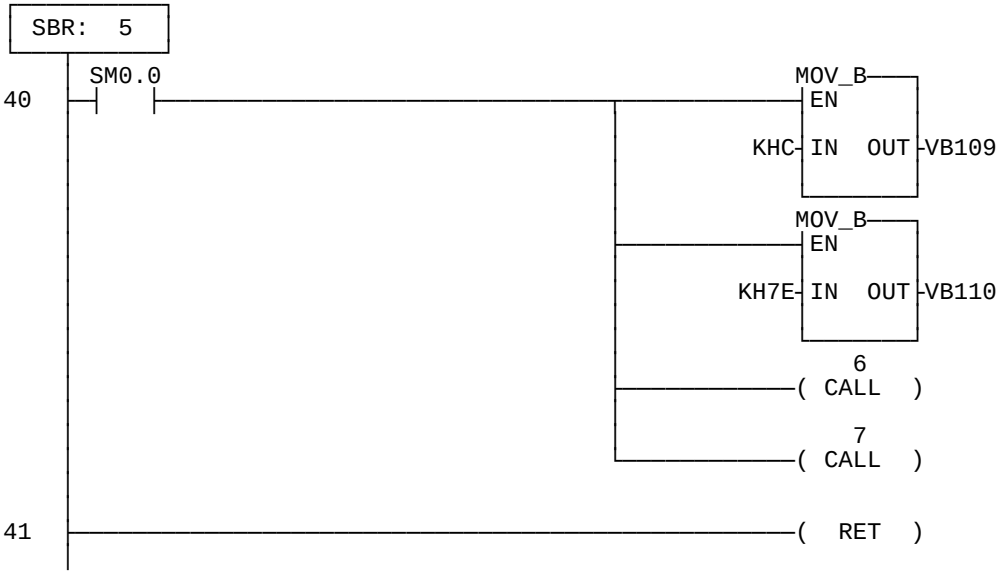
// Subroutine to handle decreasing motor frequency. Freq. decreased by value
// in VW202. If I0.5 is ON, it is doubly decremented. On underflow, freq. set to
// 0.



SBR	**4**	**// Decrease freq.**
LD	**SM0.0**	**// Always On Bit**
-I	**VW202,VW200**	**// by -factor**
LD	**I0.5**	**// Double it?**
-I	**VW202,VW200**	
LD	**SM1.2**	**// Did we underflow 0?**
MOVW	**0,VW200**	**// If so, set to 0.**
LD	**M0.1**	**// If in,**
CALL	**2**	**// send msg. w/ decreased freq.**
RET		

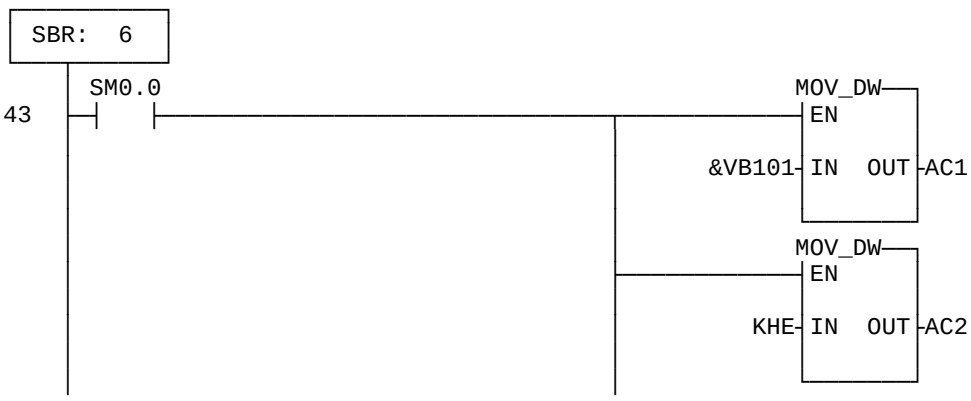
SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

// Subroutine to stop motor.

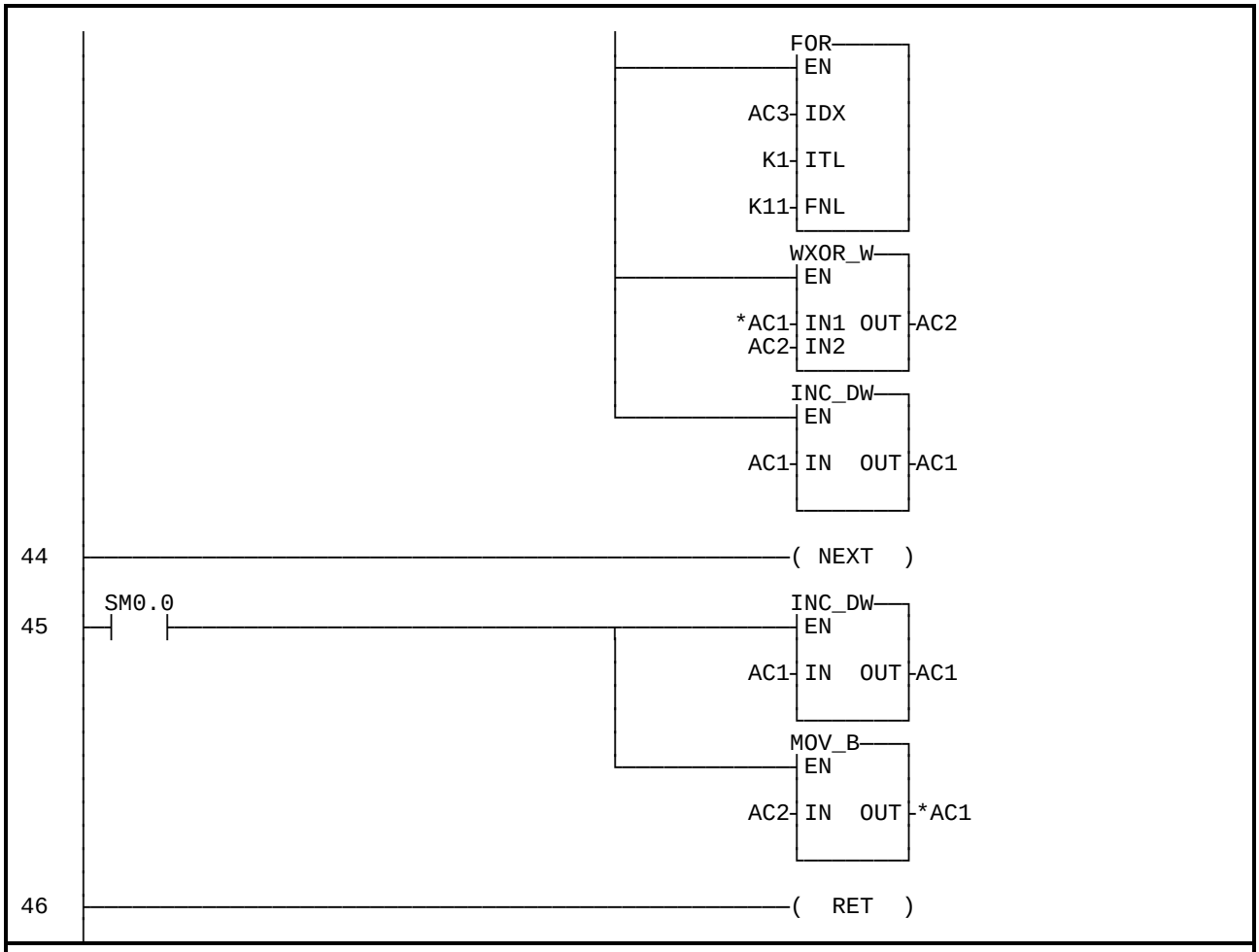


SBR	**5**	**// Stop motor**
LD	**SM0.0**	**// Always On Bit**
MOVB	**16#0C,VB109**	**// Set up STW for**
MOVB	**16#7E,VB110**	**// stop.**
CALL	**6**	**// Calculate BCC**
CALL	**7**	**// Initiate/time XMT**
RET		

// Calculates the XOR block-checksum for the msg. and places it in the buffer.
// HEX Msg. is: 02|0C|ADR|BYTE 1|...|BYTE 10|BCC
// AC1 set to LEN byte(0C) of msg. on entry
// AC1 points to BCC on exit
// BCC has been put into msg. on exit
// AC2 contains BCC on exit

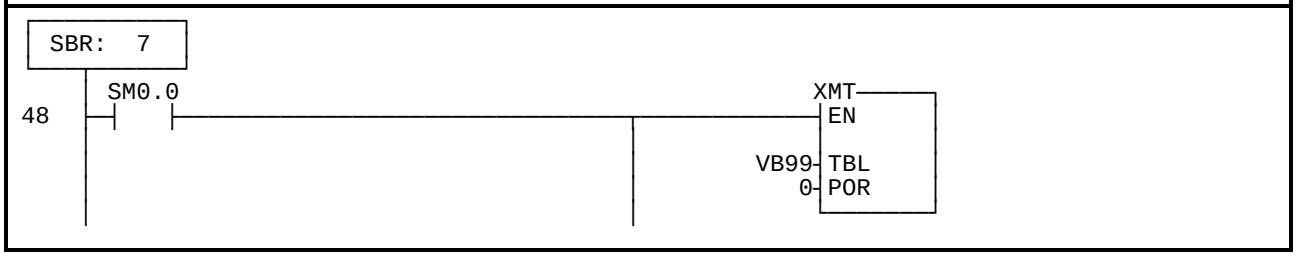


SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

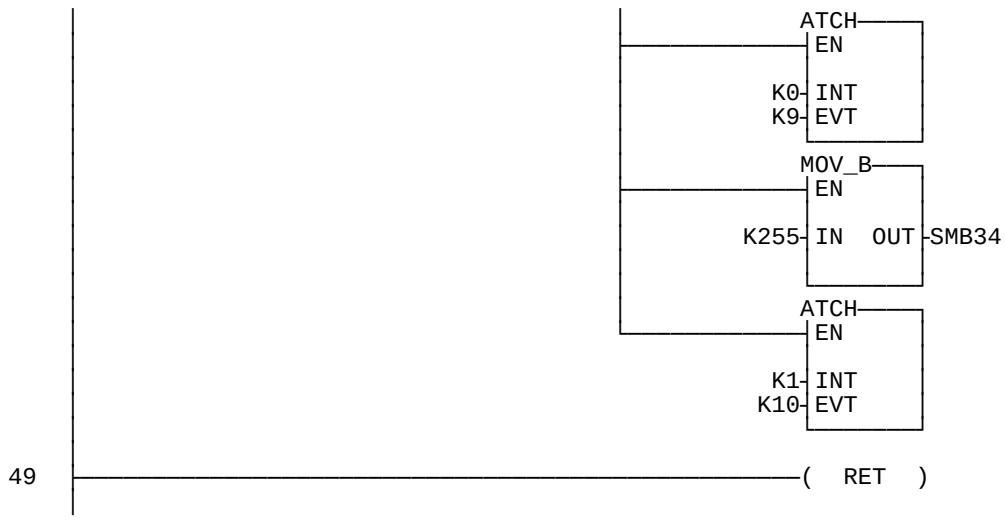



```
SBR     6          // Calculates BCC for USS 5-Word
LD      SM0.0      // Always On Bit
MOVD    &VB101,AC1 //  Set buffer address pointer
MOVD    16#0E,AC2  // 2 XOR 12 - 1st two bytes of msg
FOR     AC3, 1, 11 // Calc. BCC for next 11 chars.
XORW    *AC1,AC2
INCD    AC1        //  Increment address pointer
NEXT
LD      SM0.0      // Always On Bit
INCD    AC1        // Advance pointer to BCC position
MOVB    AC2,*AC1   // Save BCC in msg.
RET
```

// Initiate XMT of msg. & timing of XMT operation



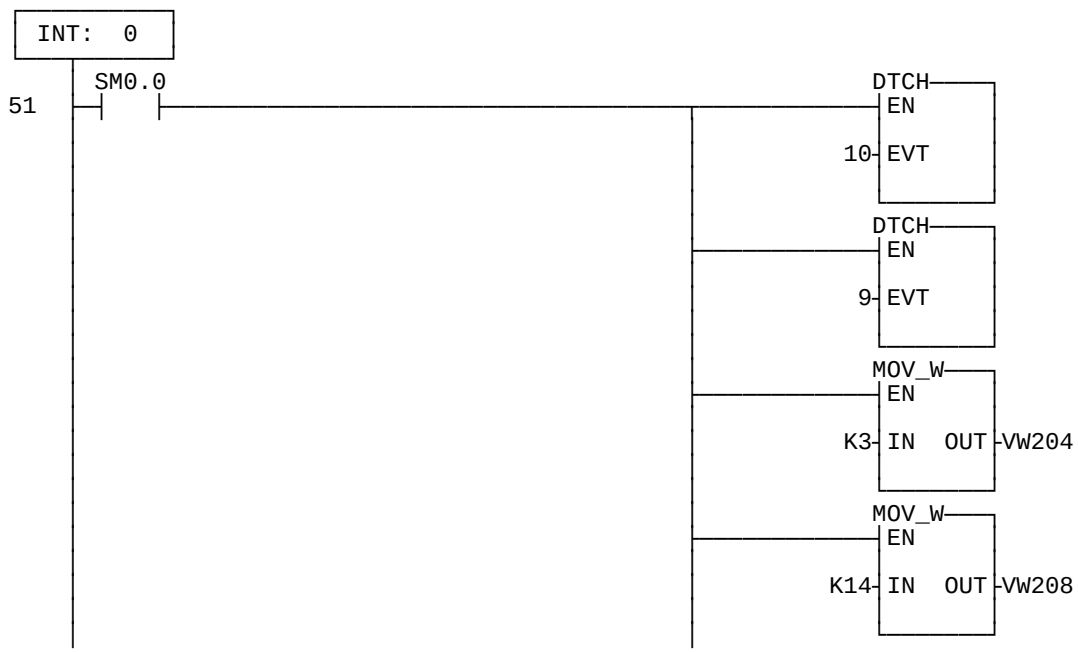
SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.



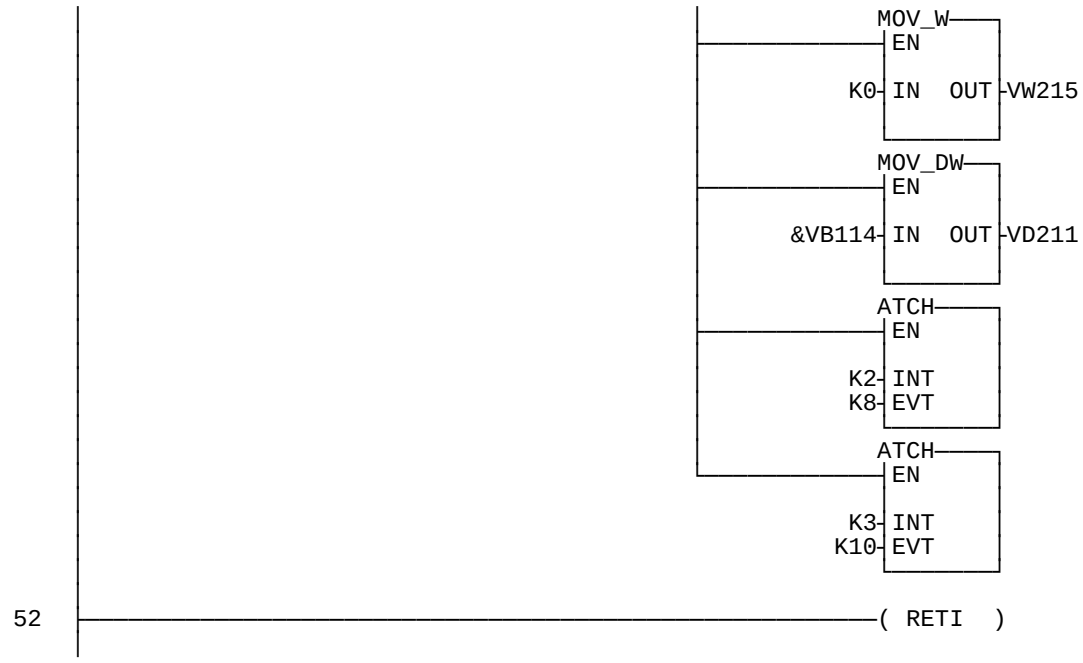
SBR	**7**	**// Initiate XMT & capture XMT interrupt**
LD	**SM0.0**	**// Always On Bit**
XMT	**VB99,0**	
ATCH	**0,9**	**// Catch XMT w/ INT 0**
MOVB	**255,SMB34**	**// Set up timing XMT operation**
ATCH	**1,10**	**// for 1/4 second. Should only take about 7ms @ 19.2Kb**
RET		

Interrupt routines

// XMT interrupt handler. Turn OFF XMT timing and set up for receiving slave response.

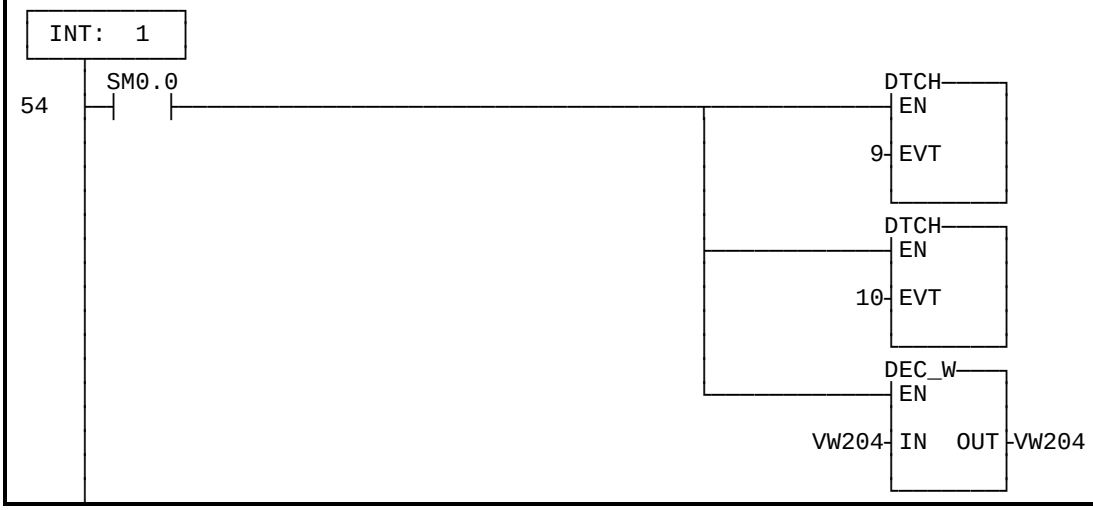


SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.



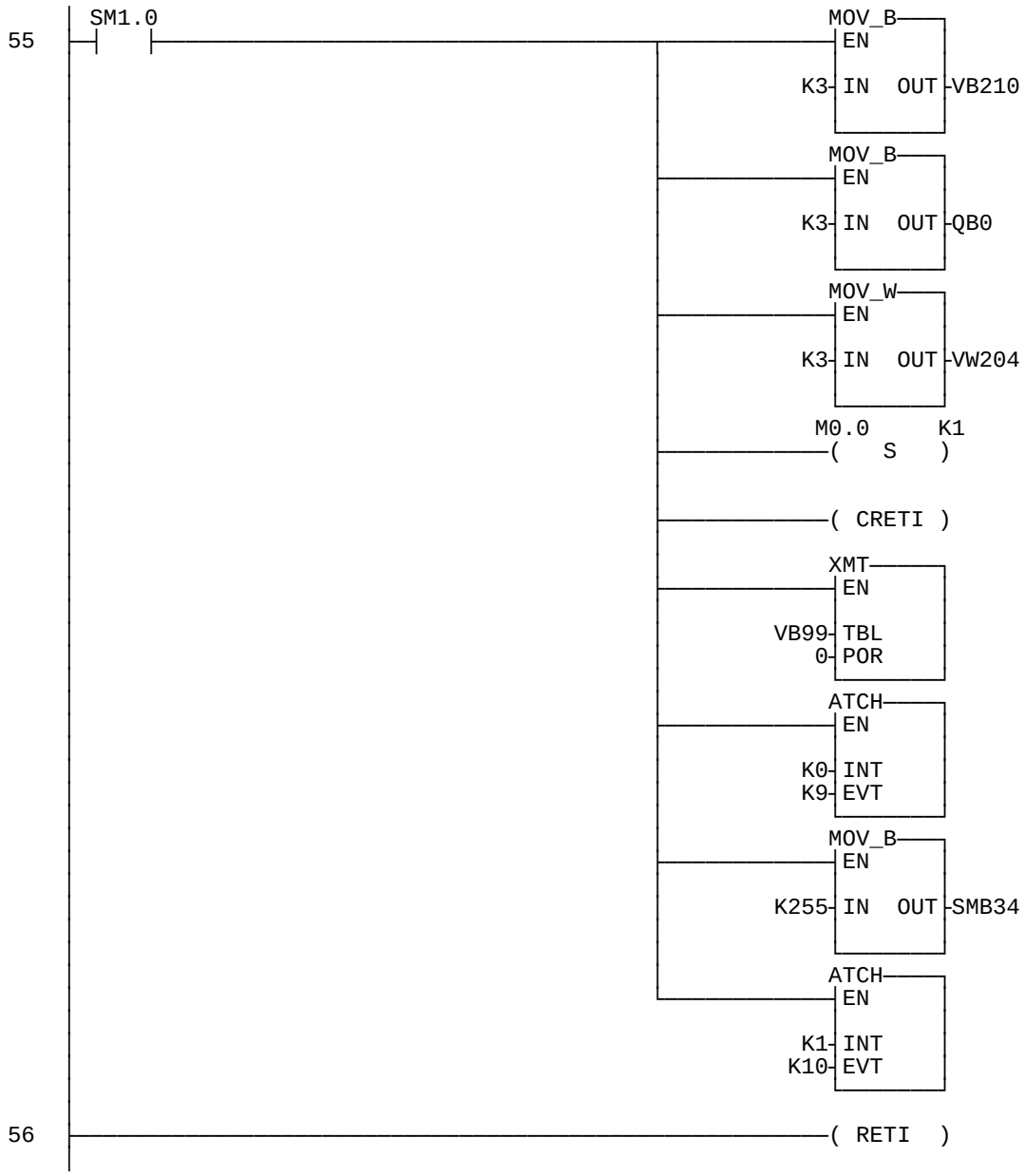

```
INT     0              // XMT interrupt handler
LD      SM0.0          // Always On Bit
DTCH    10             // Quit timing XMT
DTCH    9              // Free XMT event
MOVW    3,VW204        // Refresh XMT retry count
MOVW    14,VW208       // No. chars. to receive in response msg.
MOVW    0,VW215        // Clear BCC accumulator
MOVD    &VB114,VD211 // Set up pointer to receive buffer.
ATCH    2,8            // Catch RCV w/ INT 2
ATCH    3,10           // Start timing receive operation (for 1/4 second)
RETI
```

// This handler gains control if XMT times out. The XMT operation is
// repeated until the XMT retry count is decremented to 0.



SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

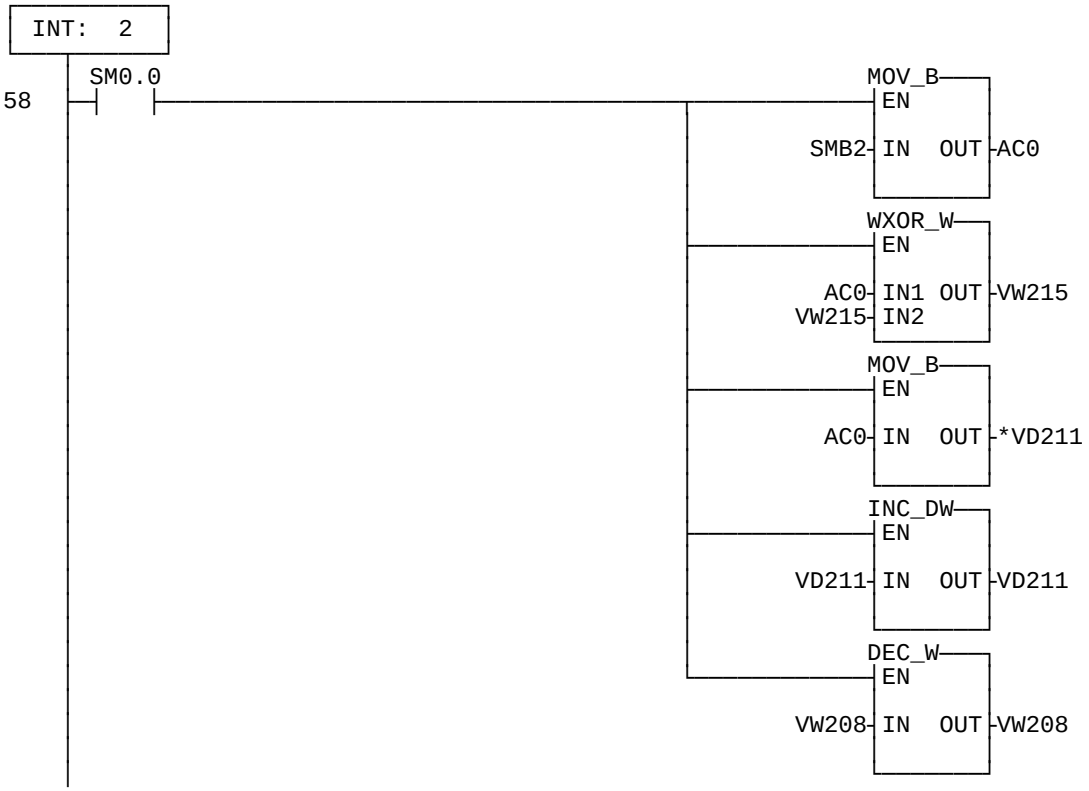
```
INT     1       // Timed interrupt 0 handler - transmit
LD      SM0.0   // Always On Bit
DTCH    9       // Free XMT event
DTCH    10      // Quit timing
DECW    VW204   // Check retry count
```



SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

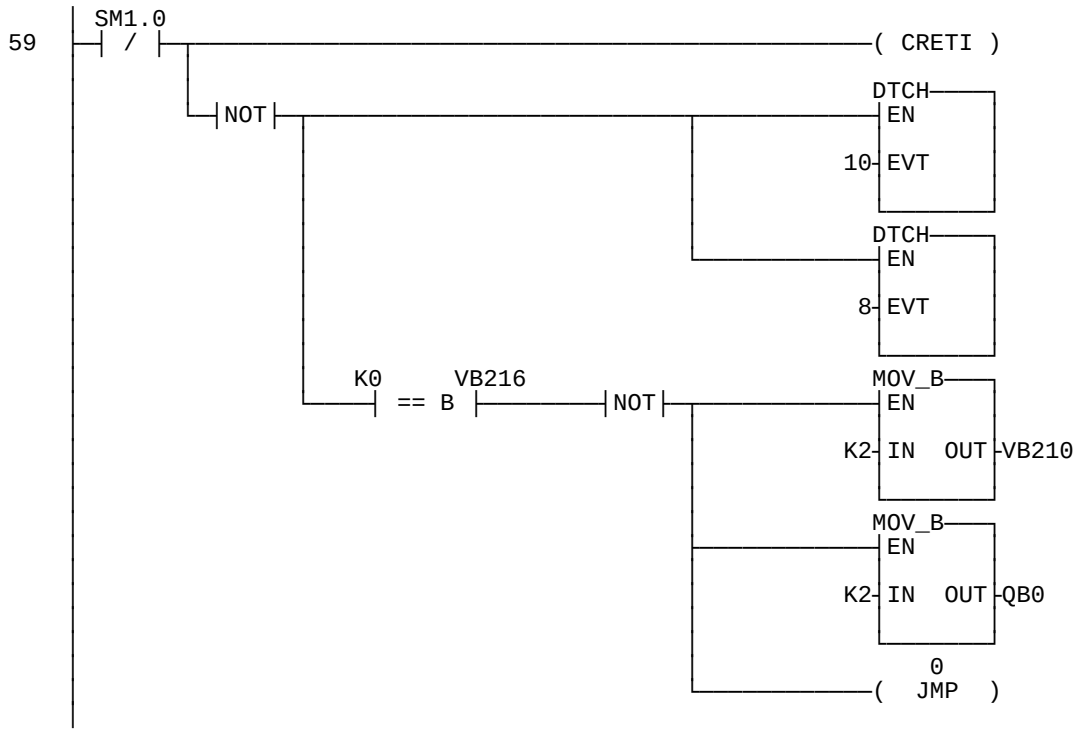
```
LD      SM1.0
MOVB    3,VB210// Indicate transmit timeout
MOVB    3,QB0
MOVW    3,VW204     // Refresh transmit retry count
S       M0.0,1      // Enable RUN, RAMP
CRETI               // Try it again.
XMT     VB99,0
ATCH    0,9         // Catch XMT w/ INT 0
MOVB    255,SMB34   // Set up timing XMT operation
ATCH    1,10        // for 1/4 second.  Should only take about 7ms @ 19.2Kb
RETI
```

```
// This handler counts characters received & performs error checks.
// If an error was detected, the operation is repeated until the receive retry count
// is decremented to 0.
```



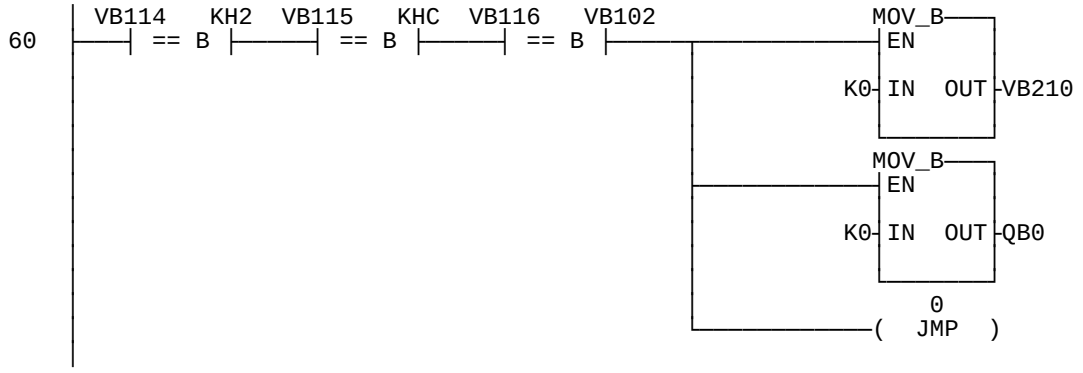

```
INT     2           // Receive char. handler
LD      SM0.0
MOVB    SMB2,AC0    // Get char. received
XORW    AC0,VW215   // Accumulate BCC
MOVB    AC0,*VD211  // Put char. received into buffer
INCD    VD211       // Advance buffer pointer
DECW    VW208       // Decrement no. chars. yet to receive
```


SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

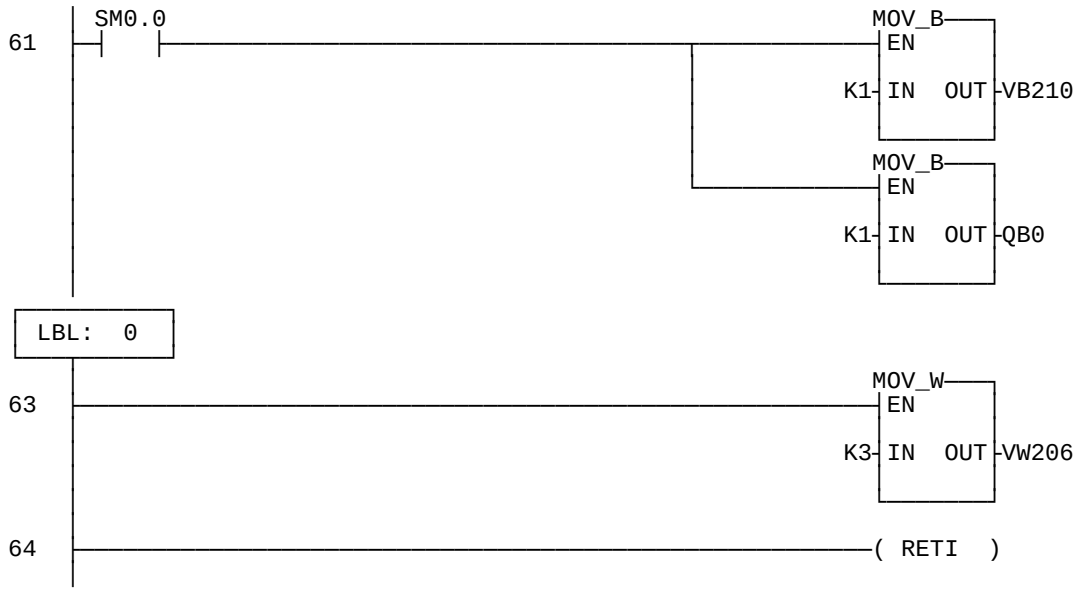



```
LDN     SM1.0     // Check for finished
CRETI
NOT
DTCH    10        // Quit timing receive
DTCH    8         // Turn off receiving
AB=     0,VB216// Check calculated BCC for 0
NOT
MOVB    2,VB210// Bad BCC operation code
MOVB    2,QB0
JMP     0         // BCC OK; check other parts of msg.
```


SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.



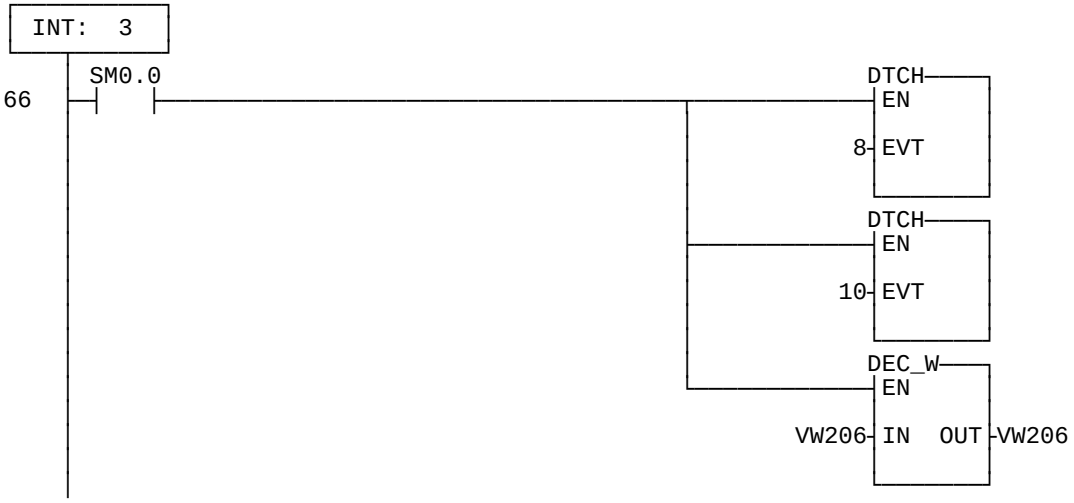

```
LDB=        VB114,16#02     // STX 1st char.?
AB=         VB115,16#0C     // Length = 12?
AB=         VB116,VB102     // Same slave that msg. was sent to?

// Any other specific checks would go here depending upon expected response

MOVB        0,VB210// Operation OK
MOVB        0,QB0
JMP         0
```



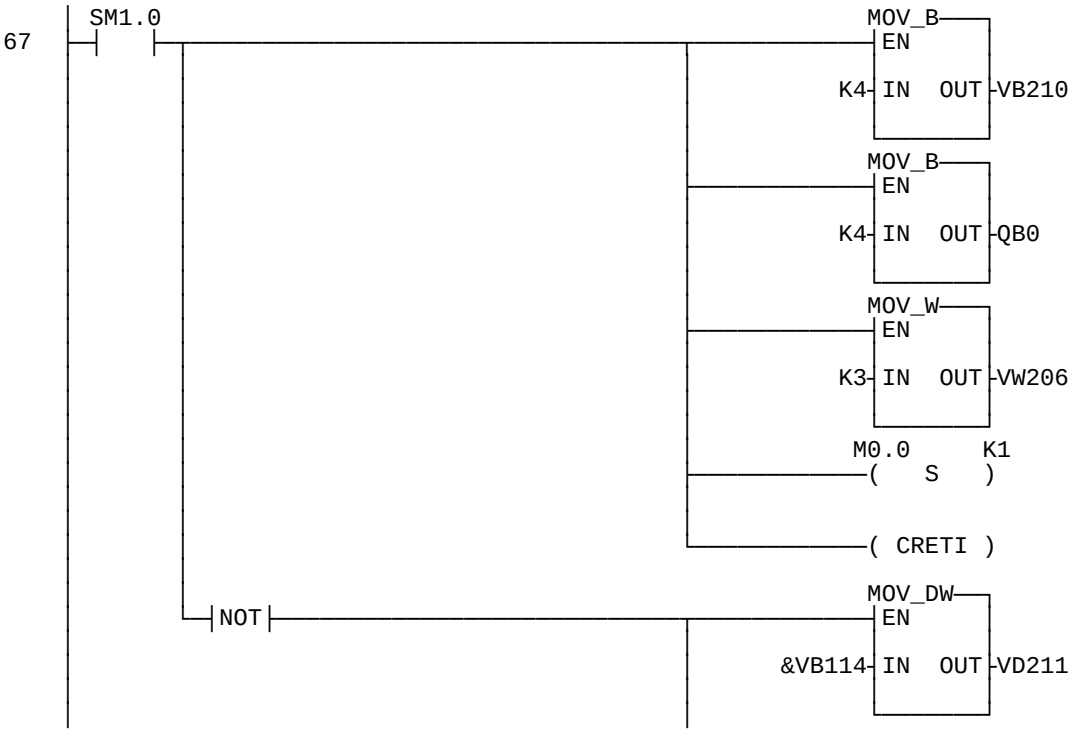

```
LD          SM0.0
MOVB        1,VB210// Something in msg. bad
MOVB        1,QB0
LBL         0
MOVW        3,VW206         // Refresh receive retry count
RETI
```


SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

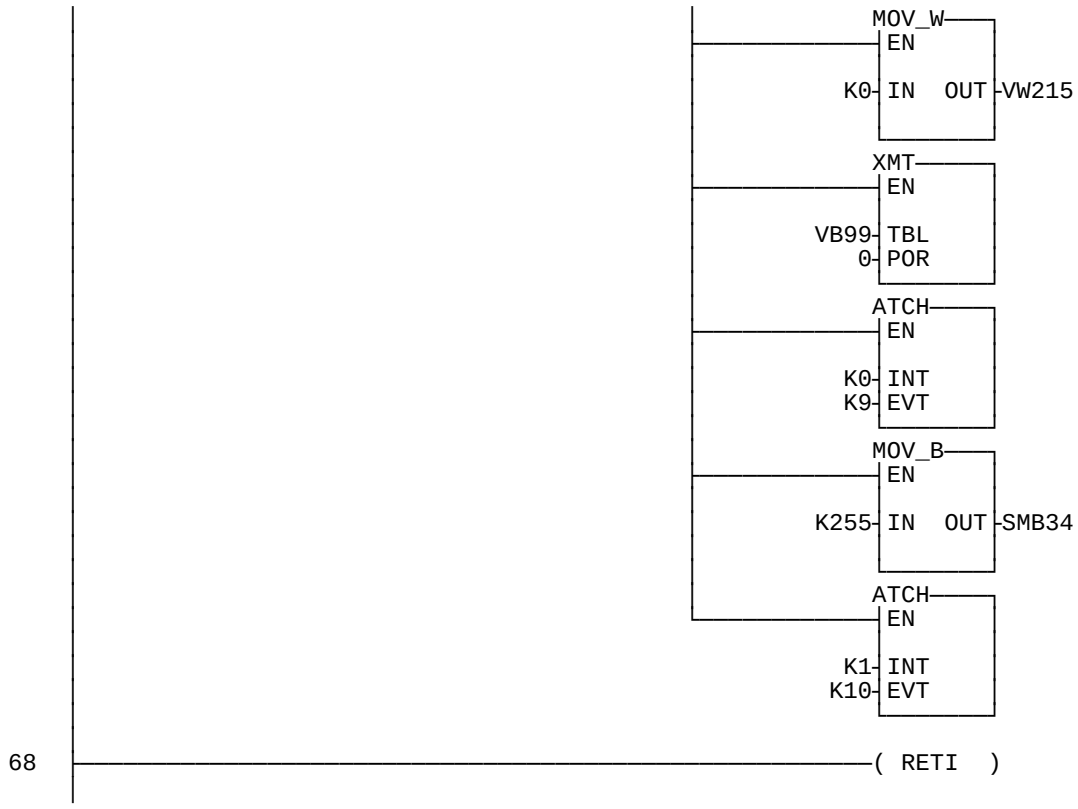
// This handler gains control if the response receive times out. The msg. is re-sent
// and another receive is attempted. The operation is repeated on timeout until the
// receive retry count is decremented to 0.



INT	**3**	**// Timed interrupt 0 handler - receive**
LD	**SM0.0**	
DTCH	**8**	**// Free receive event**
DTCH	**10**	**// Quit timing receive**
DECW	**VW206**	**// Check retry count**



SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.




```
LD      SM1.0
MOVB    4,VB210// Indicate receive timeout
MOVB    4,QB0
MOVW    3,VW206       // Refresh receive retry count
S       M0.0,1        // Enable RUN, RAMP
CRETI
NOT
MOVD    &VB114,VD211 // Set up pointer to receive buffer.
MOVW    0,VW215       // Clear BCC accumulator
XMT     VB99,0        // Re-transmit & try it again.
ATCH    0,9           // Catch XMT w/ INT 0
MOVB    255,SMB34     // Set up timing XMT operation
ATCH    1,10          // for 1/4 second.  Should only take about 7ms @ 19.2Kb
RETI
```


SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

Conversion Notes

To Convert from IEC STL to S7-Micro/DOS STL:

- Add a 'K' before all non-Hex numerical constants (i.e. 4 K4)
- Replace '16#' with 'KH' for all Hex constants (i.e. 16#FF KHFF)
- Commas denote field divisions. Use arrow or TAB keys to toggle between fields.
- To convert an S7-Micro/DOS STL program to LAD form, every network must begin with the word 'NETWORK' and a number. Each network in this Application Tip program is designated by a number on the ladder diagram. Use the INSNW command under the EDIT menu to enter a new network. The MEND, RET, RETI, LBL, SBR, and INT commands each receive their own networks.
- Line-Comments denoted by '//' are not possible with S7-Micro/DOS, but Network-Comments are possible.

General Notes

The SIMATIC S7-200 Application Tips are provided to give users of the S7-200 some indication as to how, from the view of programming technique, certain tasks can be solved with this controller. These instructions do not purport to cover all details or variations in equipment, nor do they provide for every possible contingency. Use of the S7-200 Application Tips is free.

Siemens reserves the right to make changes in specifications shown herein or make improvements at any time without notice or obligation. It does not relieve the user of responsibility to use sound practices in application, installation, operation, and maintenance of the equipment purchased. Should a conflict arise between the general information contained in this publication, the contents of drawings or supplementary material, or both, the latter shall take precedence.

Siemens is not liable, for whatever legal reason, for damages or personal injury resulting from the use of the application tips.

All rights reserved. Any form of duplication or distribution, including excerpts, is only permitted with express authorization by SIEMENS.

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

SIMATIC

S7-200 Примеры

Группа	Тема
6	**Использование HSC входа S7-200 CPU 214 для записи аналогого значения при помощи аналогого/частотного преобразователя**

Краткое описание

В данном примере описано, как, при помощи частотного проеобразователя, можно использовать быстрый счетчик CPU 214 для записи аналогого напряжения. Частотный преобразователь изменяет измеряемое напряжение (0-10V) в прямоугольный импульсный сигнал (0-2000Гц). Данный сигнал подается на вход быстрого счетчика CPU 214. Импульсы подсчитываются, а затем, после истечения заданного промежутка времени, обрабатываются для определения частоты, и тогда м.б. вычислено значение напряжения.

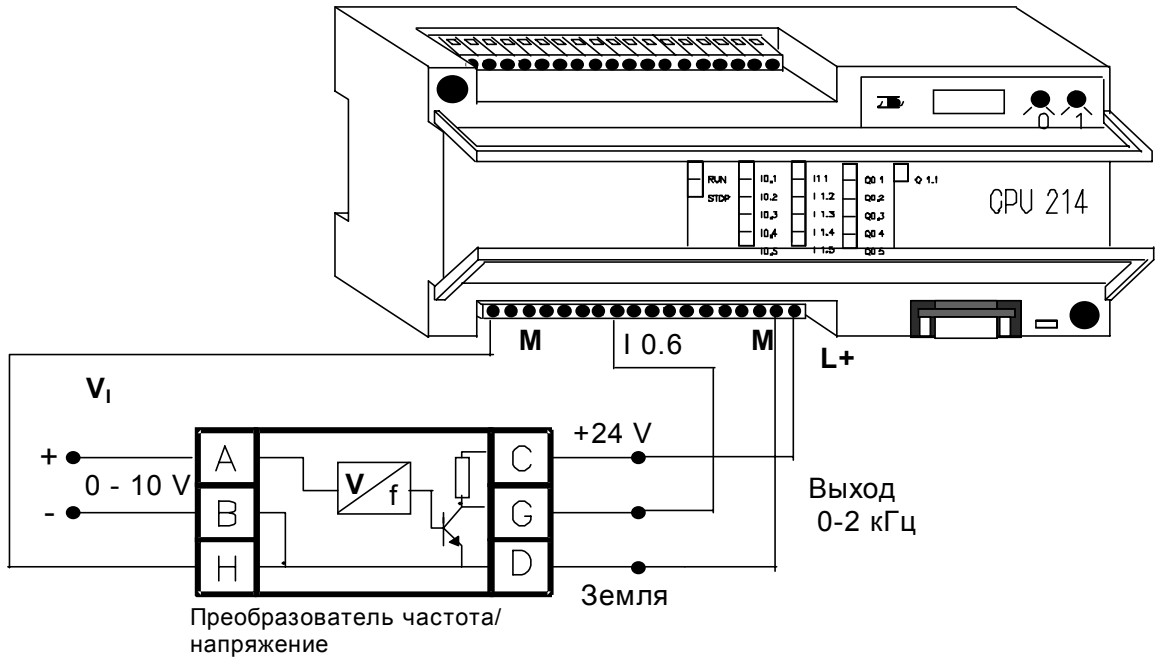


Рисунок 29.1

Требования к аппаратуре

Для данной программы используются следующие компоненты:

1 CPU CPU 214 AC/DC/RLY или DC/DC/DC
1 преобразователь частоты в напряжение - SFW 01 (Фирма Trдnkner)
(Адрес: Engineering Office Trдnkner, Industrial Area North PF38, 09618 Brand-Erbisdorf, Germany)

Технические данные:

Напряжение питания:	24V DC
Вход:	0 - 10V DC
Выход:	Квадратичный импульс, Земля - 24V
Диапазон измерений:	0 - 10V -> 0 - 2000Гц
Соотношение:	200Гц / V (линейная)

Как альтернатива м.б. использован другой преобразователь частоты в напряжение:
(Замечание: При использовании данного преобразователя, программа д.б. изменена - увеличить диапазон измерений с 2000Гц до 5000Гц.)

1 преобразователь частоты в напряжение - Модель 8507 (Calex)
(Адрес: 2401 Stanwell Drive, Concord, CA 94520, United States of America)
(510-687-4411)

Технические данные:

Напряжение питания:	24V DC
Вход:	0 - 10V DC
Выход:	Квадратичный импульс, Земля - 24V
Диапазон измерений:	0 - 10V -> 0 - 5000Гц
Соотношение:	500Гц / V (линейная)

Структура программы

Главная программа	Вызов подпрограммы SBR0 в первом цикле
SBR0	Инициализация быстрого счктчика и прерываний таймера
INT0	Программа обработки прерывания таймера для вычисления значения быстрого счетчика

Описание программы

Главная программа в первом цикле вызывает программу инициализации (специальный флаг SM0.1 равен "1" только в первом цикле).

Попрограмма SBR 0 выполняет инициализацию. Сначала она предустанавливает управляющий байт SMB47 быстрого счутчика HSC1 в десятичное значение ‘FC’. Этим задается "прямое" направление счета выбранного счетчика и разрешается обновление как предустановленного значения (конечное значение), так и текущее значение (начальное значение).

Команда ‘HDEF’ переводит HSC1 в Режим 0 (без входа сброса или запуска и без внешнего задатчика направления счета. Текущее значение сбрасывается в ноль в специальном флаге двойного слова SMD48D, а предустановленное значение с помощью произвольного значения FFFF (hex). Загружая 100 в специальный байт флагов SMB34 устанавливаем прерывание по времени 0 равным 100ms. Программа обработки прерывания 0 назначается данному прерыванию по времени и оно разрешается. Команда HSC1 запускает быстрый счетчик.

Программа обработки прерывания 0 активируется каждые 100ms, считывает значение быстрого счетчика и устанавливает его в ноль. Вычисление значения производится с учетом соотношения преобразования: 0-2кГц соответствуют 0-10V. В данном примере счктчик просто делиться на 2 и данное значение загружается в байт выходов QB0 для отображения измеряемого напряжения при помощи LED (светодиодов). Отображаемое значение соответствует 10-fold full значению, например, счетчик = 200; делим на 2 = 100; это соответствует 10.0V. Т.к. счетчик считает 200 импульсов в 100 ms, то это означает, что 2000 Гц => 10V. Значение счетчика 104 соответствует напряжению 5.2V.

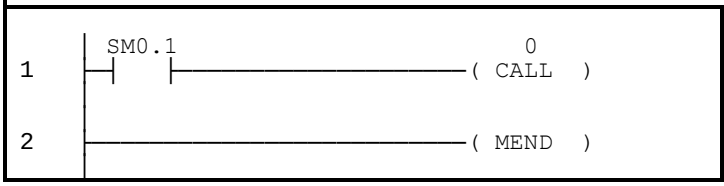
Замечание: Прерывание по времени м.б. установлено в интервале 5 - 255 ms. Однако, вычисление значения и сброс быстрого счетчика м.б. задержано с помощью флага, наколько необходимо; так что м.б. получены больший интервал опроса и соответственно большая точность; но приэтом необходимо учитывать большее время актуализации. Пример: прервание по времени установлено равным 100ms. При каждом вызове значение флага увеличивается на 1. Только, когда флаг достигнет 10, м.б. вычислено значение и сброшен быстрый счетчик. Это озанчает, что м.б. принято максимально 2000 импульсов для 10V; т.е. вычисление значения производится только при 5/1000 V, например, в 10 раз точнее, чем в вышеприведенном примере, но так же в 10 раз медленнее.

Размер программы составляет 45 слов.

LAD (S7-MicroDOS)	STL (IEC)

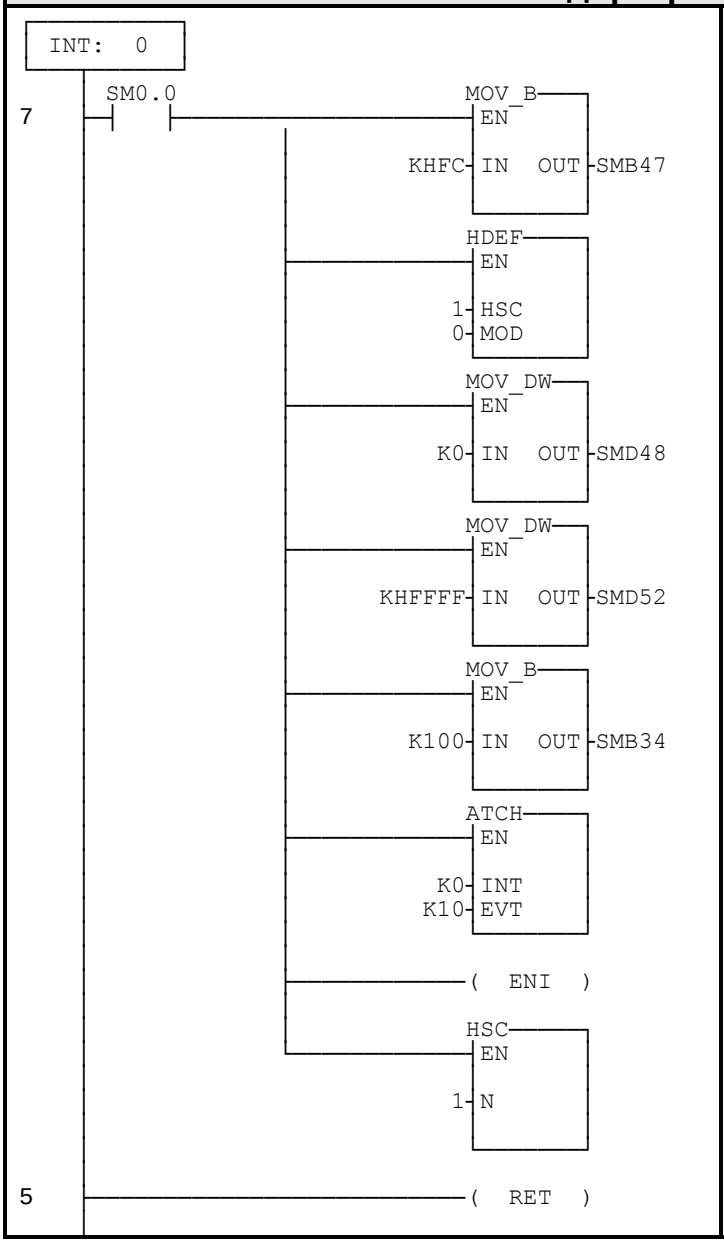
Главная программа

// Заголовок: использование DI как AI с преобразователем частоты в напряжение



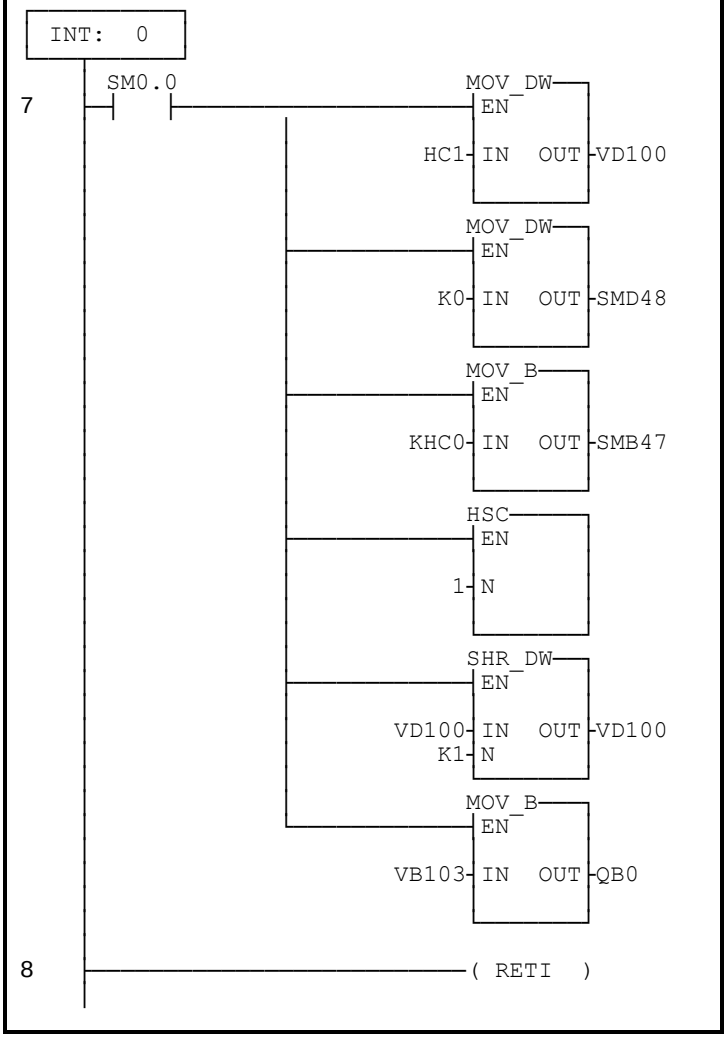

```
LD      SM0.1          // Бит первого цикла
CALL    0              // Вызов
                       // подпрограммы 0

MEND                   // Конец главной
                       // программы
```

Подпрограммы




```
SBR     0              // Начало
                       // подпрограммы 0
LD      SM0.1
MOVB    16#FC, SMB47
                       // Установить
                       // управляющий бит
                       // HSC1

HDEF    1,0            // HSC1,
                       // конфигурируется
                       // в режим 0

MOVD    0,SMD48        // Сброс текущего
                       // значения HSC1

MOVD    16#ffff, SMD52
                       // HSC1 задание
                       // предустановок

MOVB    100, SMB34     // установить прерыв
                       // по времени 0
                       // равным 100ms

ATCH    0, 10          // назначить для
                       // прерывания
                       // по времени 0
                       // (Событие 10)
                       // программу
                       // обработки
                       // прерываний 0
ENI                    // разрешить все
        // прерывания

HSC     1              // Программиро ва-
                       // ние HSC1

RET                    // Конец
        // подпрограммы 0
```

Программы обработки прерываний



INT	**0**	**// Начало // программы // прерывания 0**
LD	**SM0.0**	**// Бит асегда "1"**
MOVD	**HC1, VD100**	**// Сохранить счетчик**
MOVD	**0, SMD48**	**// Сбросить текущее // значение HSC1**
MOVB	**16#C0, SMB47**	**// записать новое // текущее значение**
HSC	**1**	**// программирова- // ние HSC1**
SRD	**VD100, 1**	**// Разделить счетчик // пополам**
MOVB	**VB103, QB0**	**// Прочесть 10-fold // полное значение**
RETI		**// Конец программы // прерываний 0**

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все прав защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

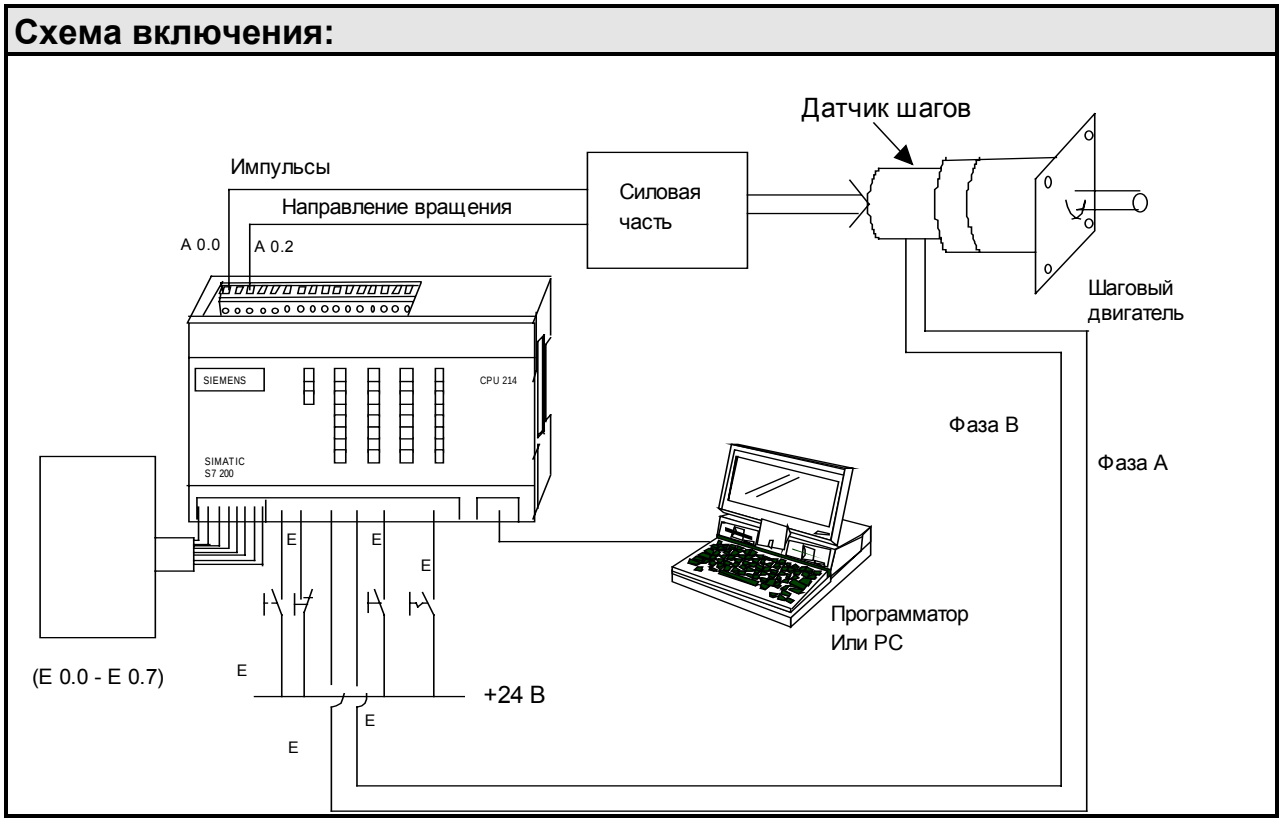
SIMATIC
S7-200 Примеры

Группа	Пример к теме
5	**Управляемое позиционирование с помощью CPU 214 с контролем и коррекцией перемещения**

Краткое описание

Этот пример применения построен на примере No. 23. Здесь производится относительное позиционирование, которое будет дополнено контролем перемещения с помощью инкрементального датчика. Для обработки сигнала от датчика используются входы быстрого счета CPU 214, которые могут обрабатывать сигналы до 7 КГц. При этом можно определить ошибку позиционирования, которая, например, возникает из-за потери шагов при превышении частоты пуска-останова. Если определена ошибка позиционирования, то производится попытка коррекции понижением частоты.

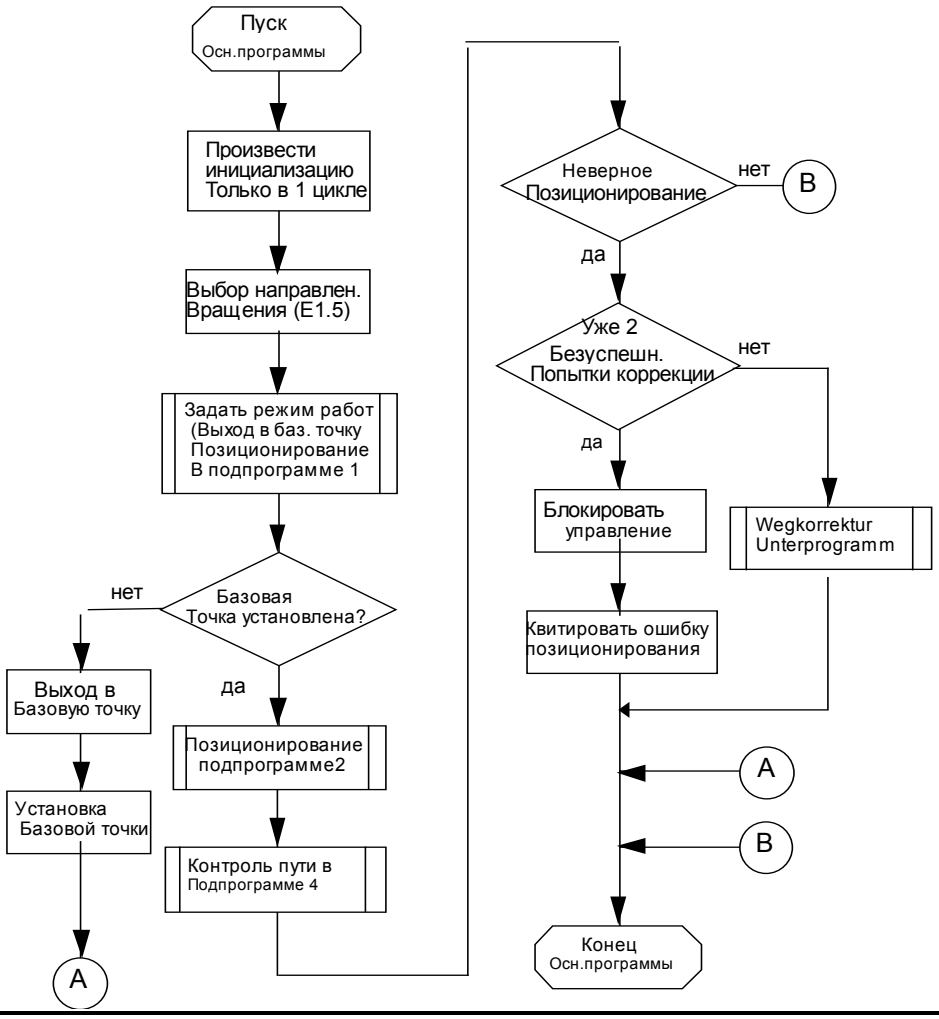
Схема включения:



Описание аппаратных средств

Кол-во	Приборы	Изготовитель/Заказной номер
1	SIMATIC S7-200 CPU 214	Siemens/ 6ES7 214 - 1AC00 - 0XB0
1	PC/PPI-кабель	Siemens/ 6ES7 901 - 3BF00 - 0XA0
1	Программатор или PC	
1	Шаговый двигатель с соответствующими силовой частью и кабелем связи	
1	Кабель для управляющих сигналов к силовой части	
1	Инкрементальный датчик 24 В	
1	Муфта	
1	Кабель для сигналов датчика	
9	Переключателя	
3	Кнопка	

Структура программы



Описание программы вкл. листинг

В первом цикле программы (SM 0.1=1), как и в примерах No. 22 и No. 23, устанавливаются важнейшие параметры. Здесь происходит инициализация быстрого счетчика HSC 2 как A/B-счетчика с внешним входом сброса. HSC 2 считает сигналы инкрементального датчика угловых шагов, который служит для проверки позиционирования. Последовательности сигналов A и B датчика подаются при этом на входы CPU E 1.2 или E 1.3.

Выбор направления движения, блокировка кнопок, выбор режима работы и процесс позиционирования происходят аналогично примеру 23 (см. описание программы). В противоположность примеру 23 позиционирование контролируется инкрементальным датчиком. По окончании выдачи импульсов запускается время ожидания T1, с тем чтобы исключить влияние колебаний скручивания муфты соединяющей двигатель с датчиком.

Сравнение Задано-Истинно

По завершении работы T1 вызывается подпрограмма 4 сравнения заданного с истинным. Если позиция оси находится внутри полосы допуска ±2 шага от заданной позиции, то позиционирование считается корректным. Если истинная позиция лежит вне заданной области, причиной чего может, например, быть потеря шагов двигателем при работе с частотой превышающей Пуск-Останов-частоту, то на выходе A 1.1 появится соответствующее сообщение.

Коррекция пути

Если опознается ошибка позиционирования, то запускается 2ое время ожидания T2. По окончании его вычисляется корректирующее число шагов из разницы между заданным и истинным значениями. Частота двигателя ри выполнении коррекции находится ниже частоты Пуска-Останова, с тем чтобы избежать потери шагов.

Прерывание попытки коррекции

Если заданная позиция не достигается после двух попыток коррекции, то управление из безопасности будет блокировано (меркер M 0.2=1). Только после нажатия кнопки квитирования на E 1.4 блокировка управления может быть снята. После этого может быть проведен заново выход в базовую точку.

Список сигналов

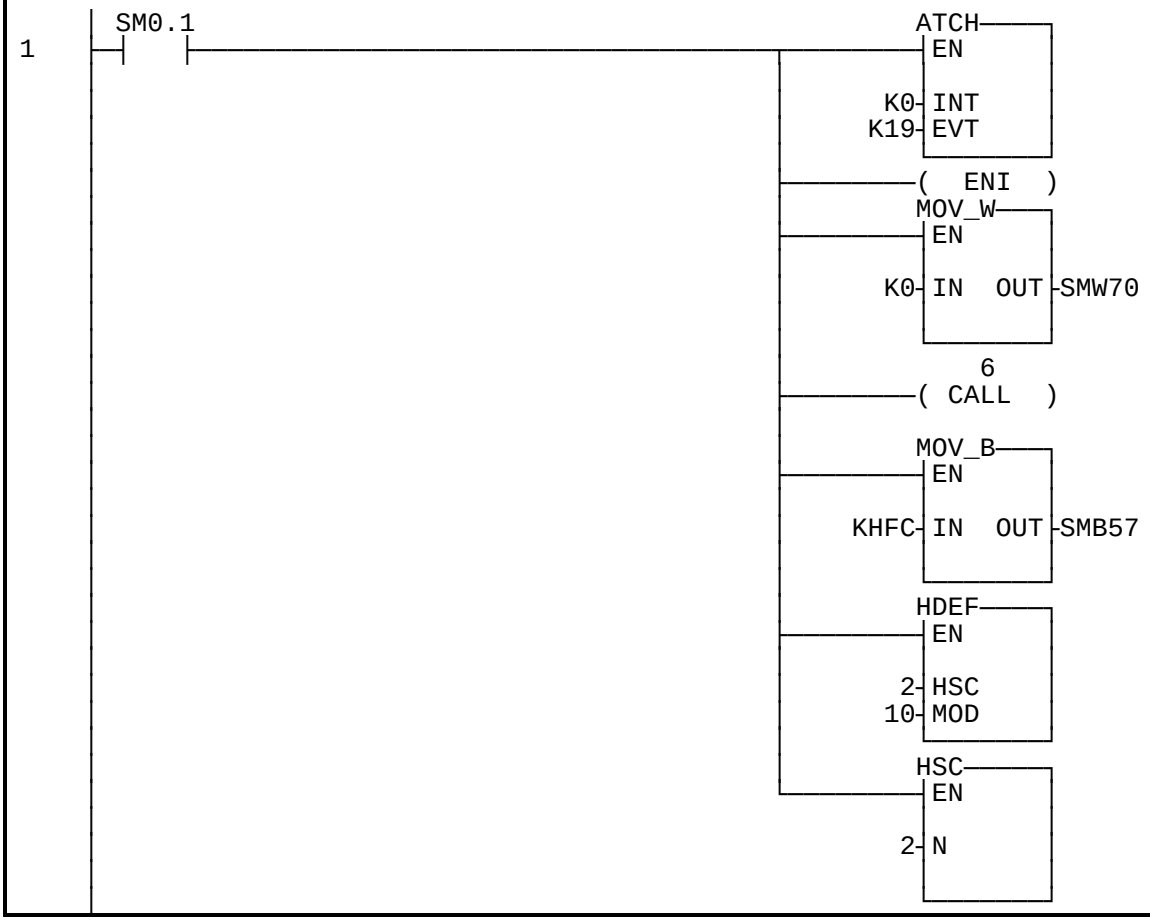
Входы:	E 0.0 - E 0.7	Угол позиционирования в градусах (двоич. код)
	E 1.0	Кнопка „ПУСК двигателя“
	E 1.1	Кнопка „ОСТАНОВ двигателя“
	E 1.2	Сигнал датчика, дорожка A
	E 1.3	Сигнал датчика, дорожка B
	E 1.4	„Устан./стереть базовую точку“ Кнопка квитир.
	E 1.5	Переключатель направления движения
Выходы:	A 0.0	Импульсный выход
	A 0.2	Сигнал направления движения
	A 1.0	Индикация режима работы
	A 1.1	Индикация ошибки позиционирования

Меркеры:	M 0.1	Двигатель в движении
	M 0.2	Меркер блокировки
	M 0.3	Меркер базовой точки
	M 0.4	Первое позиционирование завершено
	M 1.1	Время ожидания T1 завершено
	MD 8, MD 12	Вспомогат. меркер при вычислении числа шагов
	M 20.0	Выдача импульсов завершена
	MW 25	Счетчик неверных позиционирований
Сод. Акку:	AC0	Нижняя граница допуска
	AC1	Верхняя граница допуска
	AC2	Заданное значение
	AC3	Вспомогательный регистр

KOP (S7-MicroDOS)	AWL (TOOLITE2)

Основная программа

// Инициализация



```
LD      SM0.1                  // в первом цикле...
ATCH  0, 19  // Назначение подпрограммы прерыванийt 0 (Завершена                    //
последовательность импульсов)
ENI                            // Все прерывания деблокированы
MOVW 0, SMW70                  // Ширина импульса = 0  при широтно-импульсной модуляции
CALL    6                      // дальнейшая инициализация в ППр 6

// Быстрый счетчик HSC2

MOVB    16#FC, SMB57  // Байт управления для HSC 2
HDEF    2, 10                  // HSC 2 как A/B-счетчик с внешним сбросом
HSC     2                      // HSC 2 активирован
```

// Скорость позиционирования

```
2   |   MW25          K0                                    MOV_W
    |---------| == W |-------------------------------------| EN
    |                                                       |
    |                                                 K200- IN   OUT -SMW68
    |                                                       |
    |                                                       |
```

```
LDW=    MW25, 0                    // если нет ошибки позиционирования
MOVW 200, SMW68                    // быстрое позиционирование (T=200 мксек)
```

// Деблокировка вращения влево

```
3   |  M0.1    E1.5                                        A0.2     K1
    |-| / |----|    |-------------------------------------(   S    )
    |
```

```
LDN     M0.1                       // Двигатель выключен
U       E1.5                       // а переключатель направления движения = 1
S       A0.2,1                     // Деблокировка вращения влево
```

// Деблокировка вращения вправо

```
4   |  M0.1    E1.5                                        A0.2     K1
    |-| / |----| / |-------------------------------------(   R    )
    |
```

```
LDN     M0.1                       // Двигатель выключен
UN      E1.5                       // ... а переключатель направления движения = 0
R       A0.2,1                     // Деблокировка вращения вправо
```

Copyright ©1996 by SIEMENS

// Активирование блокировки

```
5   |  E1.1                                                  M0.2     K1
    |-|   |-----------------------------------------------------(   S   )
    |                                |
    |   MW25          K3             |
    |-------| == W |-------------|
    |
```

LD	**E1.1**	**// Нажата кнопка "ОСТАНОВ двигвтеля"**
OW=	**MW25, 3**	**// или 3 неверных позиционирования**
S	**M0.2,1**	**// Блокировка активирована**

// Подавление блокировки

```
6   |  E1.1    E1.0     MW25          K2                  M0.2     K1
    |-| / |----| / |----------| <= W |-------------------(   R   )
    |
```

LDN	**E1.1**	
UN	**E1.0**	**// Обе кнопки деблокированы**
UW<=	**MW25,2**	**// и < 3 неверных позиционирований**
R	**M0.2,1**	**// Подавление блокировки**

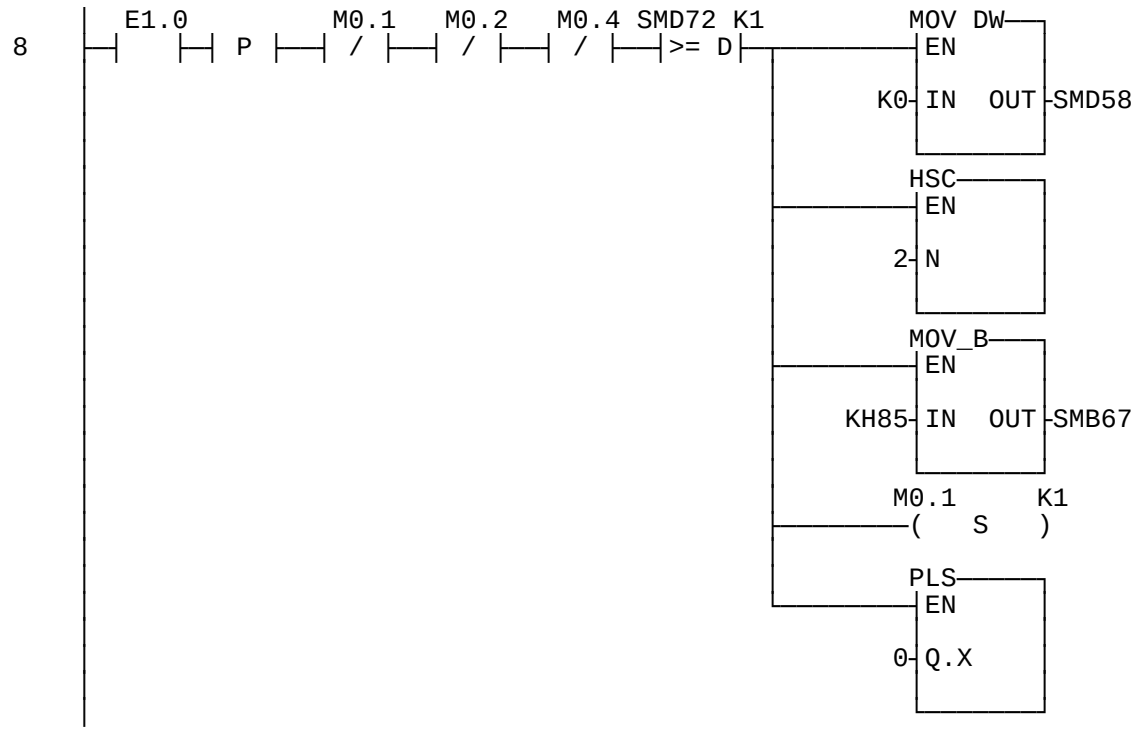
// Задание режима работ (Выход в базовую точку / Позиционирование)

```
7   |  E1.4            M0.2    M0.4                          1
    |-|   |----| P |----| / |----| / |-------------------( CALL  )
    |
```

LD	**E1.4**	
EU		**// Нажата кнопка "Установка/стирание базовой точки"**
UN	**M0.2**	**// и блокировка неактивна**
UN	**M0.4**	**// и нет позиционирования**
CALL 1		**// Режим работы задается в ППр 1**

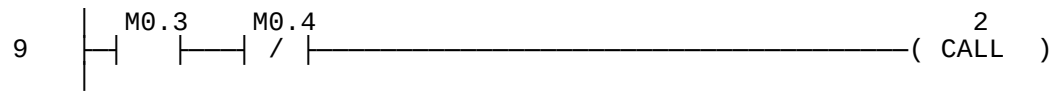
Copyright ©1996 by SIEMENS

// Пуск привода




```
LD      E1.0
EU                          // Нажата кнопка "ПУСК двигателя"
UN      M0.1                // и двигатель в покое
UN      M0.2                // и блокировка неактивна
UN      M0.4                // и нет позиционирования
UD>=    SMD72, 1            // и число шагов >= 1
MOVD    0, SMD58
HSC      2                  // Стартовое значение HSC 2 установить в 0
MOVB  16#85, SMB67          // Активирование функции выдачи импульсов PTO
S       M0.1,1              // Установка меркера "Двигатель в движении"
PLS     0                   // Пуск выдачи импульсов на A0.0
```

// Позиционирование




```
LD      M0.3                // Активен режим работы "Позиционирование"
UN      M0.4                // и нет позиционирования
CALL    2                   // вычисление числа шагов вППр 2
```

// Коррекция пути

```
     |  M1.1     MW25      K1          M0.2              TON——T98
10   |—|   |————————| >= W |————————| / |————————————|IN      |
     |                                                  |        |
     |                                              K100-|PT      |
     |                                                  |        |
     |                                                  |_______|
```

LD	**M1.1**	**// T1 завершил работу**
UW>=	**MW25,1**	**// и опознано неверное позиционирование**
UN	**M0.2**	**// и блокировка неактивна**
TON	**T98, 100**	**// Запуск времени ожидания T2 (1сек)**

```
     |  T98                                                  5
11   |—|   |————————————————————————————————————————————( CALL  )
     |                                                  |
     |                                                  |      MOV_W——┐
     |                                                  |————————|EN      |
     |                                                  |        |        |
     |                                                  |  K1000-|IN  OUT|-SMW68
     |                                                  |        |        |
     |                                                  |        |_______|
     |                                                  |      MOV_B——┐
     |                                                  |————————|EN      |
     |                                                  |        |        |
     |                                                  |   KH85-|IN  OUT|-SMB67
     |                                                  |        |        |
     |                                                  |        |_______|
     |                                                  |     M0.1     K1
     |                                                  |————————(   S   )
     |                                                  |
     |                                                  |      PLS——┐
     |                                                  |————————|EN      |
     |                                                  |        |        |
     |                                                  |     0-|Q.X     |
     |                                                  |        |        |
     |                                                  |        |_______|
     |                                                  |     M1.1     K1
     |                                                  |————————(   R   )
```

LD	**T98**	**// T2 завершил равботу**
CALL	**5**	**// Вычисление числа шагов коррекции в Ппр 5**
MOVW	**1000, SMW68**	**// Коррекция пути с 1 КГц**
MOVB	**16#85, SMB67**	**// Активирована функция PTO**
S	**M0.1,1**	**// Установка меркера "Двигатель в движении"**
PLS	**0**	**// Запуск выдачи импульсов на A 0.0**
R	**M1.1,1**	**// Сброс T2**

Copyright ©1996 by SIEMENS

// Контроль пути

```
12  | M20.0    M0.2                                    TON——T97
    |—|   |——| / |————————————————————————————————————|IN     |
    |                                                 |       |
    |                                              K50-|PT     |
    |                                                 |       |
    |                                                 |_______|
```

LD	**M20.0**	**// Завершена выдача импульсов**
UN	**M0.2**	**// и блокировка неактивна**
TON	**T97, 50**	**// Запуск времени ожидания T1 (500 мсек)**

```
13  | T97                                          M1.1     K1
    |—|   |————————————————————————————————————————(   S   )
    |                                          |
    |                                          |        4
    |                                          |————————( CALL  )
    |                                          |
    |                                          |      M20.0    K1
    |                                          |————————(   R   )
    |
```

LD	**T97**	**// T1 завершил работу**
S	**M1.1,1**	**// меркер конца работы T1**
CALL	**4**	**// Контроль пути: вызов UP 4**
R	**M20.0,1**	**// Сброс T1**

// Останов привода

```
14  | E1.1          M0.1                                  0
    |—|   |——| P |——|   |————————————————————————————( CALL  )
    |
```

LD	**E1.1**	
EU		**// Нажата кнопка "ОСТАНОВ двигателя"**
U	**M0.1**	**// и двигатель в движении**
CALL	**0**	**// Вызов подпрограммы 0**

// Квитирование ошибки после 3 неверных позиционирований

```
15  | E1.4          MW25          K3                      6
    |—|   |——| P |————————| == W |————————————————————( CALL  )
    |
```

LD	**E1.4**	
EU		**// Нажата кнопка квитирования**
UW=	**MW25,3**	**// и 3 неверных позиционирования**
CALL	**6**	**// назад к исходному состоянию в ППр 6**

Copyright ©1996 by SIEMENS

```
16 |-------------------------------------------------------( MEND  )
```

MEND // Конец основной программы

Подпрограммы

// Подпрограма 0 "Останов привода"

```
SBR:  0
18 |---|SM0.0|-------------------------------------------------------
   |                                                  MOV_B
   |                                                 |EN
   |                                                 |
   |                                          KHCB-|IN  OUT|-SMB67
   |                                                 |
   |                                                 |
   |                                                 |PLS
   |                                                 |EN
   |                                                 |
   |                                                 |0-|Q.X
   |                                                 |
   |                                                 |
   |                                                 |M0.1     K1
   |                                                 |-------( R  )
```

SBR	**0**	
LD	**SM0.0**	**// Всегда в единице**
MOVB	**16#CB, SMB67**	**// Активирование широтно-импульсной модуляции**
PLS	**0**	**// Останов выдачи импульсов на A0.0**
R	**M0.1,1**	**// Сброс меркера "Двигатель в движении"**

```
19 |-------------------------------------------------------( RET  )
```

RET // Конец подпрограммы 0

// Подпрограмма 1 "Задание режима работы“

```
SBR:  1
21 |---|M0.1|-------------------------------------------------------
   |                                                          0
   |                                                  ( CALL  )
```

SBR	**1**	
LD	**M0.1**	**// Двигатель в движении**
CALL	**0**	**// Останов привода**

// Требование выхода в базовую точку

```
22  |  M0.3                                          M0.3     K1
    |-| |-------------------------------------------(   R   )
    |                                                 |
    |                                                 A1.0     K1
    |                                                 |-------(   R   )
    |                                                 |
    |                                                 |
    |                                                 |      MOV_DW
    |                                                 |-------|EN
    |                                                 |       |
    |                                                 |  KH   -IN  OUT-SMD72
    |                                                 |1999997C|
    |                                                 |       |
    |                                                 |
    |                                                 |-------( CRET  )
```

LD	**M0.3**	**// "Позиционирование" активно**
R	**M0.3,1**	**// Сброс меркера базовой точки**
R	**A1.0,1**	**// Стереть сообщение "Позиционирование активно"**
MOVD	**16#1999997C, SMD72**	**// макс. число импульсов для нового выхода в баз.точку**
CRET		

// Требование „Позиционирования“

```
23  |  M0.3                                          M0.3     K1
    |-| / |-------------------------------------------(   S   )
    |                                                 |
    |                                                 A1.0     K1
    |                                                 |-------(   S   )
    |
```

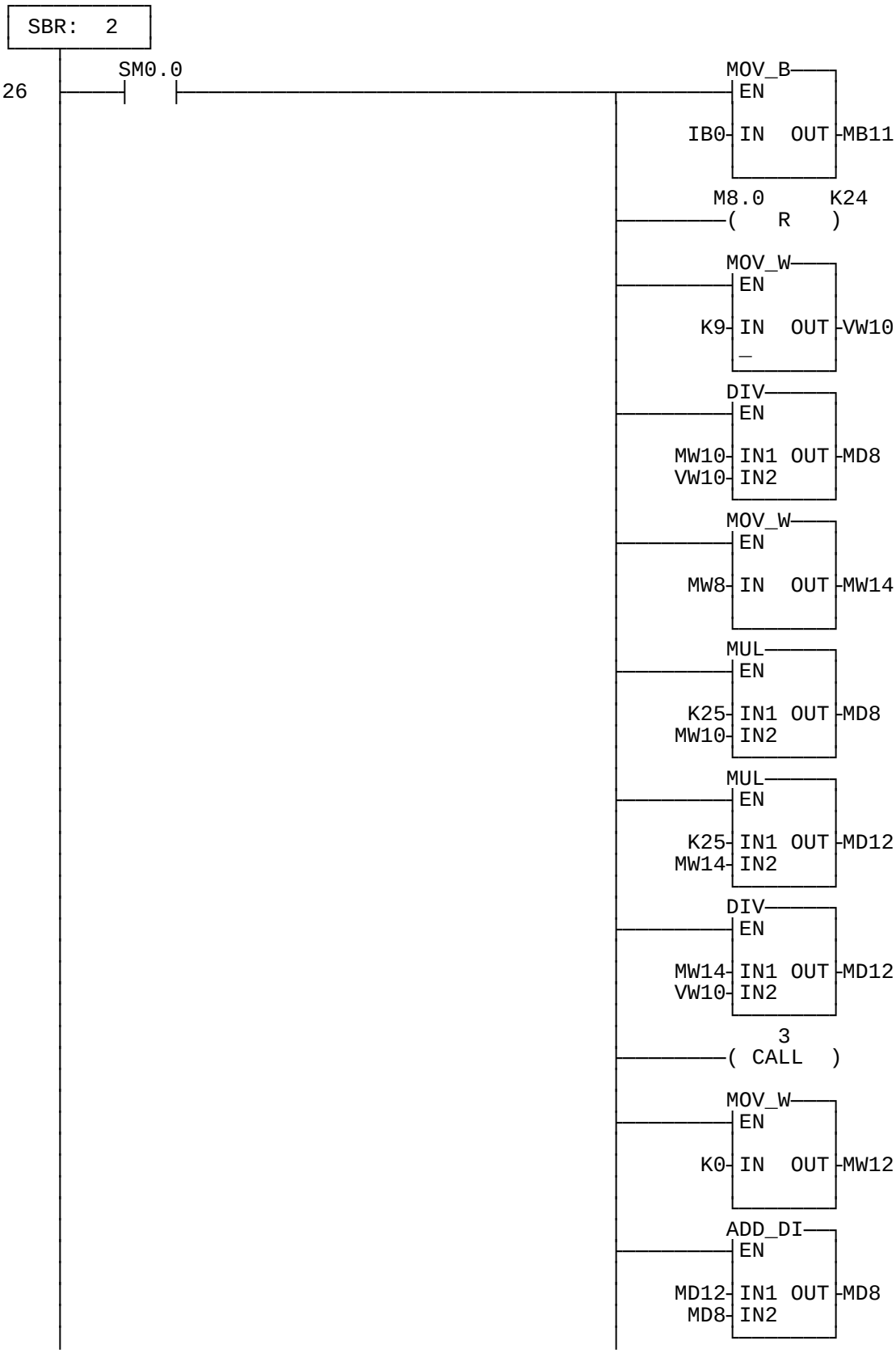
LDN	**M0.3**	**// Базовая точка еще не установлена**
S	**M0.3,1**	**// Установить меркер базовой точки**
S	**A1.0,1**	**// Вывод сообщения "Позиционирование активно"**

```
24  |-------------------------------------------------------(  RET  )
    |
```

RET **// Конец подпрограммы 1**

Copyright ©1996 by SIEMENS

// Подпрограмма 2 "Вычисление числа шагов, границы допуска"



```
SBR 2
LD      SM0.0                    // Всегда в единице
MOVB    EB0, MB11                // Скопировать угол поворота из EB0 в MD8
R       M8.0, 24                 // Стереть MB8 - MB10
MOVW    9,VW10
DIV     VW10, MD8                // Угол / 9 = q1 + r1 (q = Quotient)
MOVW  MW8, MW14            // r1 (Остаток 1) занести в MD 12
MUL     25, MD8                  // q1*25 -> MD8
MUL     25, MD12
DIV     VW10, MD12         // r1*25 / 9  = q2 + r2 (Quotient 2 + Остаток 2)
CALL    3                        // Округлить число шагов в подпрограмме 3


MOVW  0, MW12                    // Стереть r2
+D      MD12, MD8                // Занести число шагов в MD8
MOVD    MD8, AC2                 // Число шагов=Заданное значение
MOVD    AC2, SMD72
```

```
LD      E1.5                     // при вращении влево
INVD    AC2
INCD    AC2                      // Заданное значение инвертировать
```

```
                                                  MOV_DW
                                                 EN
                                              AC2 IN   OUT AC1

                                                  SUB_DI
                                                 EN
                                              AC0 IN1  OUT AC0
                                               K2 IN2

                                                  ADD_DI
                                                 EN
                                               K2 IN1  OUT AC1
                                              AC1 IN2
```

```
LD      SM0.0
MOVD    AC2, AC0
MOVD    AC2, AC1
-D      2, AC0                    // нижняя граница допуска
+D      2, AC1                    // верхняя граница допуска
```

```
29 ──────────────────────────────────────────────( RET )
```

```
RET                               // Конец подпрограммы 2
```

```
// Подпрограмма 3 "Округление числа шагов"
```

```
SBR:  3
      MW12          K5                                INC_W
31 ──────┤ >= W ├──────────────────────────────────────EN
                                                 MW14 IN   OUT MW14
```

```
SBR     3
LDW>= MW12,5                      // если r2 >= 5 / 9
INCW    MW14                      // число шагов увеличить на 1
```

```
32 ──────────────────────────────────────────────( RET )
```

```
RET                               // Конец подпрограммы 3
```

// Подпрограмма 4: "Контроль пути"

```
SBR:  4
34  HC2        AC1        HC2        AC0        M0.4     K1
    ──────┤ <= D ├──────────────┤ >= D ├──────┬──────────(   R   )
                                              │
                                              │      M25.0     K16
                                              ├──────────(   R   )
                                              │
                                              │       A1.1     K1
                                              ├──────────(   R   )
                                              │
                                              └──────────( CRET  )
```

```
SBR     4
LDD<= HC2, AC1
UD>=    HC2, AC0                    // Истинное значение в полосе допуска
R       M0.4,1                      // Сброс меркера первого позиционирования
R       M25.0, 16                   // Сброс счетчика неверн.позиционирований
R       A1.1,1                      // Стереть индикацию неверн.позиционирования
CRET
```

```
35  SM0.0                                          INC_W
    ──┤   ├──────────────────────────────────────┬──────┤EN
                                                 │      │
                                                 │  MW25┤IN  OUT├MW25
                                                 │      │
                                                 │      └──────┘
                                                 │       A1.1     K1
                                                 └──────────(   S   )
```

```
LD      SM0.0                       // Неверное позиционирование
INCW    MW25                        // Счетчик неверн.позиц.увеличить
S       A1.1,1                      // Индикация неверного позиционирования
```

```
36  ──────────────────────────────────────────────────────(   RET   )
```

```
RET                                 // Конец подпрограммы 4
```

Copyright ©1996 by SIEMENS

// Подпрограмма 5: "Вычисление числа шагов коррекции"

```
SBR:  5
38  SM0.0                                                  SUB_DI
                                                           EN
                                                     AC2 IN1  OUT AC3
                                                     HC2 IN2
```

```
SBR     5
LD      SM0.0                    // Всегда в единице
MOVD    AC2, AC3
-D      HC2, AC3                 // Заданное-Истинное ( AC3
```

```
39  HC2        AC0                                         MOV_DW
    <= D                                                   EN
                                                     AC3 IN   OUT SMD72
```

```
LDD<= HC2, AC0                   // Истинное <Заданное
MOVD    AC3, SMD72               // Число шагов=Заданное-Истинное
```

```
40  HC2        AC0       E1.5                          A0.2     K1
    <= D                                                (   R    )
                                                        ( CRET   )
```

```
LDD<= HC2, AC0
U       E1.5                     // при вращении влево
R       A0.2,1                   // происходит коррекция правого вращения
CRET
```

```
41  HC2        AC1                                         INV_DW
    >= D                                                   EN
                                                     AC3 IN   OUT AC3
                                                           INC_DW
                                                           EN
                                                     AC3 IN   OUT AC3
                                                           MOV_DW
                                                           EN
                                                     AC3 IN   OUT SMD72
```

```
LDD>= HC2, AC1                    // Истинное >Заданное
INVD    AC3
INCD    AC3
MOVD    AC3, SMD72                // Число шагов=Истинное-Заданное
```

```
42  |    HC2          AC1      E1.5                    A0.2     K1
    |---------| >= D |----------| / |-------------------------(   S   )
    |
```

```
LDD>= HC2, AC1
UN      E1.5                      // при вращении вправо
S       A0.2,1                    // происходит коррекция в левом вращении
```

```
43  |-------------------------------------------------------(  RET  )
    |
```

```
RET                               // Конец подпрограммы 5
```

```
// Подпрограмма 6: "Инициализация после пуска программы и квитирования ошибки"
```

```
SBR:  6
    |
45  |-------|   |-------------------------------------------------------------
    |                                                  M0.0     K128
    |                                                  |-------(   R   )
    |                                                  |
    |                                                  M25.0     K16
    |                                                  |-------(   R   )
    |                                                  |
    |                                                  A1.0     K2
    |                                                  |-------(   R   )
    |                                                  |
    |                                                  MOV_DW
    |                                                  |-------| EN
    |                                                  |       |
    |                                                  KH1999997C- IN  OUT -SMD72
    |                                                  |       |
    |                                                  |       |
```

```
SBR     6
LD      SM0.0                     // Всегда в единице
R       M0.0,128                  // Сброс M 0.0-M 15.7 (MD 0-MD 12)
R       M25.0, 16                 // Сброс счетчика неверных позиционирований
R       A1.0, 2                   // Сброс индикации режима работы и неверн. позиционир.
MOVD    16#1999997C, SMD72        // Число импульсов для выхода в базовую точку
```

Copyright ©1996 by SIEMENS

```
46 |-------------------------------------------------------( RET )
```

RET // Конец подпрограммы 6

Подпрограммы прерываний

```
//*****************************************************************
// Подпрограмма прерываний 0 "Завершение выдачи импульсов"
```

```
 INT:  0
 |
 |  SM0.0                                                  M0.1     K1
48 |--|   |-------------------------------------------------(   R   )
 |                                                 |
 |                                                 |       M20.0    K1
 |                                                 |-------(   S   )
 |
```

INT	**0**	
LD	**SM0.0**	
R	**M0.1,1**	// Сброс меркера "Двигатель в движении"
S	**M20.0,1**	// Меркер „Выдача импульсов завершена“

```
 |  M0.4                                                   M0.4     K1
49 |--| / |-------------------------------------------------(   S   )
 |
```

LDN	**M0.4**	// после 1го позиционирования установить
S	**M0.4,1**	// соответствующий меркер

```
50 |-------------------------------------------------------( RETI )
 |
```

RETI // Конец подпрограммы прерываний 0

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведенного здесь.

За правильную работу системы заказчик несет ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

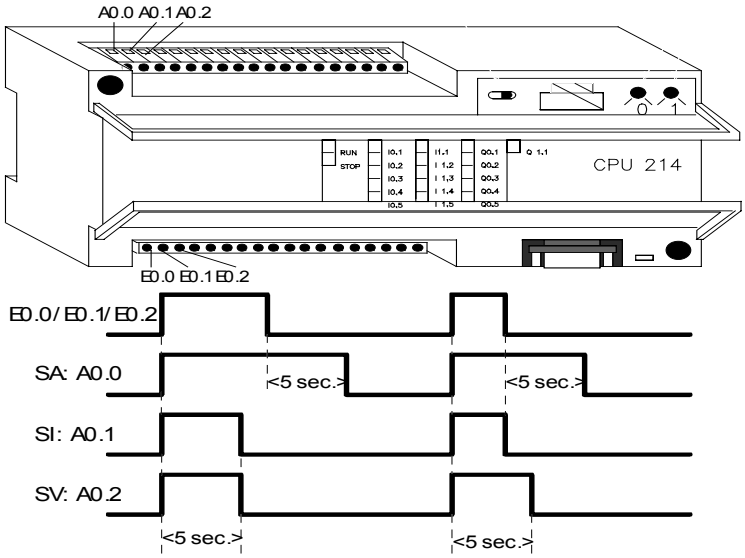
Группа	Пример к теме
1	**Реализация задержки выключения, импульса и удлиненного импульса**

Краткое описание

С помощью встроенных таймеров с задержкой включения S7-200, можно достаточно просто реализовать задержку выключения, импульс или удлиненный импульс, как это показано в данном примере.

При **‘Задержке выключения’** выходной сигнал A0.0 остается на определенное время дольше установленным, чем входной сигнал E0.0. **‘Импульс’** на выходе A0.1 держится 5 секунд после установки входного сигнала E0.1. Если вход E0.1 в течение этого времени будет сброшен, будет сброшен и ‘Импульс’. При **‘Запомненном ипульсе’** напротив выходной сигнал A0.2 все время остается активным 5 секунд, после того как вход E0.2 один раз установлен, все равно сбрасывается вход в течение этого времени или нет.

Схема включения



Описание программы вкл. листинг

Программа поделена на три части, которые независимо друг от друга реализуют задержку выключения, импульс и удлиненный импульс.

<u>Задержка выключения</u>

После включения входа E0.0 устанавливается выход A0.0. Если вход E0.0 снова сбрасывается (отрицательный фронт!) то запускается таймер T33 со временем работы 5 секунд. Пока работает таймер, т.е. бит таймера не установлен, вспомогательный меркер M0.0, а значит и выход A0.0 остается установленным. Только по прошествии 5 секунд установленный теперь бит таймера T33 сбрасывает вспомогательный меркер M0.0 и выход A0.0.

<u>Импульс</u>

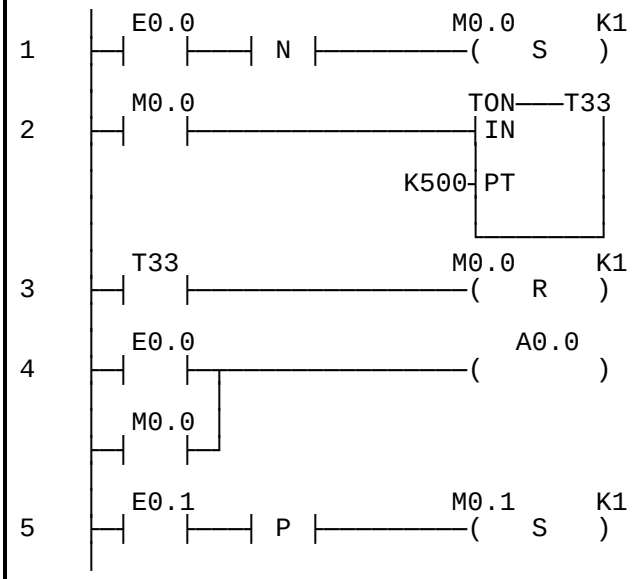
При включении входа E0.1 устанавливается выход A0.1 и по установлении вспомогательного меркера M0.1 запускается таймер T34. Выход A0.1 будет сброшен, если пройдет установленное время 5 секунд или сбросится вход E0.1.

<u>Удлиненный импульс</u>

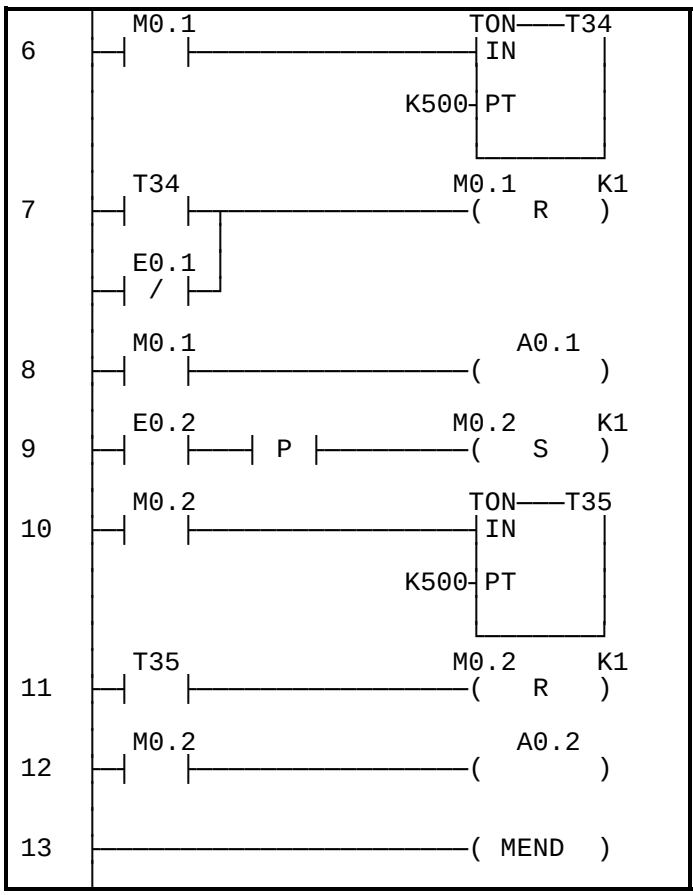
При включении входа E0.2 устанавливается выход A0.2 и по установлении вспомогательного меркера M0.2 запускается таймер T35. Выход A0.2 будет снова сброшен по прошествии 5 секунд.

KOP (S7-MicroDOS)	AWL (IEC)

Основная программа




```
// Задержка выключения SA:

LD      E0.0     // При отрицательном
ED               // фронте E0.0 устанавл.
S       M0.0, 1// вспомог.меркер  для SA
LD      M0.0     // При активном
                 // вспомогательном меркере
TON     T33, 500
                 // запускается таймер

LD      T33      // Через 5 секунд всп.
R       M0.0, 1// меркер сбрасывается
LD      E0.0     // Если E0.0 или
O       M0.0     // задержка активны
=       A0.0     // устанавл. выход A0.0

// Импульс SI:
LD      E0.1     // При положительном
EU               // фронте E0.1 устанавл.
S       M0.1,1// вспомогат. меркер для SI
```




```
LD      M0.1    // При активном
                // вспомогательном меркере
TON     T34, 500
                // запускается таймер

LD      T34     // Через 5 секунд или при
ON      E0.1    // неактивном входе
R       M0.1, 1// вспомогательный меркер
                // сбрасывается

LD      M0.1    // Вспомогательный меркер
=       A0.1    // копируется на выход A0.1

// Удлиненный импульс SV:
LD      E0.2    // При положительном
EU              // фронте E0.2 устанавл.
S       M0.2, 1// вспомогат.меркер для SV
LD      M0.2    // При активном
                // вспомогательном меркере
TON     T35, 500
                // запускается таймер

LD      T35     // Через 5 секунд
R       M0.2, 1// вспомогательный меркер
                //  сбрасывается
LD      M0.2    // Вспомогательный меркер
=       A0.2    // копируется на выход A0.2

MEND
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведённого здесь.

За правильную работу системы заказчик несёт ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

Copyright ©1996 by SIEMENS

SIMATIC
S7-200 Примеры

Группа	Тема
6	**ПИД Регулятор для S7-200 CPU 214**

Краткое описание

В данном программном примере показан метод программирования Пропорционально-Интегрально-Дифферинциального (ПИД) регулятора на CPU 214. Пример представляет собой полностью функциональную программу, требующую только привязки фактических входов и управляющих сигналов соответствующим переменным, чтобы стать работающим регулятором.

Чтобы подготовить пользователя к внесению изменений в программу, нижеследующий текст объясняет основные принципы ПИД регулятора, реализованного в простейшем коде. Ниже - краткий план всего документа:

1. Что делает программа?
2. Когда Вы можете применять ПИД регулятор?
3. Отличие автоматического от ручного режима?
4. Что и как делает ПИД регулятор?
5. Что такое Время обработки, Коэффициент усиления, Постоянные времени дифферинцирования и интегрирования?
6. Как подсчитывается Рассогласование?
7. Как подсчитывается Пропорциональная составляющая?
8. Как подсчитывается Интегральная составляющая?
9. Как подсчитывается Дифферинциальная составляющая?
10. Что, если окончательный Управляющий сигнал слишком большой?
11. Что необходимо добавить, чтобы сделать программу работоспособной для нашей системы?
12. Подстройка во время работы Постоянных времени интегрирования и дифферинцирования, Коэффициента усиления, Времени обработки и Режима.

Аппаратные требования

SIMATIC S7-200 CPU 214

ЗАМЕЧАНИЕ: ПИД регулятор работает только с SIMATIC S7-200 CPU 214, которое поддерживает арифметику с плавающей запятой.

Обзор ПИД регулятора

Что делает программа?
Данный компилируемый программный пример - скелет программы для реального ПИД регулятора и, как таковой, требует для нормальной работы внесения нескольких добавлений (например, чтение/запись входных/выходных переменных). Однако, перед тем, как обсудить это, давайте разберемся, что на самом деле делает ПИД программа на простом примере.

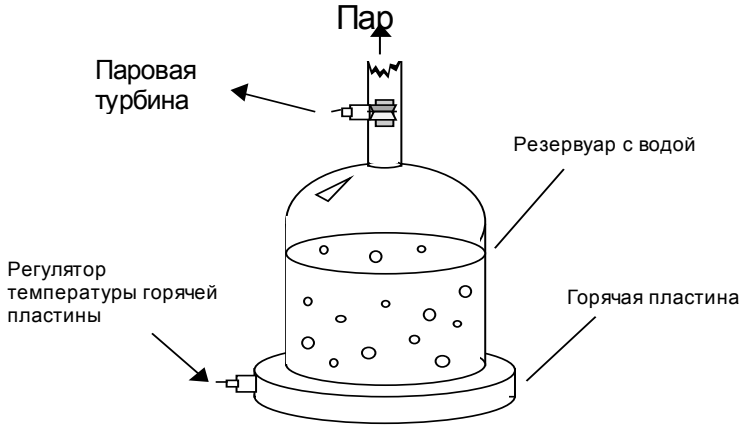


Рисунок 32.1 Пример установки

Когда Вы можете применять ПИД регулятор?
На рисунке 32.1 показан пример установки, к которой пользователь может подсоединить ПИД регулятор. На рисунке показаны резервуар с водой стоящий на горячей пластине, а так же устройство регулирования температуры горячей пластины и небольшая контрольная турбина для измерения скорости потока пара. Это установка, которая будет работать с ПИД регулятором благодаря взаимотоношению между двумя переменными: Вы можете непосредственно регулировать скорость потока пара изменяя температуру горячей пластины. На рисунке 32.2 показано, как обе переменных связаны с ПИД регулятором.

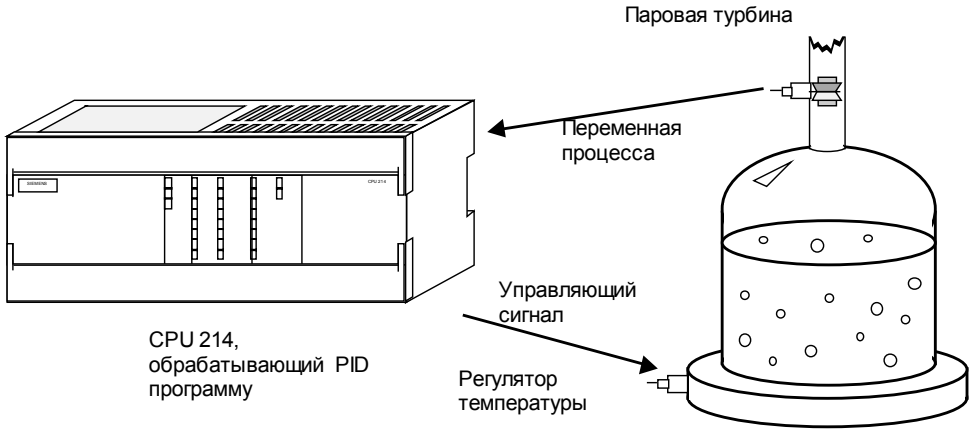


Рисунок 32.2 Пример установки управляемой ПИД

Переменные, представляющие состояние управляемой установки, называются "Переменными Процесса". В нашем примере выше, Вы можете видеть, что постоянная времени дифферинцирования, с которой пар вращает турбину, хороший индикатор события, которое мы пытаемся контролировать: скорость, с которой вода закипит. Управляющий сигнал - переменная, которая изменяется регулятором и воздействует на переменную процесса в различной степени (в зависимости от ее интенсивности), - при включении горячей пластины, вода закипает быстрее, вырабатывается больше пара и скорость вращения турбины увеличивается.

Таким образом, если есть переменные точно отражающие состояние процесса и есть настраиваемый регулятор, который пропорционально воздействует на переменную процесса, то возможно использовать ПИД регулятор. Распространенные установки, использующие ПИД регуляторы - это системы воздушного кондиционирования, смесители раствора, обогреватели и т.д.

Отличие автоматического от ручного режима?
Возможны два состояния нашего ПИД регулятора. Перевод регулятора в *Ручной* режим заставляет ПИД контур не делать ничего, и пользователь может непосредственно управлять управляющим сигналом. *Автоматический* - режим, в котором ПИД контур действительно управляет установкой. Для оставшейся части описания подразумевается, что регулятор находится в *Автоматическом* режиме.

Что и как делает ПИД регулятор?
ПИД регулятор подстраивает значение своего управляющего сигнала для достижения и балансирования значения переменной процесса данному ‘заданному значению’. Для вычисления значения управляющего сигнала для каждого случая, регулятор определяет значения трех различных составляющих: Пропорциональной, Интегральной и Дифферинциальной (используя, определенные пользователем значения: **Времени обработки**, **Коэффициента усиления**, **Постоянной времени дифферинцирования** и **интегрирования** вместе с вычисленным значением **Рассогласования**).

$$\textbf{Управляющий сигнал} = M_P + M_I + M_D$$

Формула 32.1

Что такое Время обработки, Коэффициент усиления, Постоянные времени дифферинцирования и интегрирования?
Время обработки - время цикла (в милисекундах) за которое ПИД контур пересчитывает управляющий сигнал. *Коэффициент усиления* воздействует на точность вычисления управляющего сигнала, от которого зависят все составляющие. *Постоянная времени дифферинцирования* - заданное время в милисекундах, которое используется для увеличения или уменьшения влияния Интегральной составляющей в уравнении. И наконец, значение *постоянная времени интегрирования* используется для управлением влиянием дифферинциальной составляющей в уравнении. Каждое их этих значений д.б. задано пользователем перед запуском ПИД регулятора.

Если пользователю не нужно интегрального воздействия (нет И в вычислении ПИД), то значение постоянной времени интегрирования задают равным 0. Если пользователю не нужно дифферинциального воздействия (нет Д в вычислении ПИД), то значение постоянной времени дифферинцирования задают равным 0. Если пользователю не нужно пропорционального воздействия (нет П в вычислении ПИД), то значение коэффициента усиления задают равным 0 (обычно коэффициент усиления - множитель при вычислении интегрального или дифферинциального коэффициента, но в случае, если он равен 0, его убирают из вычисления, для того чтобы иметь возмость управлять И-, ИД- или Д-контуром).

Как подсчитывается Рассогласование?
Рассогласование подсчитывается как разница между нормализованными значениями заданного значения (ЗЗ) и переменной процесса (ПП). Регулятор вычисляет это значение в три шага. Первые два шага - изменить заданное значение и переменную процесса на значения в диапазоне от 0 до 1 (нормализованная шкала). Это делается по формуле:

ЗЗ = текущее_ЗЗ / максимальное_значение
ПП = текущее_ПП / максимальное_значение

Формула 32.2 & 32.3

В данной формуле значения **текущее_ЗЗ** и **текущее_ПП** - текущие значения, которые поступают на регулятор, а составляющая **максимальное_значение** - максимальное значение, которое может быть. Например, если значения **текущее_ЗЗ** и **текущее_ПП** были прочитаны, как значения от 0 до 32,000, то составляющая **максимальное_значение** должна иметь значение 32,000.

После того, как эти два значения вычислены, составляющая рассогласования подсчитывается следующим образом:

Рассогласование = ЗЗ - ПП
Формула 32.4

Как подсчитывается Пропорциональная составляющая?
Пропорциональный составляющая, $\mathbf{M_P}$, вычисляется по следующей формуле:

$\mathbf{M_P}$ **=** ***Коэффициент усиления * Рассогласование***
Формула 32.2

Возвращаясь назад к нашему примеру с резервуаром с водой, пропорциональная составляющая говорит, что до тех пор пока скорость турбины выше заданного значения, температура пропорционально уменьшается для снижения скорости и до тех пор пока скорость турбины ниже заданного значения, температура пропорционально возрастает для увеличения скорости.

Как подсчитывается Интегральная Составляющая?
Интегральная составляющая, $\mathbf{M_I}$, вычисляется по следующей формуле:

$\mathbf{M_I}$ **= Накопленное значение + (**$\mathbf{C_I}$ ***** ***Рассогласование*****)**
Формула 32.3

В данной формуле появляются два новых термина. Первый - $\mathbf{C_I}$ - коэффициент Интегральной составляющей - вычисляется из **Постоянной времени дифферинцирования**:

$\mathbf{C_I}$ **=** ***Коэффициент усиления*** *** (Время обработки /**
Постоянная времени интегрирования)
Формула 32.4

Оба термина - **Время обработки** и **Постоянная времени интегрирования** - были введены ранее, но в данной формуле их использование очевидно. Чем больше значение **Постоянной времени интегрирования**, тем меньше влияние будет иметь интегральная составляющая на управляющий сигнал, в то время как большее **Время обработки** дает большее влияние (**Время обработки** так же воздействует на дифферинциальную составляющую, которая будет объяснена позднее).

Термин **Накопленное значение** в Формуле 32.3 соответствует (с технической стороны) области под кривой графика рассогласования по времени.

В принципе однако, **Накопленное значение** (в идеале) растет вместе с уровнем управляющего сигнала сохраняя систему стабильной и позволяя Пропорциональной и Дифферинциальной составляющим компенсировать любые небольшие отклонения. В нашем примере с резервуаром с водой это означает, что со временем части **Накопленного значения** $\mathbf{M_I}$ будут единственными значащими составляющими конечного значения управляющего сигнала, а составляющие $\mathbf{M_P}$ и $\mathbf{M_D}$ будут активны (не равны 0) только, когда возникают отклонения.
Для времени **n** формула для $\mathbf{M_I}$ и составляющей **Накопленное значение** следующая:

$\mathbf{M_{I,n}}$ **= Накопленное значение**$_{\mathbf{n-1}}$ **+ (**$\mathbf{C_i}$ *** Рассогласование)**
Накопленное значение$_{\mathbf{n}}$ **=** $\mathbf{M_{I,n}}$
Формула 32.5

Как подсчитывается Дифферинциальная составляющая?

Дифферинциальная составляющая для заданного времени **n** вычисляется по следующей формуле:

$$\mathbf{M_D = C_D * (ПП_{n-1} - ПП_n)}$$

Формула 32.6

В этой формуле вводится только один новый термин - $\mathbf{C_D}$, который вычисляется по Формуле 32.7.

C_D = *Коэффициент усиления* * (Постоянная времени дифферинцирования / Время обработки)

Формула 32.7

Термин **Время обработки** (который так же используется при вычислении $\mathbf{C_I}$) - то же самое время обработки, упоминавшееся ранеее. В Дифферинциальной составляющей **Время обработки** обратно пропорционально дифферинциальному компоненту, т.к. **Постоянная времени дифферинцирования** прямо пропорциональна.

Что, если окончательное значение управляющего сигнала слишком большое?
Для многих процессов (таких как наш резервуар с водой) переменная процесса не мгновенно реагирует на изменение значения управляющего сигнала - если вода в резервуаре была холодна как лед, то даже управляющий сигнал равный 100% не вызывает мгновенного увеличения потока пара. И наоборот, если вода кипит и мы устанавливаем управляющий сигнал = 0%, то это не вызывет немедленного уменьшения потока пара.

Из-за подобной "инертности системы" значение управляющего сигнала для некоторого момента времени может иметь значение больше, чем 100% или меньше, чем 0%. Поэтому ПИД программа осуществляет ограничение Управляющего сигнала. Если управляющий сигнал больше, чем 100%, то он ограничивается = 100%. Если управляющий сигнал падает ниже 0%, то он фиксируется = 0%.

Осталось решить только одну проблему - относительно части **Накопленного значения** Интегральной составляющей. Когда управляющий сигнал системы остается на большой период времени = 100% (например, при начальном нагреве холодной воды в нашем резервуаре), интегральная сумма (которую представляет составляющая **Накопленное значение**) может вырасти в очень большое значение. Это означает, что когда переменная начинает реагировать, составляющая **Накопленное значение** будет сохранять вычисляемый управляющий сигнал на уровне 100% до тех пор, пока она не уменьшится. В результате управляющий сигнал колеблется от одного предела к другому; но это можно избежать используя ограничение Накопленного значения.

Есть несколько различных типов ограничения Накопленного значения, но на самом деле только один используется в программе. Существует два различных условия, при возникновении которых Накопленное значение ограничивается и им соответсвует две формулы:

Если Управляющий сигнал > 1
Управляющий сигнал = 1 - (M_P + M_D)

Формула 32.8

Если Управляющий сигнал < 0
Управляющий сигнал = - (M_P + M_D)

Формула 32.9

Как показано в формуле, если **Управляющий сигнал** становится больше, чем 1, то его **Накопленное значение** подстраивается так, что сумма $\mathbf{M_P}$, $\mathbf{M_D}$, и **Накопленного значения** будет = 1. И наоборот, если **Управляющий сигнал** падает ниже 0, то его **Накопленное значение** подстраивается так, что сумма будет = 0. Подстраиваемое **Накопленное значение** ограничивается так, что его максимальное значение = 1, а минимальное = 0.

Что необходимо добавить, чтобы сделать программу работоспособной для нашей системы?

1. Чтение переменной процесса
2. Запись Управляющего сигнала
3. Установка Заданного значения
4. Подстройка шкалы для Входа и Заданного значения
5. Подстройка шкалы для Управляющего сигнала
6. Подстройка значений **Постоянных времени интегрирования и дифферинцирования**, **Коэффициента усиления** и **Времени обработки**.

Чтение переменной процесса
Переменная процесса (переменная, которая реально отражает состояние управляемой системы) д.б. прочитана в программе пользователя в VW3002.

Запись Управляющего сигнала
ПИД контур помещает значение Управляющего сигнала в VW3006, где к нему может обращаться программа пользователя.

Установка Заданного значения
Заданное значение д.б. помещено в VW3004 в программе пользователя.

Подстройка шкалы для Входа и Заданного значения
В начале каждого вызова ПИД программы (сразу после **LBL 0**) программа преобразует текущее_ПП значение в нормализованное (0-1) значение базируясь на шкале от 0 до 32,000. Если прочитанное значение лежит в другой шкале, Вы должны изменить 32,000 на новое значение между метками LBL 0 и LBL20. Например, если шкала от 0 до 100, Вы должны изменить 32,000 на 100.

Подстройка шкалы для Управляющего сигнала
В сегменте, сразу перед командой **RETI** в конце программы, значение Управляющего сигнала изменяется из нормализованного значения (0-1) на масштабированное число (0-32,000). Как и в предыдущем абзаце, для изменения шкалы управляющего сигнала, надо просто изменить 32,000 на новое максимальное значение.

*Подстройка значений **Постоянных времени интегрирования и дифферинцирования**, **Коэффициента усиления** и **Времени обработки***
Последнее, что необходимо реализовать в программе пользователя, это задание значений **Постоянных времени интегрирования и дифферинцирования**, **Коэффициента усиления**, и **Времени обработки**. Это д.б. сделано в подпрограмме, которая вызывается только в первом цикле. Значения д.б. расположены, как показано:

Постоянная времени интегрирования (VD3020 (реальное)
Постоянная времени дифферинцирования (VD3016 (реальное)
Коэффициент усиления (VD3008 (реальное)
Время обработки (VD3012 (реальное)

Подстройка во время работы Постоянных времени интегрирования и дифферинцирования, Коэффициента усиления, Времени обработки и Режима
Значения **Постоянных времени интегрирования и**, **дифферинцирования**, **Коэффициента усиления** и **Времени обработки** устанавливаются во время первого прохождения ПИД контура, и любые последующие изменения значений в VD3008-VD3020 не будут восприняты в программе пока не будет установлен бит актуализации управляющего байта. Управляющий байт - VB3001, а бит актуализации - 7. Если V3001.7 равен 1, то при следующем прохождении ПИД контура все вышеназванные значения будут актуализированы.

Управляющий байт так же м.б. использован для изменения режима ПЛК. Когда установлен бит актуализации, значение в V3001.0 показывает новый режим. Значение = 1 означает *Автоматический* режим, а 0 - *Ручной*.

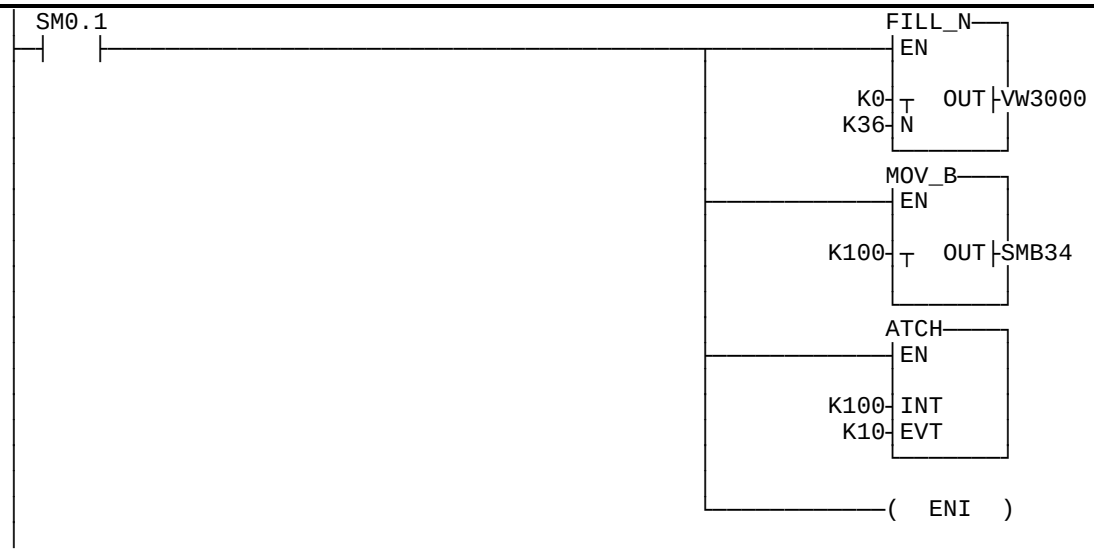
Размер программы - 335 слов.

Главная программа

```
// Описание:
//
// Данная программа пользователя S7-200 реализует простой ПИД контур.
// Тело ПИД контура встроено в пользовательскую программу обработки прерываний,
// присоединеннную к прерыванию по времени. Блок V памяти, начиная с VB3000,
// зарезирвирован для ПИД алгоритма. Данная область памяти разделена на две части:
// на область пользователя и рабочую область переменных, как показано ниже:
//
//    VB3000          --> управляющий_байт                        [битовое поле, описано ниже]
//    VB3001          --> байт_состояния                    [битовое поле, описано ниже]
//    VW3002          --> текущее_ПП                              [целое со знаком, диапазон 0-32000]
//    VW3004          --> текущее_ЗЗ                              [целое со знаком, диапазон 0-32000]
//    VW3006          --> текущий_Управляющий_сигнал        [целое со знаком, диапазон 0-32000]
//    VD3008          --> текущий_Коэффициент_усиления      [реальное]
//    VD3012          --> текущее_Время_обработки           [целое со знаком, в msec интервале]
//    VD3016          --> Постоянная_времени_дифферинцирования  [реальное, в msec]
//    VD3020          --> Постоянная_времени_интегрирования     [реальное, в msec]
//    VD3024-3029     --> зарезервировано для пользователя
//          -------------------------------------------------------
// рабочая    VB3030          --> внутреннее_состояние      [битовое поле, описано ниже]
//            VD3031          --> ПП                                [реальное, масштабирован. 0.0-1.0]
//            VD3035          --> ЗЗ                                [реальное, масштабирован. 0.0-1.0]
//            VD3039          --> Управляющий_сигнал                [реальное, масштабирован. 0.0-1.0]
//            VD3043          --> Рассогласование           [реальное]
//            VD3047          --> Накопленное_значение      [реальное]
//            VD3051          --> ПП_OLD                            [реальное]
//            VD3055          --> И_коэф                            [реальное]
//            VD3059          --> Д_коэф                            [реальное]
//            VD3063          --> Коэффициент_усиления      [реальное]
//            VD3067          --> Время_обработки           [реальное]
//
//
// Каждый раз при обработке контура ПИД программа использует актуальное ПП и
// заданное значение, хранящиеся в области пользователя в ячейках текущее_ПП и текущее_ЗЗ.
// управляющий байт д.б. использован для разрешения изменений режима, коэффициента
// усиления, времени обработки, и постоянных времени дифферинцирования/интегрирования
// В начале, необходимые значения данных д.б. записаны в ячейки обласи пользователя и
// в управляющий байт. Затем, главный значащий бит управляющего байта д.б. установлен,
// чтобы ПИД программа приняла новые значения. ПИД программа очистит MSB, когда будет
// обработано новое значение.
//
//    бит   7  6  5  4  3  2  1  0
//          x  0  0  0  0  0  0  x
//          ^                    ^
//          |                    |
//          |                    --- режим (1=автоматический, 0=ручной)
//          |
//          ---- обновление значений (1=обновление)
//
//
// Байт состояния используется для оповещения пользователя о текущей конфигурации контура.
// Формат байта состояния следующий:
//
//    бит   7  6  5  4  3  2  1  0
//          0  0  0  0  0  0  0  x
//                               ^
//                               |
//                               --- режим (1=автоматический, 0=ручной)
//
//
// Внутренний байт состояния из рабочей области данных, используется ПИД программой для
// "внутреннего" флага. Формат этого байта следующий:
//
//    бит   7  6  5  4  3  2  1  0
//          x  0  0  0  0  0  0  x
//          ^                    ^
//          |                    |
//          |                    --- режим (1=автоматический, 0=ручной)
//          ---- флаг первого цикла (1=первый цикл)
```

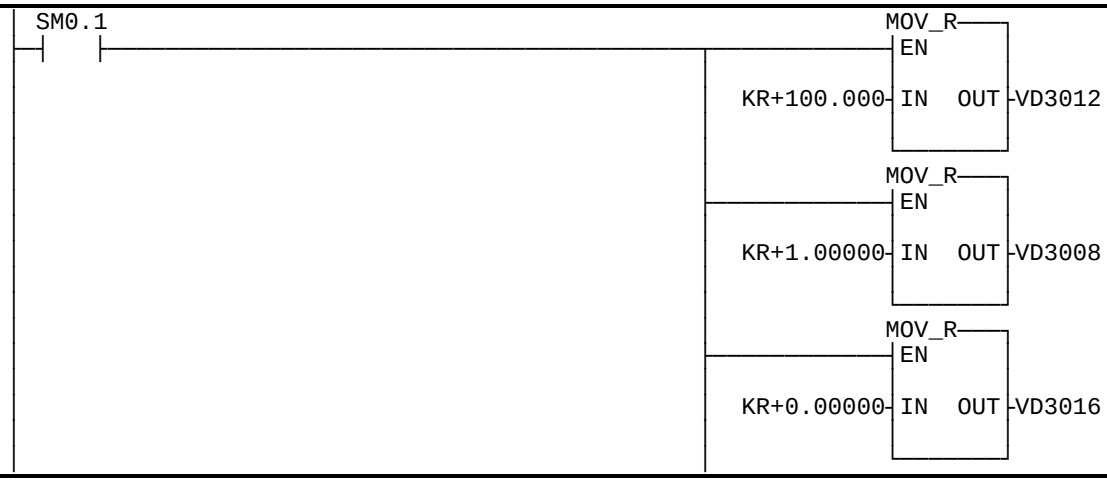
//

1



LD	**SM0.1**			**//**
FILL	**0**	**VW3000,**	**36**	**// очистить область пользователя V памяти**
MOVB	**100**	**SMB34**		**// тест кода - установка для 100 msec**
ATCH	**100**	**10**		**// время обработки (то же самое значение**
				// д.б. использовано для ПИД
				// текущее_Время_обработки (VD3012))
ENI				**// разрешить прерывания**

2



```
                                                  MUL_R
                                                 EN
                                                 KR+60000.00 IN   OUT VD3020
                                                 MOV_B
                                                 EN
                                                 KH81 IN   OUT VD3000
```

```
// Программа пользователя должна загрузить поля пользователя и запустить ПИД.
// Программа пользователя должна заменить следующую демонстрационную программу.

// *********Начало демонстрационной программы*********
LD      SM0.1                   // В первом цикле установить ПИД программу
MOVR    100.0,   VD3012         // текущее_Время_обработки должно иметь такое
                                // же значение, что и загруженное в SMB34
MOVR    1.0,     VD3008         // текущий_Коэффициент_усиления
MOVR    0.0,     VD3016         // Постоянная_времени_дифферинцирования
                                // = 0 минут
MOVR    60000.0, VD3020         // Постоянная_времени_интегрирования = 1 минута
MOVB    16#81,   VB3000         // Перевести ПИД в Автоматический режим и
                                // установить флаг первого цикла
```

```
3   SM0.0                                                  MOV_W
                                                          EN
                                                  AIW0 IN   OUT VW3002
                                                          MOV_W
                                                          EN
                                                  VW3006 IN   OUT AQW0
```

```
LD      SM0.0                   // В каждом цикле
MOVW    AIW0,    VW3002         // Поместить значение аналогого входа в
                                // текущую_ПП
MOVW    VW3006,  AQW0           // Поместить текущий_Коэффициент_усиления
                                // в Аналоговый выход

// Заданное текущее_ЗЗ д.б. загружено с устройства взаимодействия с оператором.

// ********Конец демонстрационной программы*********
```

```
4                                                          ( MEND  )
```

```
MEND                            // Конец главной программы
```

```
 INT:100
    V3000.7                                                    0
6   ─┤ / ├──────────────────────────────────────────────────( JMP )
```

// INT 100 - ПИД программа
// Проверка индикации -актуализировать ли значение контура. Если бит установлен, пересчитать
// новые интегральный и дифферинциальный коэффициенты и произвести необходимые
// изменения режима.

INT	**100**	**//**
LDN	**V3000.7**	**// Если (управляющий_байт.бит_актуализации =**
		// ИСТИНА)
JMP	**0**	**//**

```
    SM0.0                                                  MOV_R
7   ─┤   ├──────────────────────────────────────────────┬──┤EN
                                                         VD3008-IN  OUT-VD3063
                                                         MOV_R
                                                         ├──┤EN
                                                         VD3012-IN  OUT-VD3067
                                                         MOV_R
                                                         ├──┤EN
                                                         VD3067-IN  OUT-VD3055
                                                         MOV_R
                                                         ├──┤EN
                                                         VD3020-IN  OUT-AC0
                                                         DIV_R
                                                         ├──┤EN
                                                         VD3055-IN1 OUT-VD3055
                                                            AC0-IN2
                                                         DIV_R
                                                         └──┤EN
                                                         VD3016-IN1 OUT-VD3059
                                                         VD3067-IN2
```

LD	**SM0.0**		**//**
MOVR	**VD3008,**	**VD3063**	**// Коэффициент_усиления =**
			// текущий_Коэффициент_усиления
DTR	**VD3012,**	**VD3067**	**// Время_обработки = текущее_Время_обработки**
MOVR	**VD3067,**	**VD3055**	**// И_коэф = Время_обработки /**
			// Постоянная_времени_интегрирования
MOVR	**VD3020,**	**AC0**	**//**
/R	**AC0,**	**VD3055**	**//**
MOVR	**VD3016,**	**VD3059**	**// Д_коэф = Постоянная_времени_**
			// дифферинцирования / Время_обработки
/R	**VD3067,**	**VD3059**	**//**

```
    KR+0.00000    VD3063                                   MUL_R
8   ├─────────┤ == R ├──────────┤NOT├──────────────┬──────────┤EN
```

```
                                                  VD3055-IN1 OUT-VD3055
                                                  VD3063-IN2
                                                         MUL_R
                                                         EN
                                                  VD3059-IN1 OUT-VD3059
                                                  VD3063-IN2
```

LDR=	0.0,	VD3063	// Если (Коэффициент_усиления <> 0.0)
NOT			//
*R	VD3063,	VD3055	// И_коэф = И_коэф * Коэффициент_усиления
*R	VD3063,	VD3059	// Д_коэф = Д_коэф * Коэффициент_усиления

```
     V3000.0                                                V3030.7    K1
9    ─┤ / ├──────────────────────────────────────────────────(   S   )
     V3030.0                                                V3030.0    K1
     ─┤   ├──────────────────────────────────────────────────(   R   )
                                                            V3001.0    K1
                                                            (   R   )
                                                                1
                                                            (  JMP  )
```

LDN	V3000.0		// Если (управляющий_байт.режим = Ручной И
O	V3030.0		// внутреннее_состояние.режим = Автоматический)
S	V3030.7,	1	// внутреннее_состояние.первый цикл = ИСТИНА
R	V3030.0,	1	// внутреннее_состояние.режим = Ручной
R	V3001.0,	1	// байт_состояния.режим = Ручной
JMP	1		// Конец Если

```
     V3000.0                                                V3030.7    K1
10   ─┤   ├──────────────────────────────────────────────────(   S   )
     V3030.0                                                V3030.0    K1
     ─┤ / ├──────────────────────────────────────────────────(   S   )
                                                            V3001.0    K1
                                                            (   S   )
                                                               MOV_R
                                                            EN
                                                  VD3039-in  OUT-VD3047
                                                               MOV_W
                                                            EN
                                                  VW3002-in  OUT-VW3004
```

LD	V3000.0		// Иначе Если (управляющий_байт.режим =
			// Автоматический И
ON	V3030.0		// внутреннее_состояние.режим = Ручной)
S	V3030.7,	1	// внутреннее_состояние.первый_цикл = ИСТИНА
S	V3030.0,	1	// внутреннее_состояние.режим = Автоматический
S	V3001.0,	1	// байт_состояния.режим = Автоматический
MOVR	VD3039,	VD3047	// Накопленное_значение = Управляющий_сигнал
MOVW	VW3002,	VW3004	// текущее_ЗЗ = текущее_ПП

```
 LBL:  1
    SM0.0                                                    V3000.7    K1
12 ─┤   ├──────────────────────────────────────────────────────(   R   )
```

LBL	**1**		**//**
LD	**SM0.0**		**//**
R	**V3000.7,**	**1**	**// управляющий_байт.бит_обновления = FALSE**

```
 LBL:  0
    VW3002      K32000                                        MOV_R
14 ─────┤ >= W ├──────────────────────────────────────────┬──┤EN    │
                                                          KR+1.00000─┤in  OUT├─VD3031
                                                                     └──────┘
                                                                       10
                                                          └──────────(  JMP  )
```

LBL	**0**		**// Масштабировать значение текущее_ПП в**
			// сгенерированное значение ПП, используемое в
			// вычислении ПИД
LDW>=	**VW3002,**	**32000**	**// Если (текущее_ПП > 32000)**
MOVR	**1.0,**	**VD3031**	**// ПП = 1.0**
JMP	**10**		**//**

```
    VW3002       K0                                           MOV_R
15 ─────┤ <= W ├──────────────────────────────────────────┬──┤EN    │
                                                          KR+0.00000─┤in  OUT├─VD3031
                                                                     └──────┘
                                                                       10
                                                          └──────────(  JMP  )
```

LDW<=	**VW3002,**	**0**	**// Иначе Если (текущее_ПП <= 0)**
MOVR	**0.0,**	**VD3031**	**// ПП = 0.0**
JMP	**10**		**//**

```
    SM0.0                                                    SUB_DI
16 ─┤   ├──────────────────────────────────────────────────┬──┤EN    │
                                                          AC1─┤IN1 OUT├─AC1
                                                          AC1─┤IN2    │
                                                                 └──────┘
                                                             MOV_W
                                                          ├──┤EN    │
                                                        VW3002─┤in  OUT├─AC1
                                                                 └──────┘
                                                             DI_REAL
                                                          ├──┤EN    │
                                                          AC1─┤in  OUT├─VD3031
                                                                 └──────┘
                                                             DIV_R
                                                          └──┤EN    │
```

```
                                                                  VD3031-IN1 OUT-VD3031
                                                             KR+32000.0-IN2
```

```
LD      SM0.0                          // Иначе
-D      AC1,      AC1                  //
MOVW    VW3002,   AC1                  // ПП = DTR(текущее_ПП)/32000.0
DTR     AC1,      VD3031               //
/R      32000.0,  VD3031               //
```

```
LBL: 10
    VW3004      K32000                                          MOV_R
18  ---------| >= W |-------------------------------------------EN
                                                  KR+1.00000-in  OUT-VD3035
                                                                 20
                                                  ------------(  JMP  )
```

```
LBL     10                             // Масштабировать текущее_ЗЗ в
                                       // сгенерированное значение ПП, используемое в
                                       // вычислении ПИД
LDW>=   VW3004,   32000                // Если (текущее_ЗЗ > 32000)
MOVR    1.0,      VD3035               // ЗЗ = 1.0
JMP     20                             //
```

```
    VW3004       K0                                             MOV_R
19  ---------| <= W |-------------------------------------------EN
                                                  KR+0.00000-in  OUT-VD3035
                                                                 20
                                                  ------------(  JMP  )
```

```
LDW<=   VW3004,   0                    // Иначе Если (текущее_ЗЗ <= 0)
MOVR    0.0,      VD3035               // ЗЗ = 0.0
JMP     20                             //
```

```
    SM0.0                                                        SUB_DI
20  --|   |-------------------------------------------------------EN
                                                        AC1-IN1 OUT-AC1
                                                        AC1-IN2
                                                                 MOV_W
                                                  ------------EN
                                                     VW3004-in  OUT-AC1
                                                                 DI_REAL
                                                  ------------EN
                                                        AC1-in  OUT-VD3035
                                                                 DIV_R
```

```
                                                  └──────────┤EN      │
                                                             │        │
                                                   VD3035-IN1 OUT-VD3035
                                                KR+32000.0-IN2      │
                                                             └────────┘
```

```
LD      SM0.0                   // Иначе
-D      AC1,     AC1            //
MOVW    VW3004,  AC1            // ЗЗ = DTR(текущее_ЗЗ)/32000.0
DTR     AC1,     VD3035         //
/R      32000.0, VD3035         //
```

```
 LBL: 20
    V3030.0                                                  30
22  ─┤ / ├──────────────────────────────────────────────────( JMP  )
```

```
LBL     20                      // Вычислить рассогласование, Если контур в
                                // Автоматическом режиме
LDN     V3030.0                 // Если (контур в Автоматический режим)
JMP     30                      //
```

```
    SM0.0                                                  SUB_R
23  ─┤   ├──────────────────────────────────────────────────┤EN      │
                                                             │        │
                                                   VD3035-IN1 OUT-VD3043
                                                   VD3031-IN2      │
                                                             └────────┘
```

```
LD      SM0.0                   //
MOVR    VD3035,  VD3043         // Рассогласование = ЗЗ - ПП
-R      VD3031,  VD3043         //
```

```
 LBL: 30
    V3030.0                                                  40
25  ─┤ / ├──────────────────────────────────────────────────( JMP  )
```

```
LBL     30                      // Вычислить значение управляющего сигнала,
                                // Если контур в Автоматическом режиме.
                                // В данном разделе приняты следующие
                                // обозначения:
                                // AC1 = сначала содержит Mp, затем Ma
                                // AC2 = содержит Mi
                                // AC3 = содержит Md
LDN     V3030.0                 // Если (контур в Автоматическом режиме)
JMP     40                      //
```

```
26  SM0.0
    ──┤ ├──┬──────────────────────────────────────────────────────────────────────────────
              │                                                                  MUL_R
              │                                                                 ┌──────┐
              │                                                                 │ EN   │
              │                                                                 │      │
              │                                                        VD3063-  │IN1 OUT│-AC1
              │                                                        VD3043-  │IN2   │
              │                                                                 └──────┘
              │                                                                  MUL_R
              │                                                                 ┌──────┐
              │                                                                 │ EN   │
              │                                                                 │      │
              │                                                        VD3055-  │IN1 OUT│-AC2
              │                                                        VD3043-  │IN2   │
              │                                                                 └──────┘
              │                                                                  ADD_R
              │                                                                 ┌──────┐
              │                                                                 │ EN   │
              │                                                                 │      │
              │                                                        VD3047-  │IN1 OUT│-AC2
              │                                                           AC2-  │IN2   │
              │                                                                 └──────┘
              │                                                                  MOV_R
              │                                                                 ┌──────┐
              │                                                                 │ EN   │
              │                                                                 │      │
              │                                                     KR+0.00000- │in  OUT│-AC3
              │                                                                 └──────┘
```

```
LD      SM0.0                   //
MOVR    VD3063,     AC1         // Mp = Коэффициент_усиления * Рассогласование
*R      VD3043,     AC1         //
MOVR    VD3055,     AC2         // Mi = Накопленное_значение + (И_коэф *
                                // Рассогласование)
*R      VD3043,     AC2         //
+R      VD3047,     AC2         //
MOVR    0.0,        AC3         // Md = 0.0
```

```
27  KR+0.00000    VD3059          V3030.7                                  SUB_R
    ──────┤ == R ├──────────┤NOT├──┤ / ├──────────────────────────────────┌──────┐
              │                                                                 │ EN   │
              │                                                                 │      │
              │                                                        VD3051-  │IN1 OUT│-AC3
              │                                                        VD3031-  │IN2   │
              │                                                                 └──────┘
              │                                                                  MUL_R
              │                                                                 ┌──────┐
              │                                                                 │ EN   │
              │                                                                 │      │
              │                                                           AC3-  │IN1 OUT│-AC3
              │                                                        VD3059-  │IN2   │
              │                                                                 └──────┘
```

```
LDR=    0.0,        VD3059      // Если (Д_коэф <> 0.0) И
NOT                             //
        V3030.7                 // не внутреннее_состояние.первый_цикл
MOVR    VD3051,     AC3         // Md = Д_коэф * (ПП_Старое - ПП)
-R      VD3031,     AC3         //
*R      VD3059,     AC3         //
```

```
28  SM0.0
    ──┤ ├──┬──────────────────────────────────────────────────────────────────────────────
              │                                                                  MOV_R
              │                                                                 ┌──────┐
              │                                                                 │ EN   │
              │                                                                 │      │
              │                                                        VD3031-  │in  OUT│-VD3051
              │                                                                 └──────┘
              │                                                                  ADD_R
              │                                                                 ┌──────┐
              │                                                                 │ EN   │
              │                                                                 │      │
              │                                                           AC3-  │IN1 OUT│-AC1
              │                                                           AC1-  │IN2   │
              │                                                                 └──────┘
              │                                                                  ADD_R
```

```
                                                  EN
                                          AC1- IN1 OUT -VD3039
                                          AC2- IN2
```

```
LD      SM0.0                   //
MOVR    VD3031,   VD3051        //   ПП_Старое = ПП
+R      AC3,      AC1           //   Ma = Mp + Md
MOVR    AC1,      VD3039        // Управляющий_сигнал = Ma + Mi
+R      AC2,      VD3039        //
```

```
    VD3039    KR+1.00000                                  SUB_R
29 ────────┤ <= R ├──────────┤NOT├──────────────────────┬──────────
                                                         │ EN
                                                         │
                                                         │ KR+1.00000- IN1 OUT -AC2
                                                         │         AC1- IN2
                                                         │
                                                         │            MOV_R
                                                         ├──────────────
                                                         │ EN
                                                         │
                                                         │ KR+1.00000- in  OUT -VD3039
                                                         │
                                                         │
                                                         │              41
                                                         └──────────────(  JMP  )
```

```
LDR<=   VD3039,   1.0           // Если (Управляющий_сигнал > 1.0)
NOT                             //
MOVR    1.0,      AC2           // Mi = 1.0 - Ma
-R      AC1,      AC2           //
MOVR    1.0,      VD3039        // Управляющий_сигнал = 1.0
JMP     41                      //
```

```
    V3039.7                                               WXOR_DW
30 ──┤   ├──────────────────────────────────────────────┬──────────
                                                         │ EN
                                                         │
                                                         │         AC1- IN1 OUT -AC2
                                                         │ KH80000000- IN2
                                                         │
                                                         │            MOV_R
                                                         ├──────────────
                                                         │ EN
                                                         │
                                                         │ KR+0.00000- in  OUT -VD3039
                                                         │
```

```
LD      V3039.7                 // Иначе Если (Управляющий_сигнал < 0.0)
AC1     AC2                     //   Mi = -Ma
XORD    16#80000000,      AC2   //
MOVR    0.0,      VD3039        // Управляющий_сигнал = 0.0
```

```
 LBL: 41
    KR+0.00000    VD3055                                  42
32 ────────┤ == R ├──────────────────────────────────────(  JMP  )
```

```
LBL     41                      //
LDR=    0.0,      VD3055        // Если (И_коэф <> 0)
JMP     42                      //
```

```
33  |    AC2      KR+1.00000                                          MOV_R
    |-------| <= R |-----------|NOT|-------------------------------|EN
    |                                                  |
    |                                                  |  KR+1.00000-|in   OUT|-AC2
    |                                                  |
    |                                                  |
    |                                                  |
    |                                                  |
    |                                                  |-------------(  43
    |                                                  |             JMP  )
    |
```

```
LDR<=    AC2,     1.0          // Если (Mi > 1.0)
NOT                            //
MOVR     1.0,     AC2          //   Mi = 1.0
JMP      43                    //
```

```
34  |KH80000000      AC2                                              MOV_R
    |-------| == D |-------------------------------------------------|EN
    |                                  |
    |    AC2      KR+0.00000           |                  KR+0.00000-|in   OUT|-AC2
    |-------| >= R |-----------|NOT|--|
    |
```

```
LDD=     16#80000000,        AC2      // Иначе Если (Mi = -0.0) ИЛИ
LDR>=    AC2,     0.0                 //      (Mi < 0.0)
NOT                                   //
OLD                                   //
MOVR     0.0,     AC2                 //   Mi = 0.0
```

```
LBL: 43
    |
36  |-|  SM0.0                                                        MOV_R
    |  |-------------------------------------------------------------|EN
    |                                                                |
    |                                                          AC2-|in   OUT|-VD3047
    |                                                                |
    |                                                                |
```

```
LBL      43                    //
LD       SM0.0                 // Конец Если
MOVR     AC2,     VD3047       // Накопленное_значение = Mi
```

```
LBL: 42
    |
LBL: 40
    |
39  |-|  SM0.0                                                        V3030.7    K1
    |  |-------------------------------------------------------------(   R    )
    |
```

```
LBL      42                    //       Конец Если
LBL      40                    // Конец Если
LD       SM0.0                 //
R        V3030.7,  1           // внутреннее_состояние.первый_цикл = ЛОЖЬ
```

```
40  |-|  SM0.0                                                        MUL_R
    |  |-------------------------------------------------------------|EN
    |                                                                |
```

```
                                                  VD3039-IN1  OUT-AC0
                                                KR+32000.0-IN2
                                                          TRUNC
                                                          EN
                                                     AC0-in   OUT-AC1
                                                          MOV_W
                                                          EN
                                                     AC1-in   OUT-VW3006
                                                          ( RETI  )
```

// Преобразовать масштабированное значение управляющего сигнала назад в значение
// целое со знаком от 0 до 32000.

```
LD      SM0.0                   // текущий_Управляющий сигнал =
                                        // TRUNC(Управляющий_сигнал * 32000.0)
MOVR    VD3039,   AC0           //
*R      32000.0,  AC0           //
TRUNC   AC0,      AC1           //
MOVW    AC1,      VW3006        //
RETI                            // КОНЕЦ INT 100
```

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажіый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все прав защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC
S7-200 Tips

Group	Topic
1	Handling of the S7-200 Timers

Overview

Timers are used to regulate the timing of given functions within a program. The SIMATIC S7-200 series of programmable controllers features two types of timers: the On-Delay Timer (TON) and the Retentive On-Delay Timer (TONR). Both types of timers are available in three time base resolutions: 1 msec, 10 msec, and 100 msec.

This tip explains the operation and use of each timer type, with particular emphasis on the differences in timer operation between the various time base resolutions.

Figure 33.1

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

Description

General Information:

S7-200 timers are controlled with a single enabling input and have a current value that maintains the elapsed time since the timer was enabled. The timers also have a preset time value (PT) that is compared to the current value each time the current value is updated. A timer bit is set/reset based upon the result of this comparison. When the current value is greater than or equal to the preset time value, the timer bit (T-bit) is turned on. Otherwise, the T-bit is turned off. Timing stops when the current value reaches a maximum value.

When a timer is reset, its current value is set to zero, and its T-bit is turned off. Timers can be reset with the Reset instruction (this is the only way a TONR timer can be reset). Writing a zero to a timer's current value will not reset its timer bit. Likewise, writing a zero to a timer's T-bit will not reset its current value.

The two types of timers (TON and TONR) differ in the ways in which they react to the state of the enabling input. Both TON and TONR timers time up while the enabling input is on and do not time up while the enabling input is off. But when the enabling input is off, a TON timer will automatically reset, and a TONR timer will not reset. Therefore, the TON timer is best utilized when timing a single interval, while the TONR timer is more appropriate when it is necessary to accumulate a number of timed intervals.

TON and TONR timers are available in resolutions of 1 msec, 10 msec, and 100 msec, and have a maximum value of 32.767 seconds, 327.67 seconds, and 3276.7 seconds, respectively. The resolution is determined by the timer number, as shown in the chart below. Each count of the current value is a multiple of the time base. For example, a count of 50 on a 10 msec timer represents 500 msec.

	1 msec	10 msec	100 msec
TON	CPU 212: T32 CPU 214: T32, T96	CPU 212: T33-T36 CPU 214: T33-T36, T97-T100	CPU 212: T37-T63 CPU 214: T37-T63, T101-T127
TONR	CPU 212: T0 CPU 214: T0, T64	CPU 212: T1-T4 CPU 214: T1-T4, T65-T68	CPU 212: T5-T31 CPU 214: T5-T31, T69-T95

Figure 33.2

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

TON/TONR timers with 1 msec resolution:

In order to maintain the precision of the 1 msec timers, they are updated once per millisecond in the system interrupt routine that maintains the system time base. The current value is updated by incrementing it by 1, unless it is already at the maximum value.

The interrupt routine maintaining the 1 msec system time base is independent of the enabling and disabling of timers; this means a given 1 msec timer will be enabled at a point somewhere within the current 1 msec interval. Therefore, the timed interval for a given 1 msec timer can be up to 1 msec short. The user should program the preset time value to a value 1 greater than the minimum desired timed interval. For example, to guarantee a timed interval of at least 56 msec using a 1 msec timer, the preset time value should be set to 57.

Since the current value of an active 1 msec timer is updated in a system routine, the update is automatic. Once enabled, execution of a 1 msec timer's controlling TON/TONR instruction is required only to control the off/on state of the timer.

To keep the execution time of this system interrupt routine at a manageable level, only a small number of 1 msec timers can be supported. Since the 1 msec timers are updated from within an interrupt routine, the current value and the T-bits of these timers can be updated anywhere in the scan and will be updated more than once per scan if the scan time exceeds one millisecond. Therefore, these values are not guaranteed to remain constant throughout a given execution of the main user program.

A reset of an enabled 1 msec timer will turn it off, in addition to zeroing its current value and clearing its T-bit.

TON/TONR timers with 10 msec resolution:

In order to support a larger number of 10 ms timers with precision, all enabled 10 msec timers are updated at the beginning of each scan by adding a 10 msec accumulation value to the current value. This accumulation value is determined at the beginning of the scan and consists of the number of 10 msec intervals that have transpired since the last update (the beginning of the previous scan). This single accumulation value is applied to all enabled 10 msec timers.

The process of accumulating 10 msec intervals is performed independently of the enabling and disabling of timers, so a given 10 msec timer will be enabled at a point somewhere within the current 10 msec interval. Therefore, the timed interval for a given 10 msec timer can be up to 10 msec short. The user should program the preset time value to a value 1 greater than the minimum desired timed interval. For example, to guarantee a timed interval of at least 140 msec using a 10 msec timer, the preset time value should be set to 15.

Since the current value of an active 10 msec timer is updated at the beginning of the scan, the update is automatic. Once enabled, execution of a 10 msec timer's controlling TON/TONR instruction is required only to control the on/off state of the timer. Unlike the 1 msec timers, a 10 msec timer's current value is updated only once per scan and will remain constant throughout a given execution of the main user program.

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

A reset of an enabled 10 msec timer will turn it off, in addition to zeroing its current value and clearing its T-bit.

TON/TONR timers with 100 msec resolution:

The largest number of timers are 100 msec timers, and correspondingly, they have the lowest precision. All enabled 100 msec timers are updated when the enabling timer instruction is executed by adding a 100 msec accumulation value to the current value. This accumulation value, similar to the 10 msec accumulation value, is determined at the beginning of the scan and consists of the number of 100 msec intervals that have transpired since the last update (the beginning of the previous scan). This single accumulation value is applied to all enabled 100 msec timers as they are updated.

The process of accumulating 100 msec intervals is performed independently of the enabling and disabling of timers, so a given 100 msec timer will be enabled at a point somewhere within the current 100 msec interval. Therefore, the timed interval for a given 100 msec timer can be up to 100 msec short. The user should program the preset time value to a value 1 greater than the minimum desired timed interval. For example, to guarantee a timed interval of at least 2100 msec using a 100 msec timer, the preset time value should be set to 22.

The update of 100 msec timers is not automatic, since the current value of a 100 msec timer is updated only if the timer instruction is executed. Consequently, if a 100 msec timer is enabled, but the timer instruction is not executed each scan, the current value for that timer will not be updated and it will lose time. Likewise, if the same 100 msec timer instruction is executed multiple times in a single scan, the 100 msec accumulation value will be added to the timer's current value multiple times, and it will gain time. Therefore, 100 ms timers should only be used where the timer instruction will be executed exactly once per scan.

A reset of a 100 msec timer will zero its current value and clear its T-bit.

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

Examples:

The effect of updating the current value of 1 msec, 10 msec, and 100 msec timers at different times depends on how the timers are used. For example, consider the timer operation in the following program.

Automatically Re-triggered Oneshot Timer

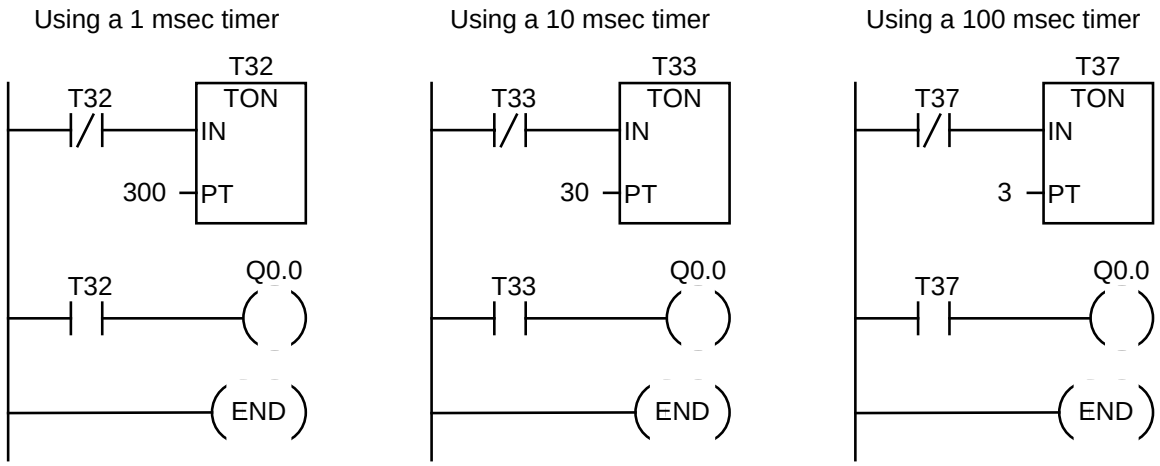


Figure 33.3

In the case where the 1 msec timer is used, Q0.0 will be turned on for one scan whenever the timer's current value is updated, after the normally closed contact T32 is executed, and before the normally open contact T32 is executed.

In the case where the 10 msec timer is used, Q0.0 will never be turned on, because the timer bit T33 will be turned on from the top of the scan to the point where the timer box is executed. Once the timer box has been executed, the timer's current value and its T bit will be set to zero. When the normally open contact T33 is executed, T33 will be off and Q0.0 will be turned off.

In the case where the 100 msec timer is used, Q0.0 will always be turned on for one scan whenever the timer's current value reaches the preset value.

A very simple modification can be made to the ladder programs above that will produce the same result for all three timers. This modification is shown below.

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

Automatically Re-triggered Oneshot Timer (Fixed)

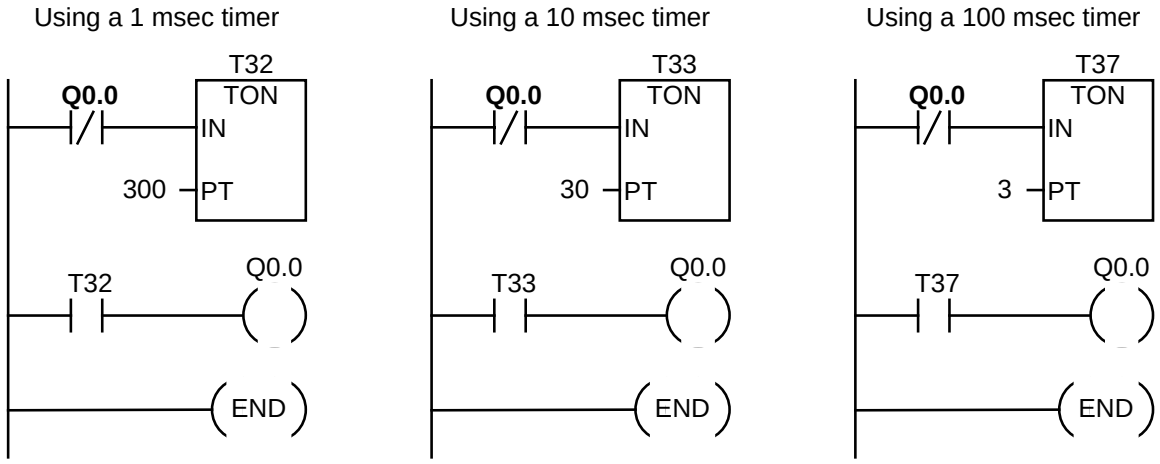


Figure 33.4

By using the normally closed contact Q0.0 instead of the timer bit as the enabling input to the timer box, the output Q0.0 is guaranteed to be turned on for one scan each time the timer reaches the preset value.

General Notes

The SIMATIC S7-200 Application Tips are provided to give users of the S7-200 some indication as to how, from the view of programming technique, certain tasks can be solved with this controller. These instructions do not purport to cover all details or variations in equipment, nor do they provide for every possible contingency. Use of the S7-200 Application Tips is free.

Siemens reserves the right to make changes in specifications shown herein or make improvements at any time without notice or obligation. It does not relieve the user of responsibility to use sound practices in application, installation, operation, and maintenance of the equipment purchased. Should a conflict arise between the general information contained in this publication, the contents of drawings or supplementary material, or both, the latter shall take precedence.

Siemens is not liable, for whatever legal reason, for damages or personal injury resulting from the use of the application tips.

All rights reserved. Any form of duplication or distribution, including excerpts, is only permitted with express authorization by SIEMENS.

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.

SIMATIC

S7-200 Примеры

Группа	Тема
6	Обработка аналоговых входов

Краткое описание

Данный пример - первое приближение к использованию аналогого модуля **EM235 3AI/1AQ** вместе с **CPU-212** или **CPU-214**. Данная программа выводит среднее значение для аналогого входа из данного числа выборок и производит проверку для определения значения выдаваемого на выходе. EM235 спараметрирован для диапазона (10V.

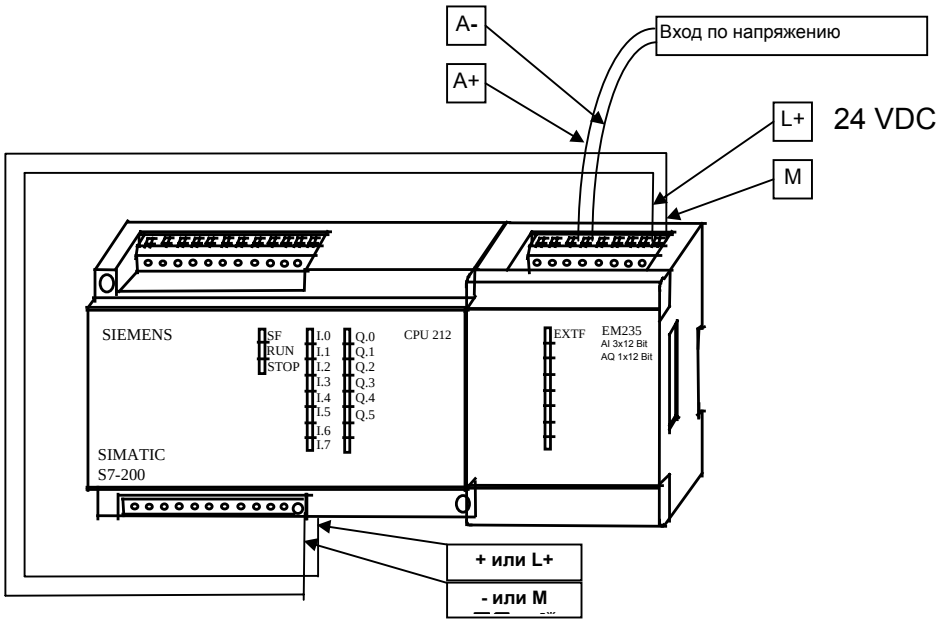


Рисунок 34.1

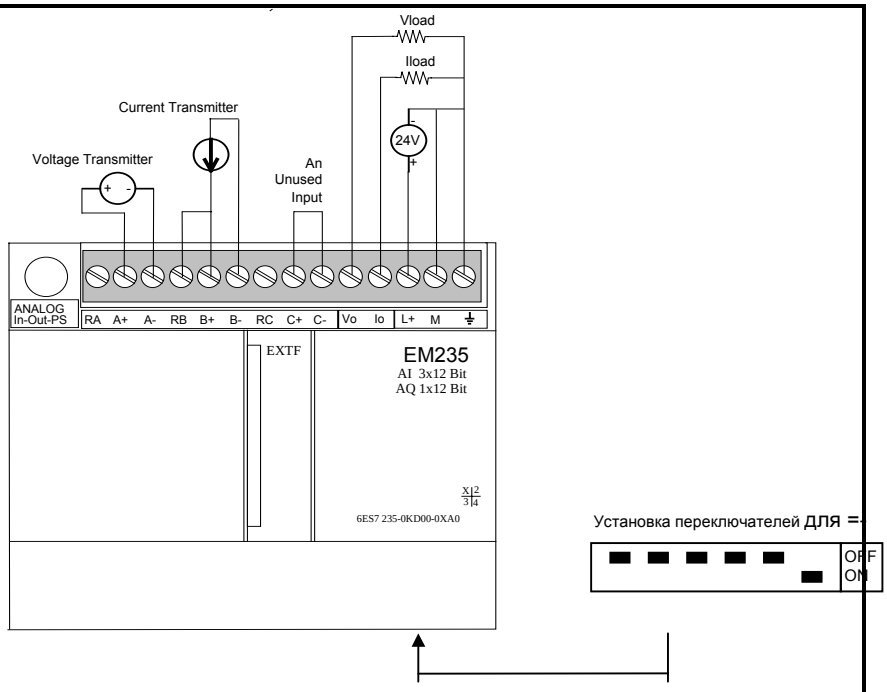
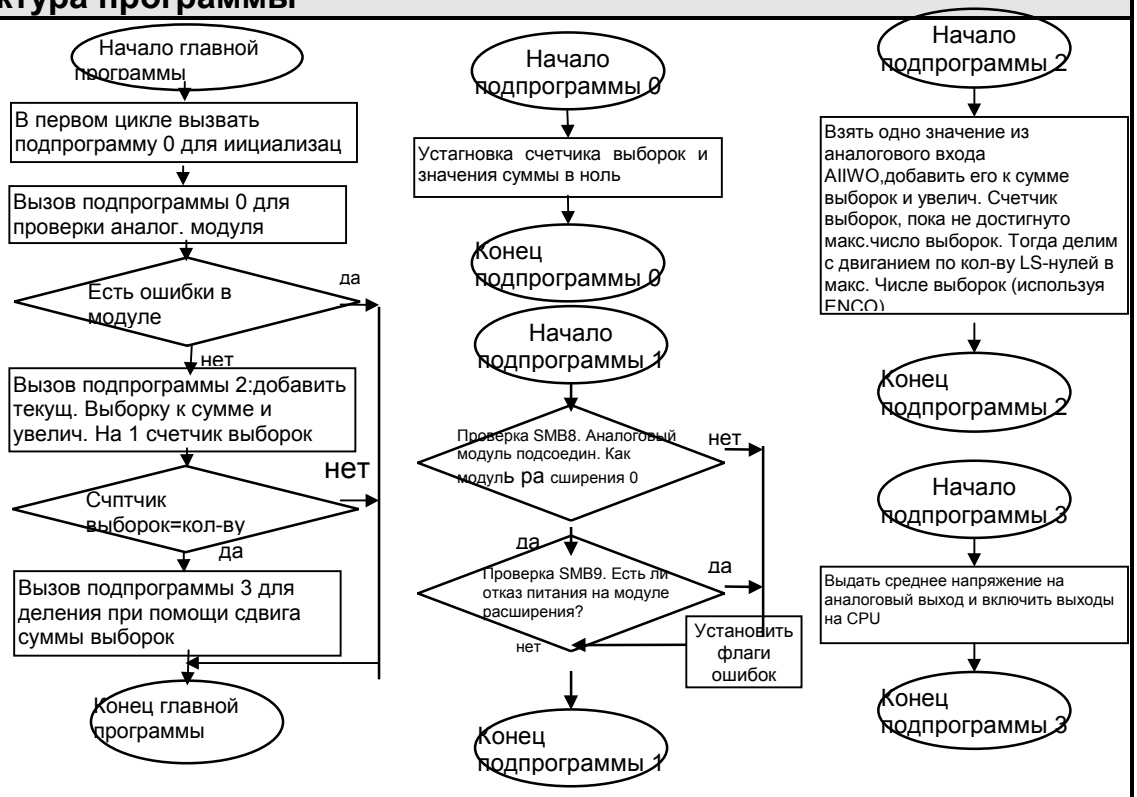


Рисунок 34.2

Структура программы



Описание программы

Данная программа илюстрирует функции аналогого модуля **EMN235** (3AI/1AQ) принимая вхлдные значения от AIW0, усредняя их для увеличения стабильности, и выдавая аналоговое напряжение на AQW0 в зависимости от вычисленного среднего.

Тестирование аналогого модуля обеспечивает информацией о ошибках модуля. Если первый модуль расширения не является аналоговым модулем, то выключается Q1.0. Другая ошибка, которую можно определить, это пропадание питания на аналоговом модуле; включается Q1.1 на CPU и EXTF на аналоговом модуле.

Данная программа использует однократное деление при помощи операции сдвига (используя 2 как число выборок), т.к. это занимает мало времени от времени цикла. Это число м.б. свободно изменено в диапазоне от 2 до 32768.

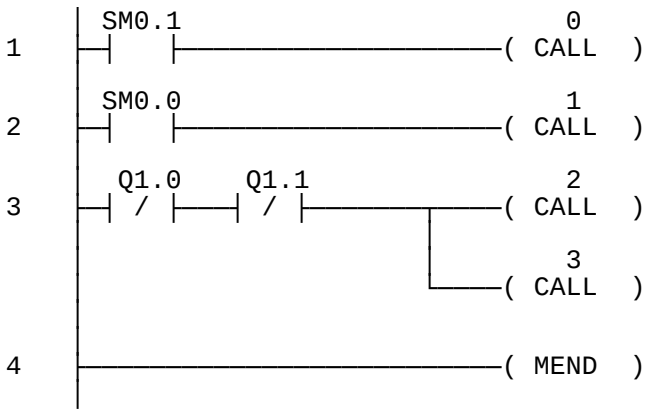
Размер слова входов 12 бит. Если число выборок больше, чем 16 (2^4), то размер суммы будет больше, чем целое слово (16 бит), тогда для хранения суммы необходимо использовать двойное слово (32 бита). Для того, чтобы добавить входное значение к сумме, необходимо преобразовать его в двойное слово.
Если входное значение - отрицательное, прибавить его к значащему слову; если - положительно, прибавить ноль для правильного значения.

Размер программы составляет 118 слов.

LAD (S7-MicroDOS)	STL (IEC)

Главная программа

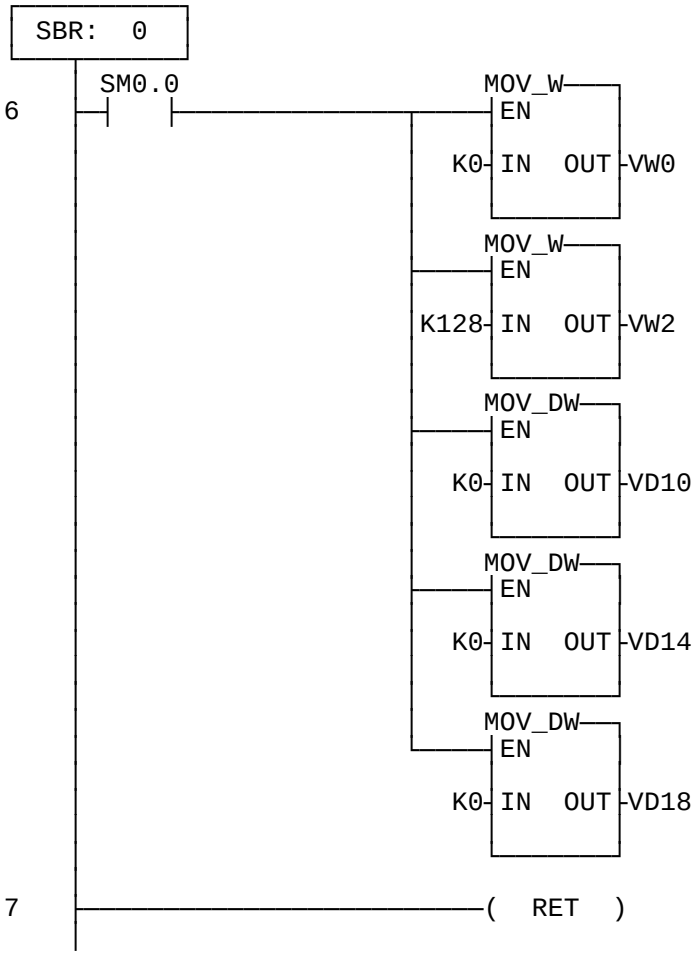
// Главная программа




```
LD      SM0.1   // Инициализация
CALL    0
LD      SM0.0   // Проверка аналогого
                // модуля
CALL    1

LDN     Q1.0    // Если нет ошибок в
AN      Q1.1    // аналоговом модуле
CALL    2       //  - Выбрать входное
                //    значение
CALL    3       //  - Выдать
        //  необходимое
        //  напряжение
MEND
```

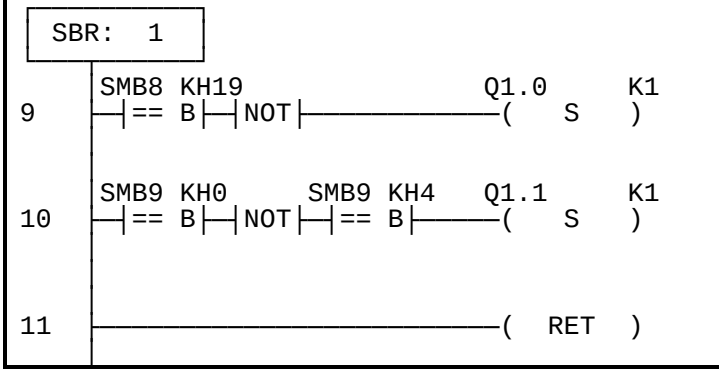
Подпрограммы

```
//*************************************************************
// Функция : Инициализация
//*************************************************************
```




```
SBR     0

LD      SM0.0
MOVW    0,VW0       // счетчик

MOVW    128,VW2     // Кол-во выборок

MOVD    0,VD10      // Текущая выборка

MOVD    0,VD14      // текущая сумма
                    // значений выборок

MOVD    0,VD18      // среднее значение

RET
```

```
//*************************************************************************
// Функция : Проверка, есть ли аналоговый модуль и раболает ли он.
//*************************************************************************
```




```
SBR     1

LDB=    SMB8,16#19
NOT                 // Проверка, есть ли
                    // модуль - как первый
S       Q1.0,1      // модуль расширения
LDB=    SMB9,16#00
NOT                 // Проверка, есть ли
                    // ошибка
AB=     SMB9,16#04
S       Q1.1,1      // Проверка,напряжение
                    // питания - OK ?
RET
```

```
// ВХОД
//*****************************************************************************
// Функция : Выборка аналогого значения : получение 1 значения от AIW0
//*****************************************************************************
```

```
 SBR:  2
    |
    | SM0.0                        MOV_W
13  |--|    |-----------------------|EN
    |                               |
```

SIMATIC
S7-200 Примеры

Группа	Тема
3	Связь между S7-200 и PC: Чтение из приложения Windows

Краткое описание

В данном примере используется программное обеспечение третьей фирмы для чтения данных из CPU серии SIMATIC S7-200 в приложение Windows. Имитируется простейшая насосная система, данные о которой передаются в различные ячейки Microsoft Excel.

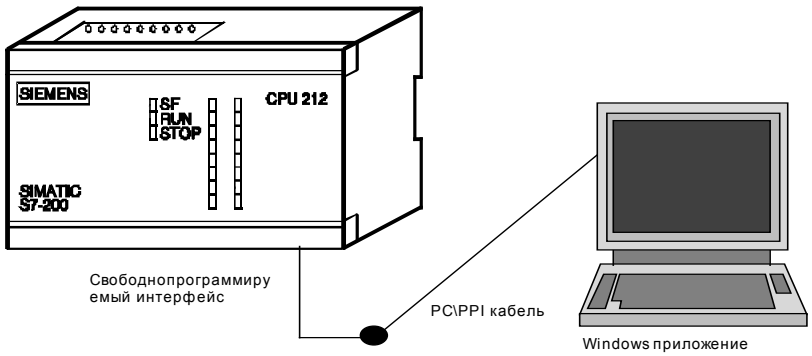
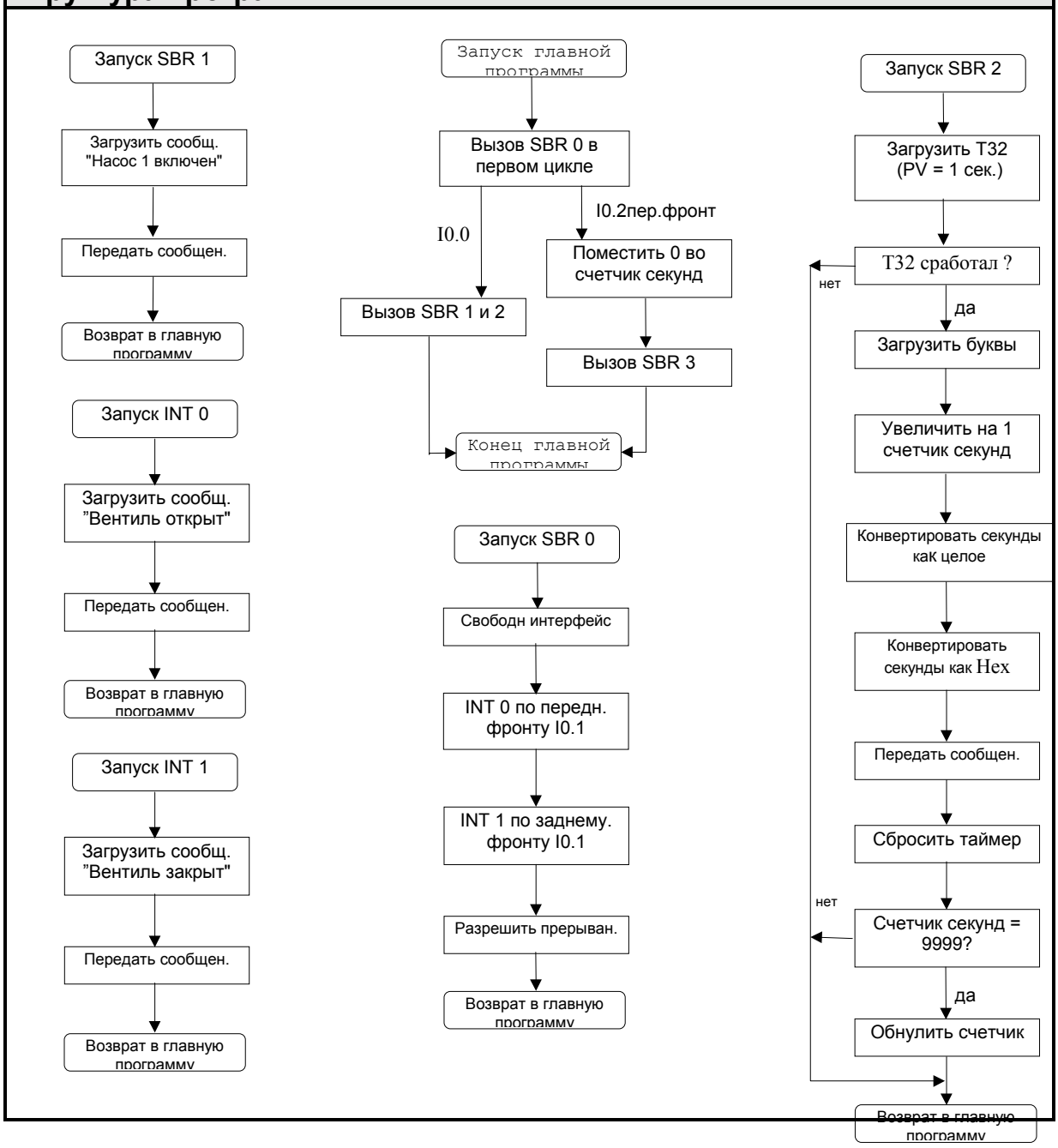


Рисунок 35.1

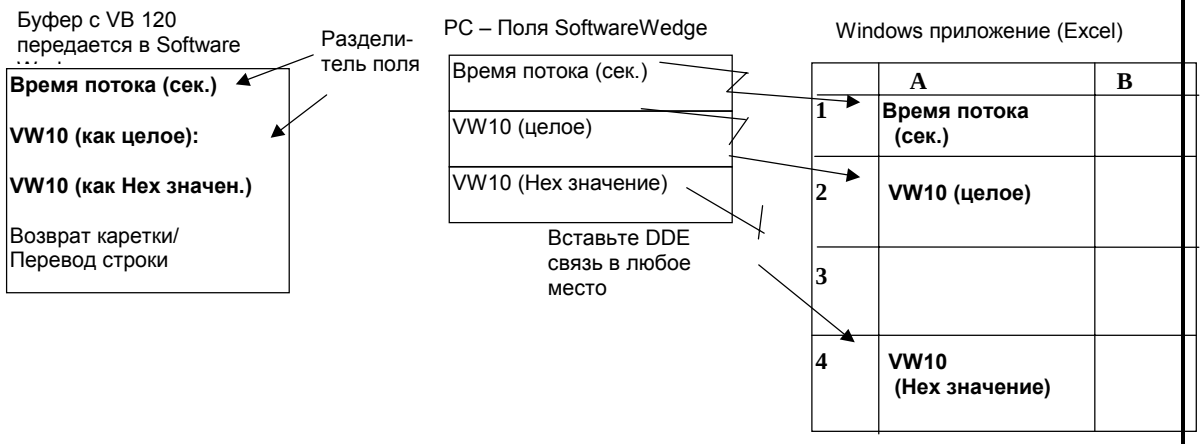
Аппаратные и программные требования

Аппаратное обеспечение:
SIMATIC CPU 214 или 212
Программное обеспечение:
SoftwareWedge for Windows: Professional Edition
Part # SW20WP или другое совместимое программное обеспечение
Любой пакет под Windows, который поддерживает связь по DDE

Структура программы



Описание программы



Программные пакеты (такие как SoftwareWedge for Windows) могут служить интерфейсом между SIMATIC CPU 214 и другими программами, работающими под Windows. Таким образом, информация из 214 м.б. отображена в любом приложении Windows, а так же информация м.б. записана в CPU из любогго приложения Windows.

В данный момент, SoftwareWedge не поддерживает отображение и обновление передаваемой информации из различных входов в различное время в различные части экрана. Однако, различные части информации из 214 м.б. переданы и отображены в различных местах. Каждая часть м.б. отображена в собственном поле SoftwareWedge. Каждое поле отделяется соответствующим знаком разделения, передаваемого вместе с телеграммой. Этот знак м.б. любым по желанию пользователя. Так же, в конце каждой передачи должен быть принят один или более знак "окончания", который м.б. задан пользователем.

После загрузки пакета SoftwareWedge выберите режим DDE сервера и задайте имя, тему и подходящий элемент приложения DDE. Затем настройте установки в списке Port на скорость передачи 9600 Бод, без четности 8 Бит/знак, и 1 стоповый бит. Так же не забудьте установить номер Com порта. Затем Вы должны определить структуры записей, которые будут вводиться. Для нашей программы началом записи является любой принятый знак, окончанием - знаки возврат каретки и перевод строки; максимальное число полей = 3, а разделителем является знак двоеточия (: (ASCII 58). И наконец в приложении Windows используйте команду связи Копировать/Вставить, чтобы вставить различные поля данных в желаемые части экрана.

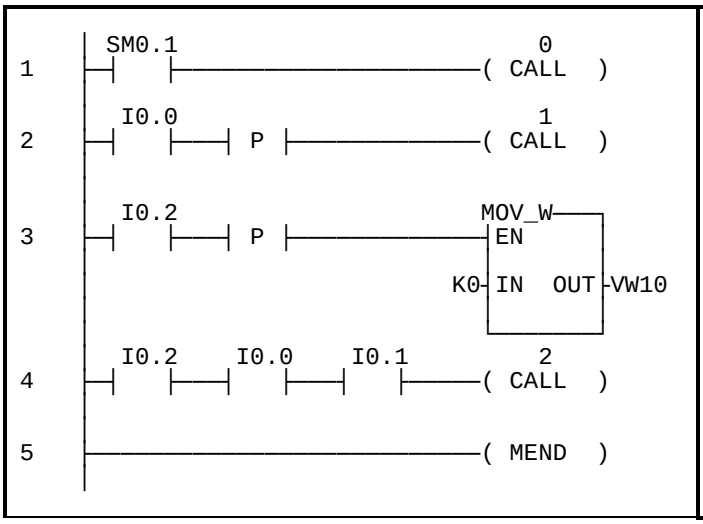
Опционально: Software Wedge поддерживает автоматическое изменение формата переменных перед тем, как передать их в другое приложение Windows.

Размер программы = 158 слов.

LAD (S7-MicroDOS)	STL (IEC)

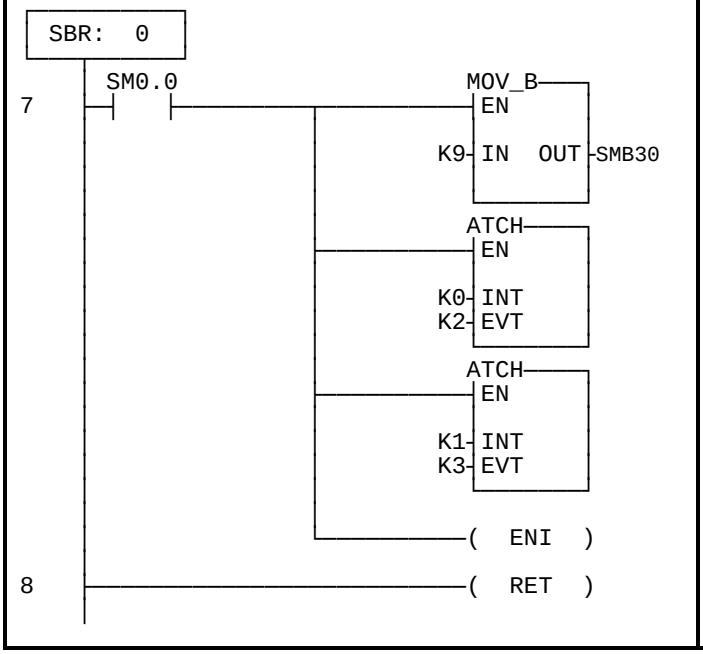
Главная программа

```
// Title= Интерфейс Windows: Чтение в Windows

// Данная программа - пример того, как CPU серии S7_200 может передавать данные
// в любую программу под Windows, используясing соответствующее программное
// обеспечение (см. выше).

// В данном примере используется простая программа для насосной системы.
// Предполагается, что I0.0 включает главный насос. Потом I0.1 открывает или
// закрывает аварийный вентиль. I0.2 открывает или закрывает главный вентиль,
// который пускает поток жидкости.

// Оператор, сидящий за компьютором, должен видеть сообщение, если изменяются
// эти три параметра. Для I0.0 появляется статическое сообщение " Насос 1 включен”.
// I0.1 контролирует сообщение, которое может принимать значение: “Вентиль открыт“
// или ”Вентиль закрыт". I0.2 отображает время потока для жидкости.
// Данное сообщение меняется каждую секунду, когда I0.2 включен.

// Данная программа не позволяет отображать всю информацию одновременно, а только
// если состояние каждого переключателя изменяется. Однако, в программу м.б. внесены
// несколько изменений, позволяющих отображать одновременно всю информацию. Для
// этого требуется передача ВСЕХ данных (даже тех, которые не изменились) каждый раз,
// когда происходит новая передача.
// В данной прогрмме в качестве разделителя используется знак двоеточия (:), а для
// окончания передачи - возврат каретки и перевод строки.

// С помощью данной программы делается попытка записать некоторое количество
// различной информации вприложение Windows. Для того, чтобы дать пользователю
// большое количество различных примеров в программу включены: статическое
// сообщение "Насос 1 Включен", сообщение с одним изменяющимся словом "Вентиль
// открыт/закрыт" и постоянно меняющийся вход в виде десятичного и
// шестнадцатиричного значения “Время потока (сек.) ####”.

// Следующая таблица показывает, какие значения для переменных в памяти,
// используются в программе.

// VW10          Главный счетчик в памяти, показывающий число секунд потока
// VW20          Второй счетчик в памяти - копия VW10, использующаяся в SBR 3 для
//                     преобразования IBCD, позволяющая не стирать значение главного
//                     счетчика.

// VB80          Содержит число 14 или число букв, хранящихся как
//                     шестнадцатиричное значение в таблице.
//                     Используется т.к. требуется для команды XMT.
// VD81 - VW93   Сообщение: "Насос 1 включен"

// VB100               Содержит число 12 или 14, в зависимости от состояния вентиля.
//                     Описывает число шестнадцатиричных значений в таблице.
// VD101 - VD109 Сообщение: "Вентиль открыт” или "Вентиль закрыт"
//       или
// VD101 - VW113

// VB120               Содержит число 28 или число шестнадцатиричных значений в
//                     следующей таблице.
// VD121 - VD133 Сообщение: “Время потока (сек.) ####”
// VB137               Содержит двоеточие в виде шестнадцатиричного значения для
//                     разделительного поля
// VB138 - VB141 Содержит значение второго счетчика, как ASCII значение,
//                     которое отображается в Windows как десятичное целое
// VB142               Содержит двоеточие в виде шестнадцатиричного значения для
//                     следующего разделительного поля
// VB143 - VB146 Содержит значение второго счетчика, как ASCII значение,
//                     которое отображается в Windows как шестнадцатиричное значение
// VW147               Возврат каретки, перевод строки - конец передаваемого сообщения
```



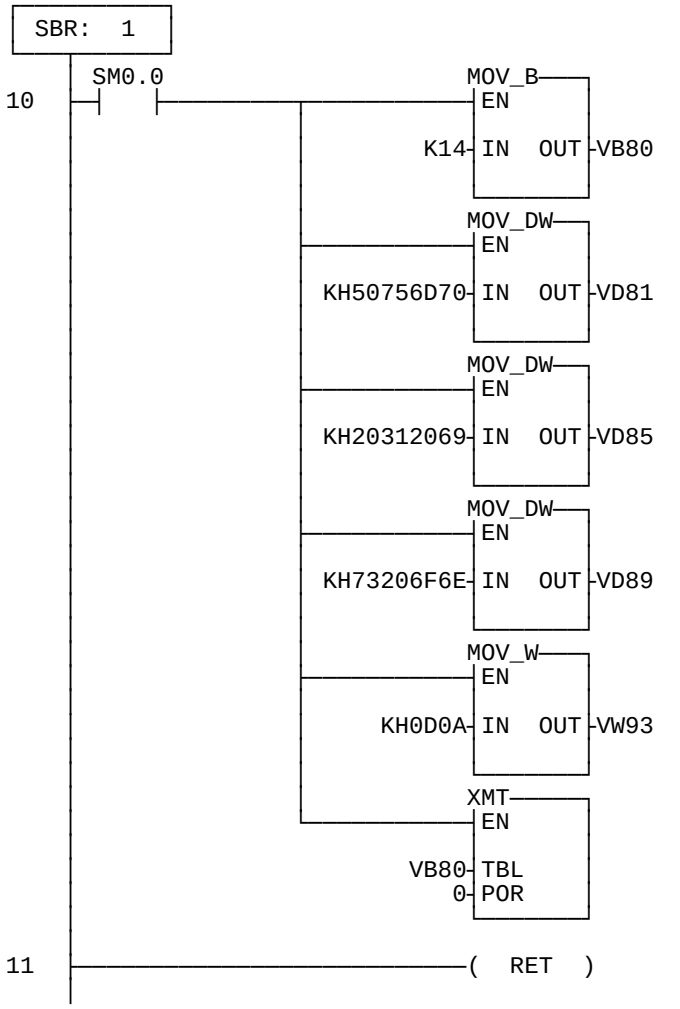

```
LD      SM0.1     // Бит первого цикла
CALL    0         // Вызов SBR 0

LD      I0.0      // Включатель насоса
EU                // По передн. фронту
CALL    1         // Вызов SBR 1

LD      I0.2      //Переключатель потока EU
                  // По передн. фронту
MOVW    0,VW10    // Записать 0 в таймер

LD      I0.2      //Переключатель потока
A       I0.0      // Насос д. работать
A       I0.1      // Вентиль д.б. открыт
CALL    2         // Вызов SBR 2
MEND              // Конец главной
                  // программы
```

Подпрограммы



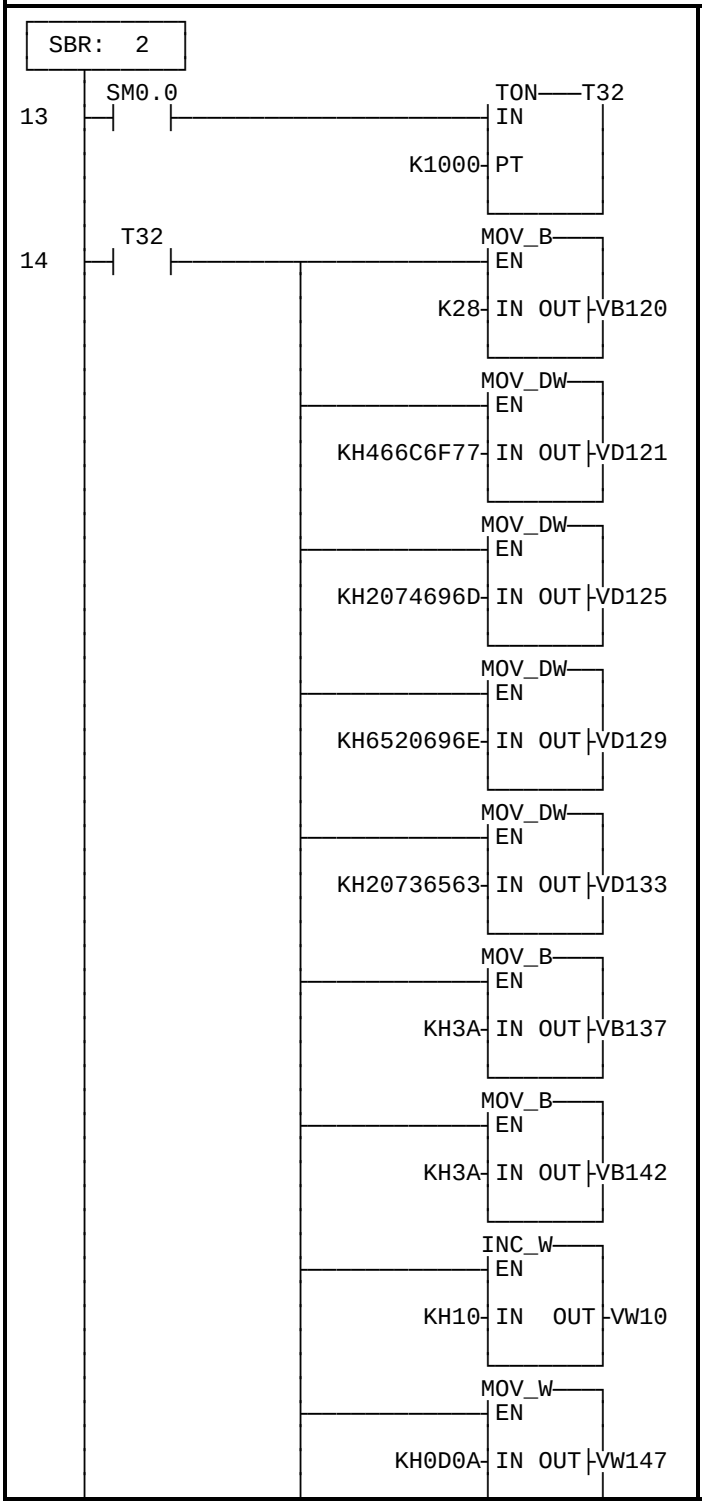

```
SBR     0         // Подпрограмма
                  // инициализации
LD      SM0.0     // Бит всегда „1“
MOVB    9,SMB30
                  // Режим свободнопрог-
                  // раммируемого
                  // интерфейса, 9600
                  // Бод, без четности,
                  // 8 Бит/знак

ATCH    0,2       // Прерывание 0
                  // привязано к событию
                  // 2 (перед. фронт I0.1)

ATCH    1,3       // Прерывание 1
                  // привязано к событию
                  // 3 (задний фронт I0.1)

ENI               // Разрешить
                  // прерывания
RET               // Возврат в главную
                  // программу
```

// Каждый раз по переднему фронту I0.0, в таблицу, начиная с VB80 загружается сообщение
// “Насос 1 включен” и передается по свободнопрограммируемому интерфейсу.

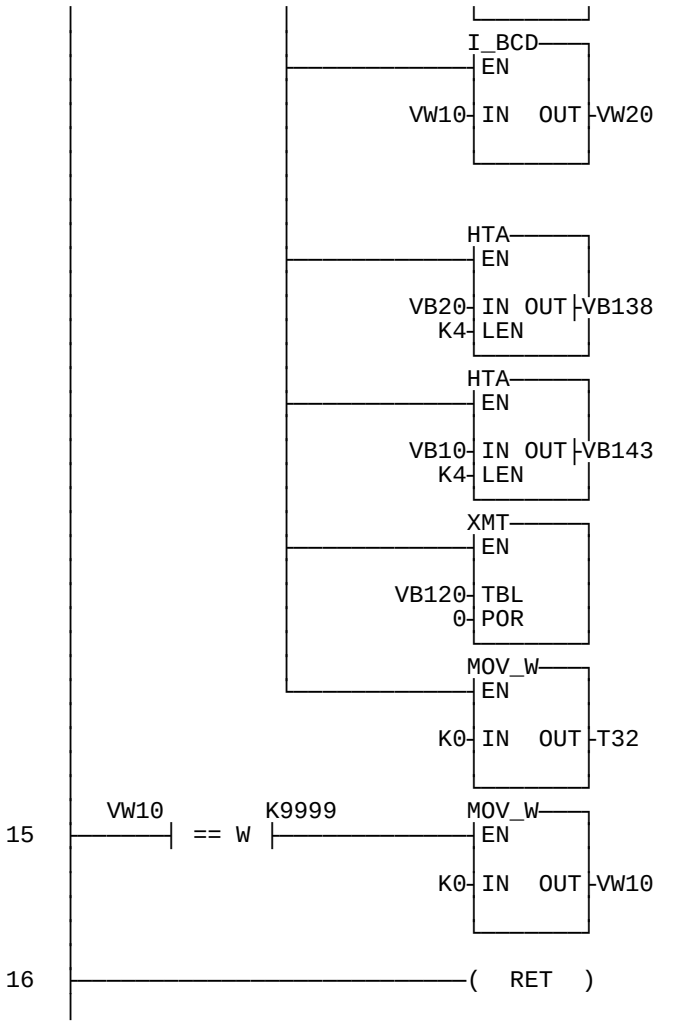



```
SBR    1          // Запись “Насос 1
                  // включен"

LD     SM0.0      // Бит всегда „1“
MOVB   14,VB80    // Загрузить длину
                  // сообщения в
                  // таблицу

MOVD   16#50756D70,VD81
                  // Сообщение в Hex -
                  // “Pump”

MOVD   16#20312069,VD85
                  // “ 1 i”

MOVD   16#73206F6E,VD89
                  // “s on”

MOVW   16#0D0A,VW93
                  // Возврат каретки и
                  // перевод строки -
                  // сигнал конца
                  // сообщения

XMT    VB80,0     // Буфер передачи
                  // VB80

RET               // Возврат в главную
                  // программу
```

```
// Когда активирован I0.2, загружается T32 (1 msec таймер TON) с предустановленным
// значением 1000, или 1 сек. Когда T32 сработет, то VW10 увеличивается на 1, и новое
// значение загружается в VW20. VW20 конвертируется из целого в двоично-десятичный
// код, а затем в ASCII значение; после этого копируется в соответствующее слово
// таблицы передачи.  VW10 так же копируется в соответствующее место таблицы, и
// шестнадцатиричное значение конвертируется в ASCII для передачи как текущее
// значение. В конце подпрограммы производитсч проверкаr для
// сброса VW10, если оно достигает значения 9999. Это делается для того, что команда
// BCDI поддерживает преобразование только слова (или 4 шестнадцатиричных значения).
// Если необходимо, то любое число м.б. сконвертировано.

// Таблица, которая начинается с VB120, актуализируется каждую секунду (по таймеру T32)
// для чтения "Время потока в сек. (Целое значение) (шестнадцатиричное значение)".
```



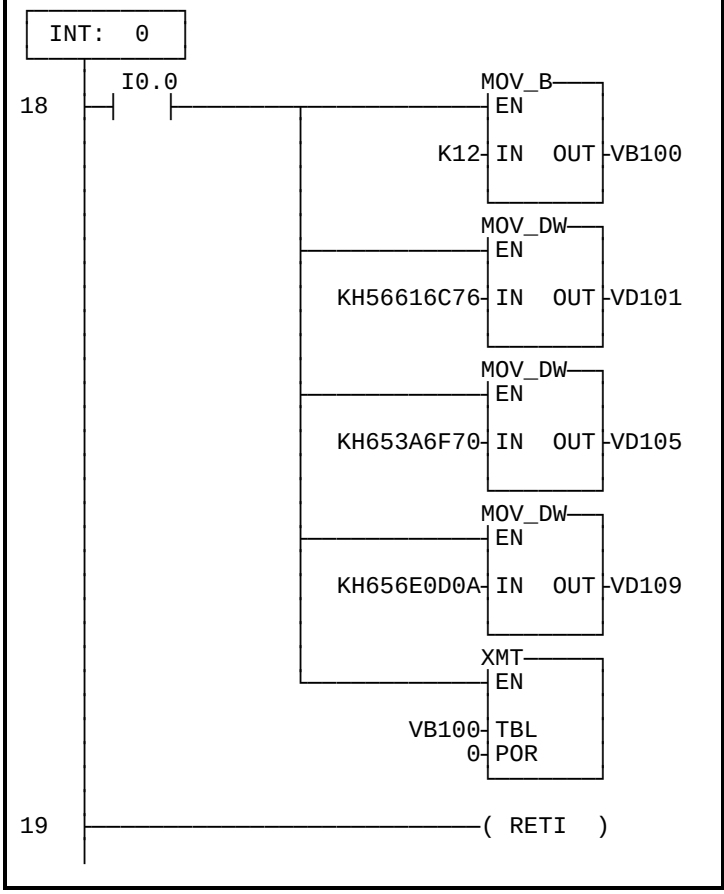

```
SBR     2
LD      SM0.0       // Бит всегда „1“
TON     T32,1000    // Загрузить таймер

LD      T32         // После 1 сек.
MOVB    28,VB120    // Загрузить значение
                    // 28 в таблицу

MOVD    16#466C6F77,VD121
                    // Сообщение в Hex -
                    // “Flow”

MOVD    16#2074696D,VD125
                    // “ tim”

MOVD    16#6520696E,VD129
                    // “e in”

MOVD    16#20736563,VD133
                    // “ sec”

MOVB    16#3A,VB137
                    // “:” - разделитель
                    // поля

MOVB    16#3A,VB142
                    // “:” - разделитель
                    // поля

INCW    VW10        // Увеличить на 1
                    // значение VW10

MOVW    16#0D0A,VW147
                    // Возврат каретки и
                    // перевод строки -
                    // сигнал конца
                    // сообщения
```



MOVW	**VW10,VW20**	
		// Скопировать
		// значение в VW20
IBCD	**VW20**	
		// Сконвертировать в
		// BCD
HTA	**VB20,VB138, 4**	
		// Сконвертировать из
		// BCD в ASCII (для
		// цифрового дисплея
		// и загрузить в
		// таблицу
HTA	**VB10,VB143, 4**	
		// Сконвертировать Hex
		// значение в ASCII (для
		// Hex дисплея) и
		// загрузить в таблицу
XMT	**VB120,0**	**// Передать таблицу в**
		// VB120
MOVW	**0,T32**	**// Сбросить таймер**
LDW=	**VW10,9999**	
		// Если поток > 9999
		// сек.
MOVW	**0,VW10**	**// Сбросить второй**
		// счетчик
RET		**// Возврат в главную**
		// программу

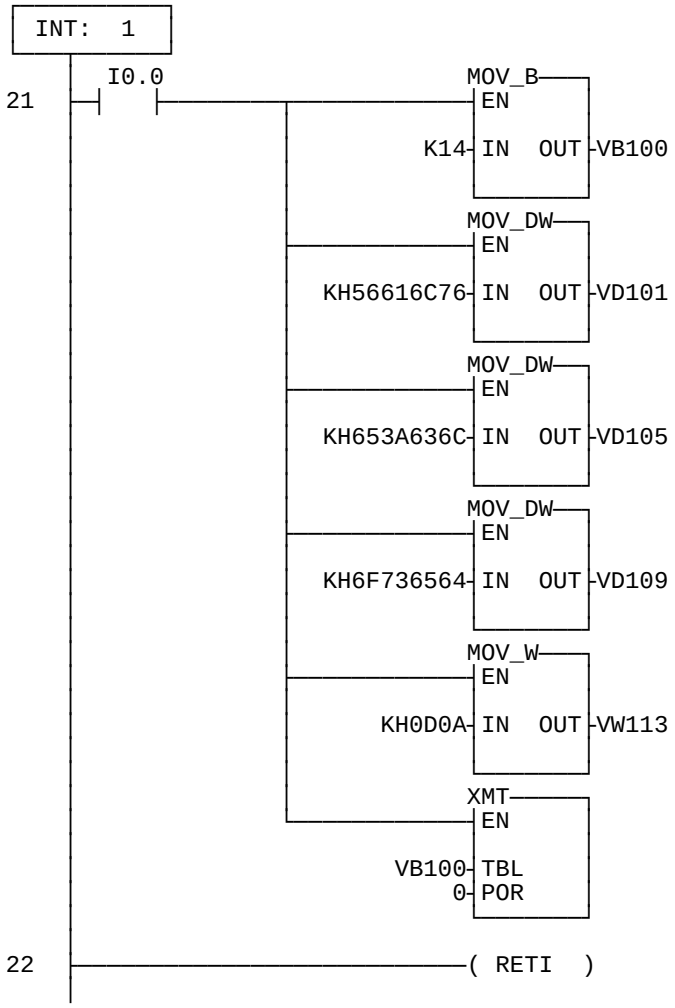
Программы обработки прерываний

// Программа обработки прерывания 0 срабатывает по переднему фронту I0.1.
// Поэтому сообщение актуализируется, только если вентиль изменил состояние.
// Передается сообщение “Вентиль открыт”




```
INT     0
LD      SM0.0        // Бит всегда „1“
MOVB    12,VB100     // Загрузить значение
                     // 12 в таблицу

MOVD    16#56616C76,VD101
                     // Сообщение в Hex -
                     // “Valv”

MOVD    16#653A6F70,VD105
                     // “e:op” - ‘:’ -
                     // разделитель поля

MOVD    16#656E0D0A,VD109
                     //  “en” Возврат
                     // каретки и
                     // перевод строки -
                     // сигнал конца
                     // сообщения

XMT     VB100,0      // Передать таблицу в
                     // VB 100

RETI                 // Возврат в главную
                     // программу
```

// Программа обработки прерывания 1 передает сообщение “Вентиль закрыт” по заднему
// фронту I0.1. Сообщение актуализируется, только если вентиль изменил состояние.




```
INT     1
LD      SM0.0        // Бит всегда „1“
MOVB    14,VB100     // Загрузить значение
                     // 14 в таблицу

MOVD    16#56616C76,VD101
                     // Сообщение в Hex -
                     // “Valv”

MOVD    16#653A636C,VD105
                     // “e cl”

MOVD    16#6F736564,VD109
                     // “osed”

MOVW    16#0D0A,VW113
                     // Возврат каретки и
                     // перевод строки -
                     // сигнал конца
                     // сообщения

XMT     VB100,0      // Передать таблицу в
                     // VB100

RETI                 // Возврат в главную
                     // программу
```

Другие замечания

SoftwareWedge™ for Windows - зарегистрированная торговая марка TAL Enterprises и защищена законами USA. Другие авторские права и информация о ограничениях смотрите в лицензионном Соглашении в руководстве SoftwareWedge. Если у Вас есть вопросы по поводу использования или функционирования SoftwareWedge, пожалуйста звоните или пишите:

T.A.L. Enterprises
2022 Wallace Street
Philadelphia, PA 19130 U.S.A.
Тел: (215)-763-2620 Факс: (215)-763-9711

Windows™ - зарегистрированная торговая марка Microsoft, защищена законами USA и других стран.

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Коментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены коментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все права защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC
S7-200 Примеры

Группа	Тема
6	**Измерение и мониторинг температуры с использованием термосопротивления PT100**

Краткое описание

В данном примере обсуждается измерение температуры и мониторинг указанных пределов с использованием аналогого модуля расширения EM235, к одному из каналов которого подключено термосопротивление PT100.

Для того, чтобы преобразовать зависимое от температуры изменение сопротивления PT100 в напряжение, используется аналоговый выход как источник постоянного тока. Выход питает датчик PT100 постоянным током 12.5mA. С помощью этой цепи генерируется линейное входное напряжение 5mV/°C. EM235 преобразует данное напряжение в цифровое значение, которое циклически читается программой. Из прочитанного значения программа вычисляет температуру [(C], используя следующую формулу:

$$T[°C] = \frac{\text{Цифровое Значение - 0(C-Смещение}}{\text{1°C-Значение}}$$

Цифровое значение = значение, хранящееся в AIWx (x=0,2,4).
0°C-Смещение = цифровое значение, измеренное при 0°C.
В нашем примере это смещение = 4000.
1°C-Значение = изменение значения при увеличении температуры на 1°C.
В нашем примере 1°C-Значение = 16.

Программа вычисляет значение до первой десятичной запятой и записывает результат в встроенную переменную Сообщения 1: ''Температура = xxx.x°C'', которое отображается на TD200. В сегменте инициализации программы пользователь может ввести верхний и нижний пределы температуры. Программа контролирует измеряемое значение и отображает предупреждение на TD200, если измеренная температура вышла из указанного диапазона. Если измеренная температура превысила вернюю границу температуры, то на второй строке дисплея TD200 появляется Сообщение 2: ''Температура > xxx.x°C''; если измеренная температура меньше нижнего предела температуры, то отображается Сообщение 3: ''Температура < xxx.x°C".

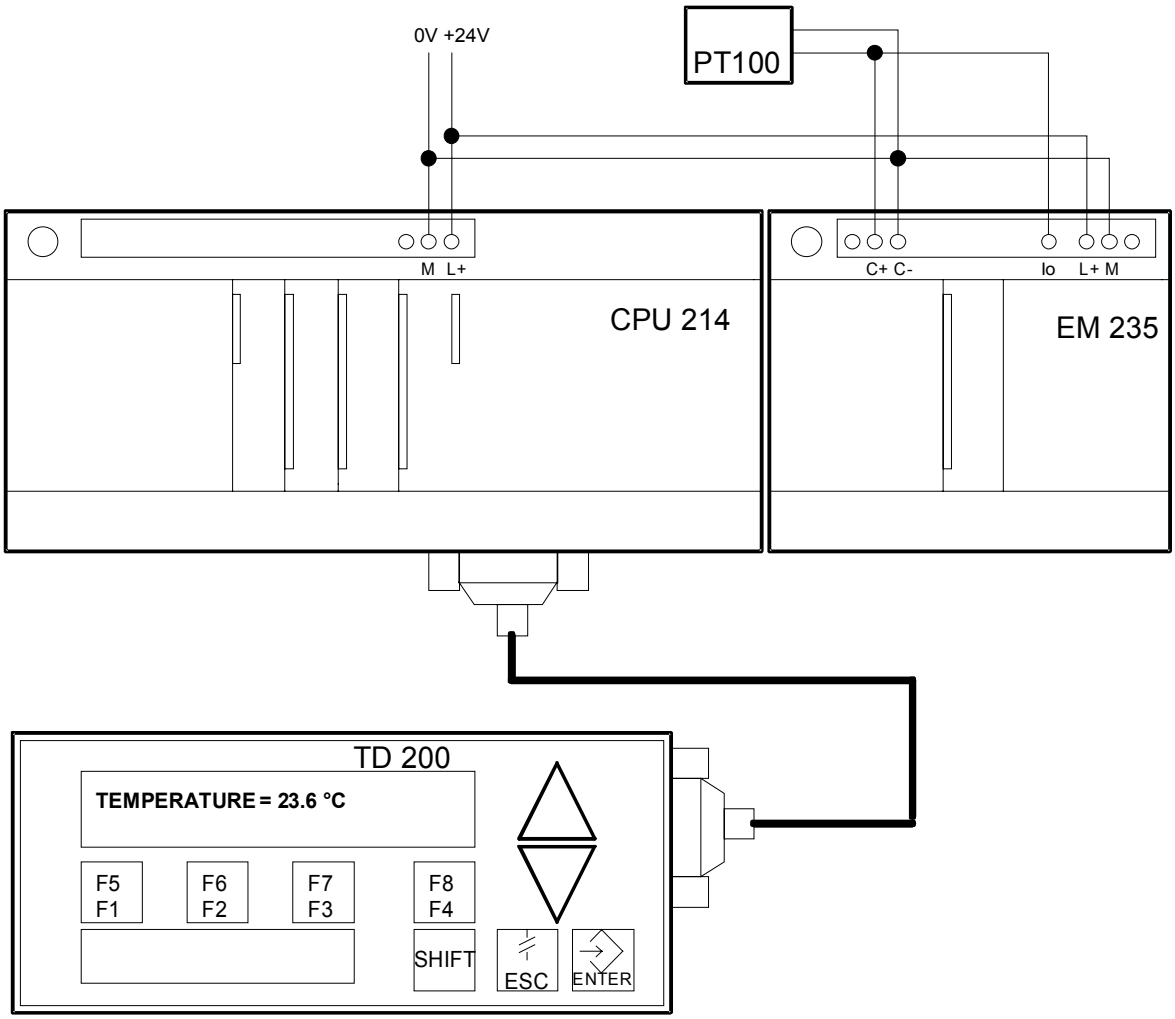
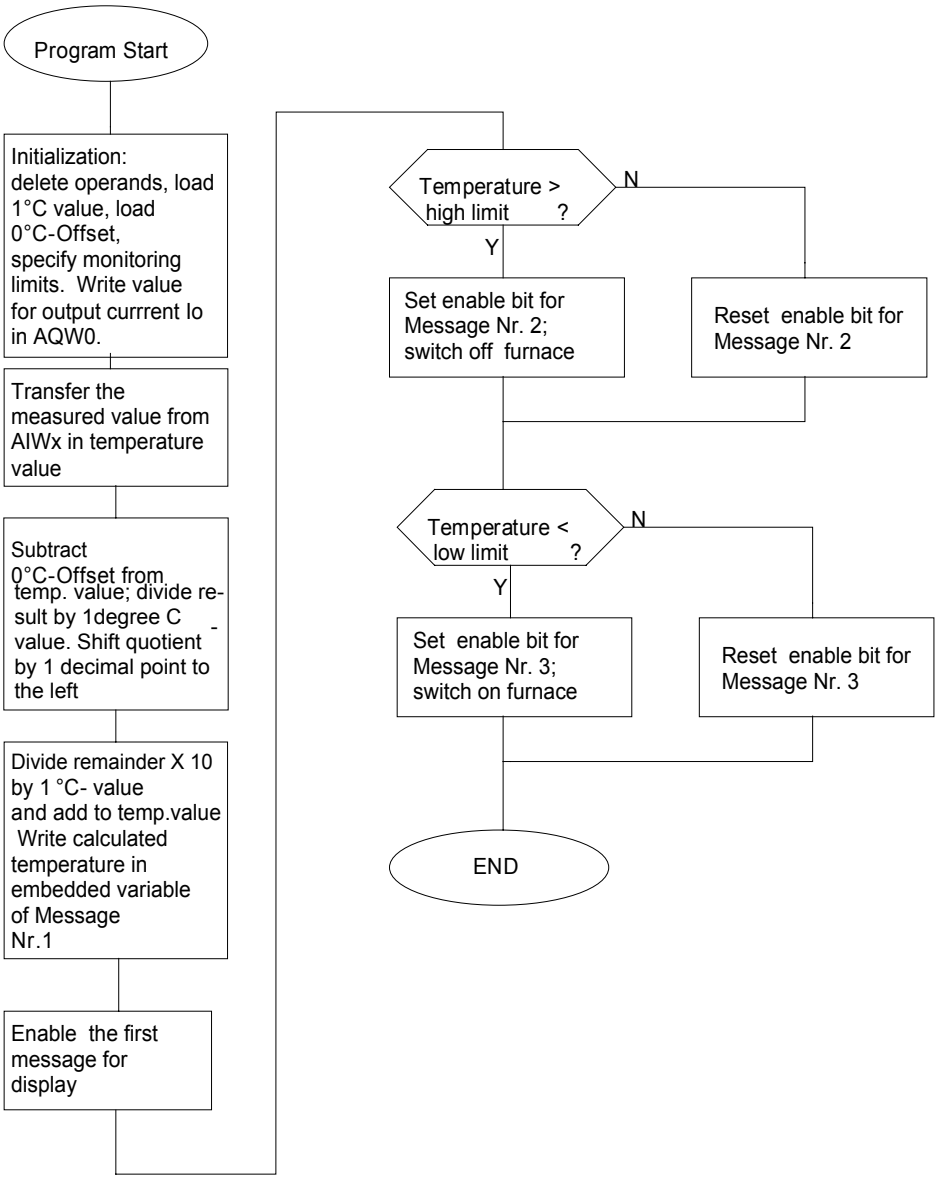

0V +24V
PT100
M L+
CPU 214
C+ C-
Io L+ M
EM 235
TD 200
TEMPERATURE = 23.6 °C
F5
F1
F6
F2
F7
F3
F8
F4
SHIFT
ESC
ENTER

Рисунок 36.1

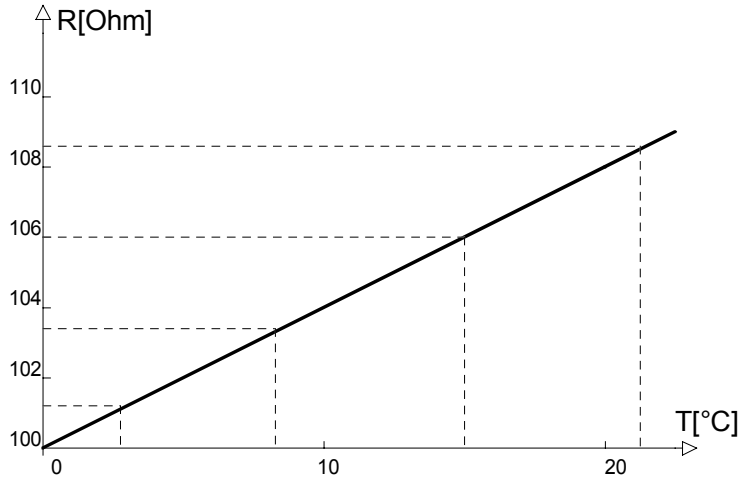
Структура программы



Описание программы

Датчик температуры:
PT100 - платиновое термосопротивление для измерения температуры в диапазоне от -60 до +400°C.

Вычисление потребляемого тока для PT100:
PT100 при температуре 0C° имеет сопротивление 100 Ом. Сопротивление с ростом температуры изменяется линейно - около 0.4 Ом на градус Цельсия.



Для генерации коэффициента напряжения 5mV/°C, необходим источник тока 12.5mA. Разрешение аналогого выхода - 10uA/Число, таким образом необходимое значение числа д.б. 1250, для того чтобы получить 12.5mA. Т.к. формат слова данных AQW сдвигается вправо на 4 бита, значение числа умножается на 16. Поэтому для инициализации тока 12.5mA для аналогого выхода I_o, в AQW0 заносится 20000.

Формула: (32000/20mA * 12.5mA = 20000)

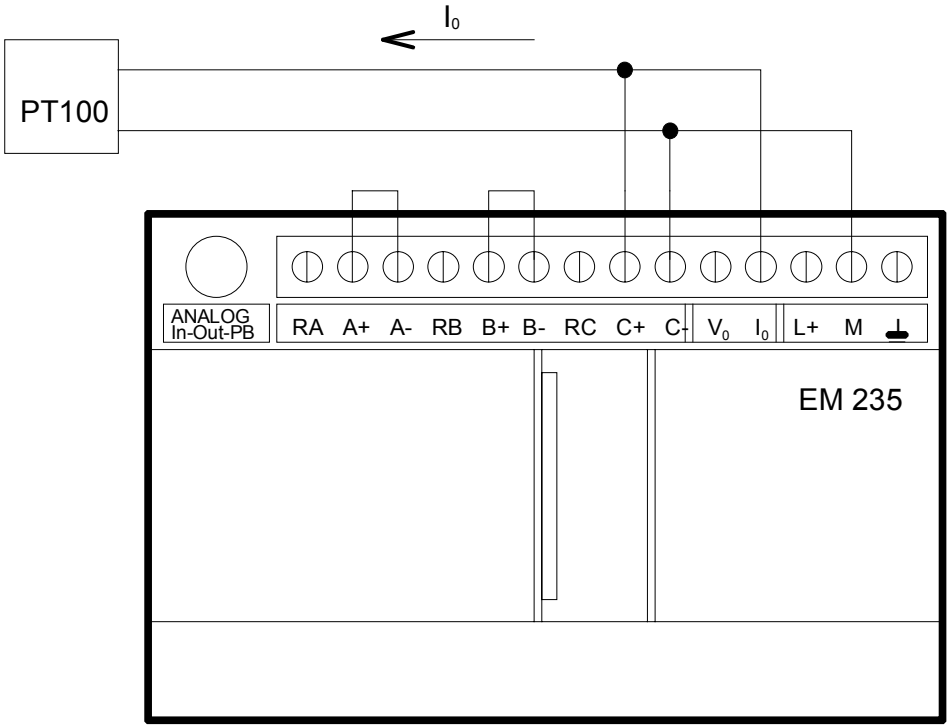


Рисунок 36.2
EM235 Цепь

На модуле EM235 при помощи конфигурационных переключателей выбран диапазон напряжения 0..10V:

Конфигурационный переключатель №:	1	3	5	7	9	11
	ON	OFF	OFF	OFF	ON	OFF

Соответствующий адрес AI слова д.б. использован в программе в зависимости от номера канала на EM235,:

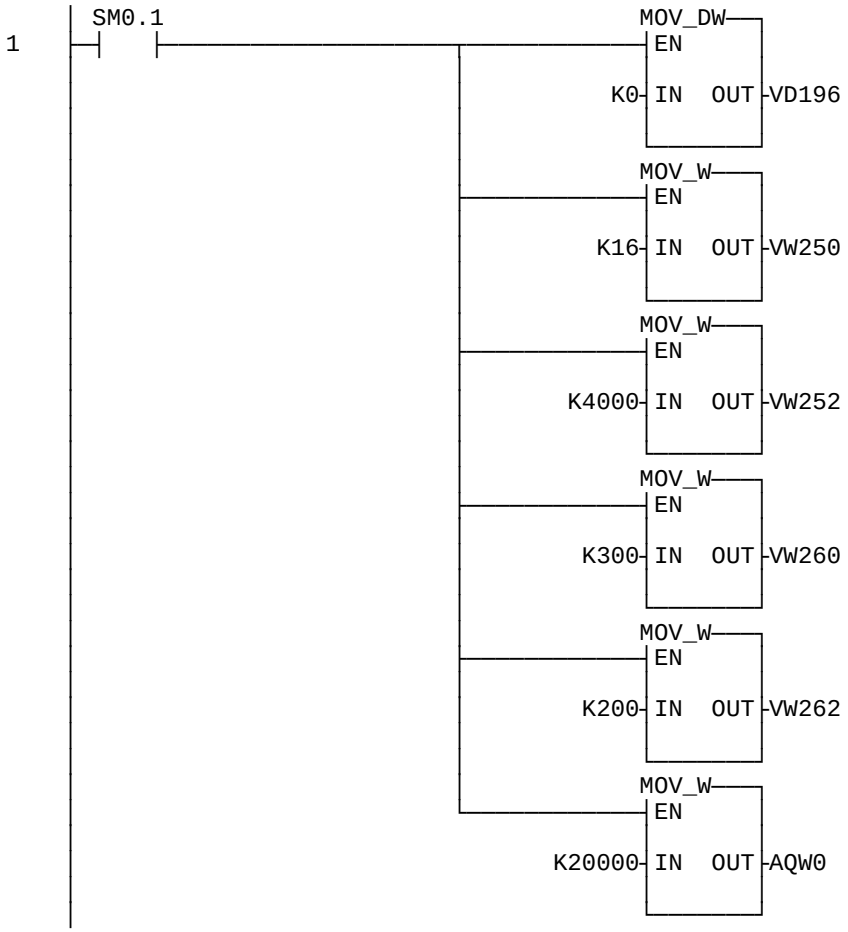
AIW0 для входного канала 1, AIW2 для входного канала 2,
AIW4 для входного канала 3, AQW0 для входного канала 1

Неиспользуемые входы EM 235 д.б. закорочены.

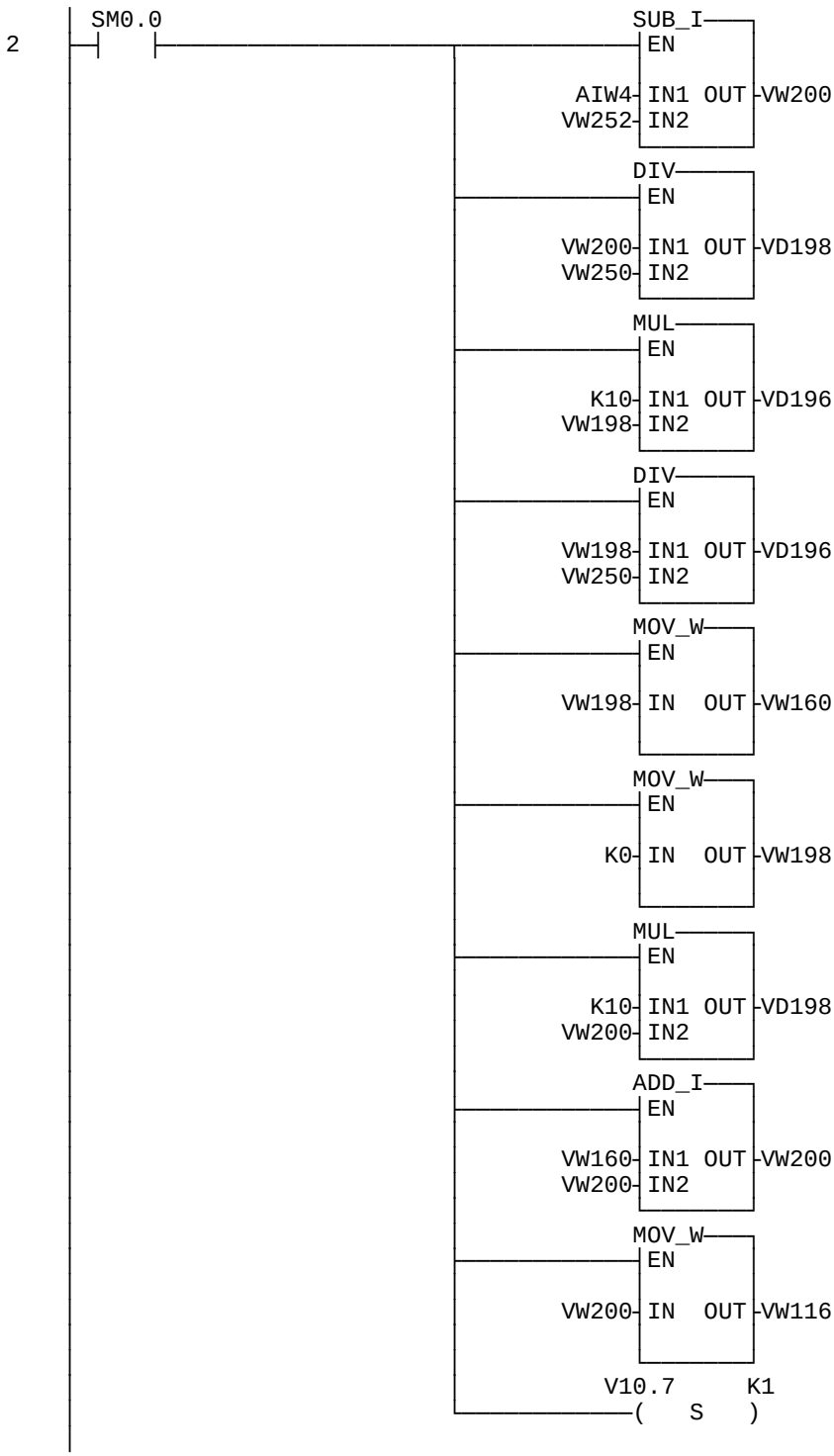
LAD (S7-MicroDOS)	STL (IEC)

Главная программа

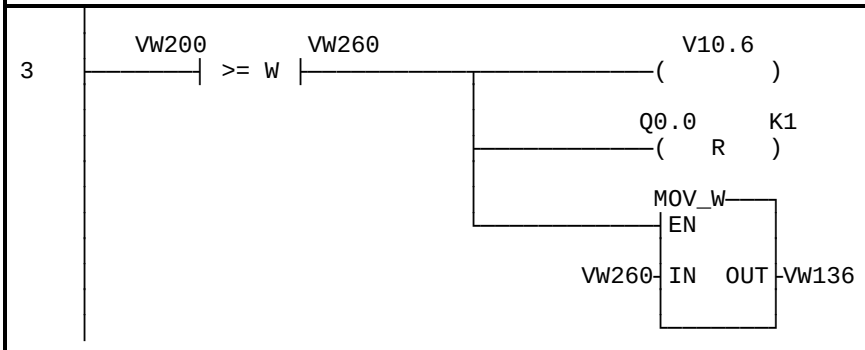
// Заголовок: Измерение температуры при помощи PT100 RTD и отображение его на TD200



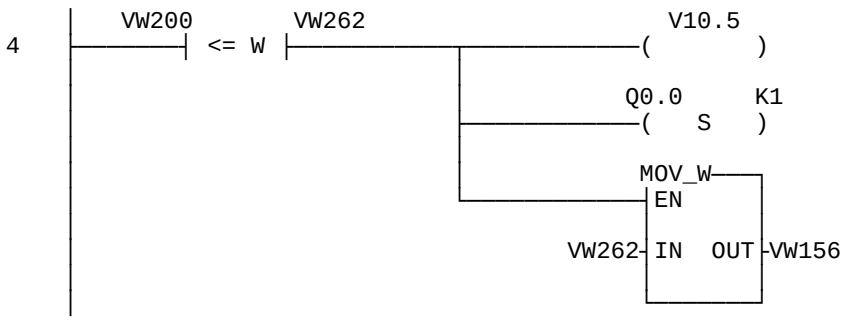

```
LD      SM0.1          // В первом цикле,
MOVD    0, VD196       // очистить VW196 и VW198
MOVW    16, VW250      // загрузить 1°C = 16 в VW250
MOVW    4000, VW252    // 0°C-Смещение = 4000
MOVW    300, VW260     // Верхний предел температуры  30°C
MOVW    200, VW262     // Нижний предел температуры 20°C
MOVW    20000, AQW0    // Управляющее значение для I0
```


2
SM0.0
SUB_I
EN
AIW4 IN1 OUT VW200
VW252 IN2
DIV
EN
VW200 IN1 OUT VD198
VW250 IN2
MUL
EN
K10 IN1 OUT VD196
VW198 IN2
DIV
EN
VW198 IN1 OUT VD196
VW250 IN2
MOV_W
EN
VW198 IN OUT VW160
MOV_W
EN
K0 IN OUT VW198
MUL
EN
K10 IN1 OUT VD198
VW200 IN2
ADD_I
EN
VW160 IN1 OUT VW200
VW200 IN2
MOV_W
EN
VW200 IN OUT VW116
V10.7 K1
(S)

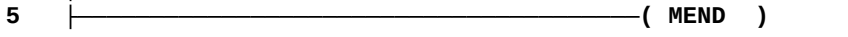
```
LD      SM0.0            // Всегда загружать измеренное
MOVW    AIW4, VW200      // значение в VW200
-I      VW252, VW200     // вычесть 0°C-Смещение
DIV     VW250, VD198     // разделить значение
MUL     10, VD196        // остатка x 10
DIV     VW250, VD196     // 10 x остаток / 30 = 1 цифра после запятой
MOVW    VW198, VW160     // сохранить одну цифру после запятой
MOVW    0, VW198         // стереть VW198
MUL     10, VD198        // Значение температуры x10
+I      VW160, VW200     // Значение температуры x10 + 1 цифра после запятой
MOVW    VW200, VW116     // передать результат в VW116 для отображения
S       V10.7, 1         // Выдать Сообщение 1
```




```
LDW>=   VW200, VW260     // если превышен верхний предел
=       V10.6            // Выдать Сообщение 2
R       Q0.0, 1          // Выключить печку
MOVW    VW260, VW136     // Верхний предел в VW136 для
                         // отображения Сообщения 2
```




```
LDW<=   VW200, VW262     // если меньше нижнего предела
=       V10.5            // Выдать Сообщение 3
S       Q0.0, 1          // Включить печку
MOVW    VW262, VW156     // Нижний предел в VW156 для
                         // отображения Сообщения 3
```




```
MEND                     // Конец программы
```

Блок данных DB1 (V Память):

```
// Блок параметров для TD200 для отображения сообщений
// со встроенным значением измеренной температуры и
// предупреждений со встроенными температурными пределами.

VB0     'TD'              // Идентификатор TD200
VB2     16#10             // Язык - Английский, обновлять так быстро, как возможно
VB3     16#00             // Дисплей с 20 знаками
VB4     3                 // Число сообщений = 3
VB5     0                 // Маркер MB0 для F - клавиш
VB6     0
VB7     100               // VB100 = Начало сообщения
VB8     0
VB9     10                // VB10 = Адрес битов разрешения сообщений

VB100   'Temperature='    // Текст1 = 12 знаков Сообщения 1
VB112   ' '               // Два пробела
VB114   16#00             // MSB байта формата - без обработки, без подтверждения
VB115   16#11             // LSB байта формата - 1 слово с 1 цифрой после запятой
VB116   16#00             // значение встроенного слова - MSB
VB117   16#00             // значение встроенного слова - LSB
VB118   16#DF             // Текст2 = 2 знака
VB119   'C'
VB120   'Temperature>'    // Text1 = 12 знаков Сообщения 2
VB132   ' '               // Два пробела
VB134   16#00             // MSB байта формата - без обработки, без подтверждения
VB135   16#11             // LSB байта формата - 1 слово с 1 цифрой после запятой
VB136   16#00             // значение встроенного слова - MSB
VB137   16#00             // значение встроенного слова - LSB
VB138   16#DF             // Text2 = 2 знака
VB139   'C'

VB140   'Temperature<'    // Text1 = 12 знаков Сообщения 3
VB152   ' '               // Два пробела
VB154   16#00             // MSB байта формата - без обработки, без подтверждения
VB155   16#11             // LSB байта формата - 1 слово с 1 цифрой после запятой
VB156   16#00             // значение встроенного слова - MSB
VB157   16#00             // значение встроенного слова - LSB
VB158   16#DF             // Текст2 = 2 знака
VB159   'C'
```

Copyright ©1996 by SIEMENS

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все права защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC

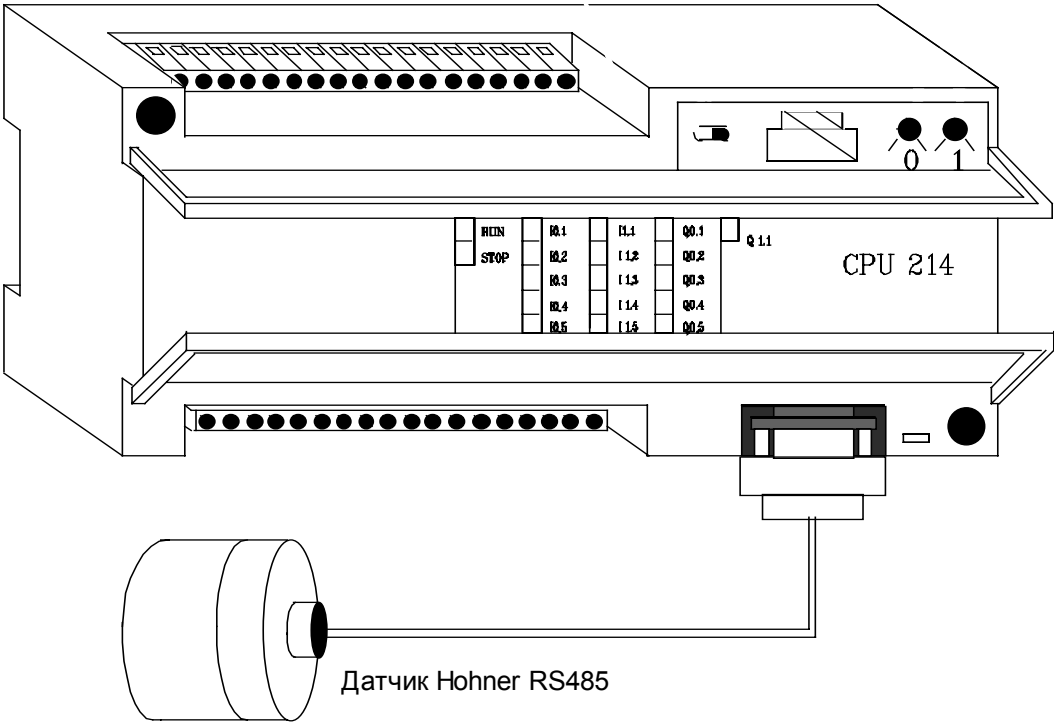
S7-200 Примеры

Группа	Тема
3	**Подключение датчика RS485 к PPI интерфейсу S7-200**

Краткое описание

Данный пример описывает как м.б. считано значение с датчика положения RS485 (производитель: Hohner Engineering) используя PPI интерфейс S7-200.

Схема



Описание аппаратуры

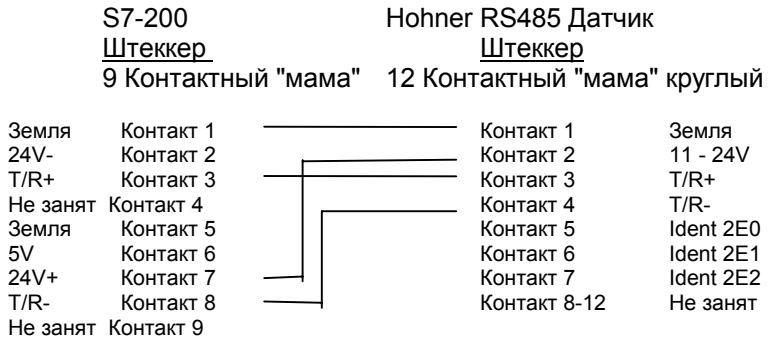
Существует два типа датчиков, имеющих разное разрешение:

1. Датчик с разрешением 13 бит: Тип SVE10-06213
2. Датчик с разрешением 25 бит: Тип SHE86-10214

Производитель: Hohner Electrical Engineering, D-78532 Tuttlingen/Germany

Теоретически возможно подключить 8 датчиков к интерфейсу RS485.

Распайка кабеля:

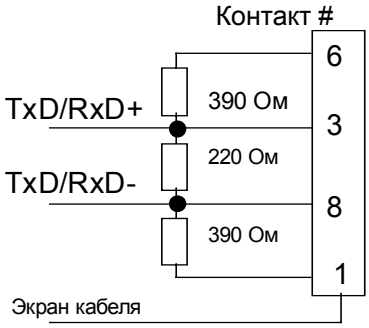


Проблемы с интерфейсом

При использовании датчика Hohner вместе с S7-200 могут возникнуть следующие проблемы:

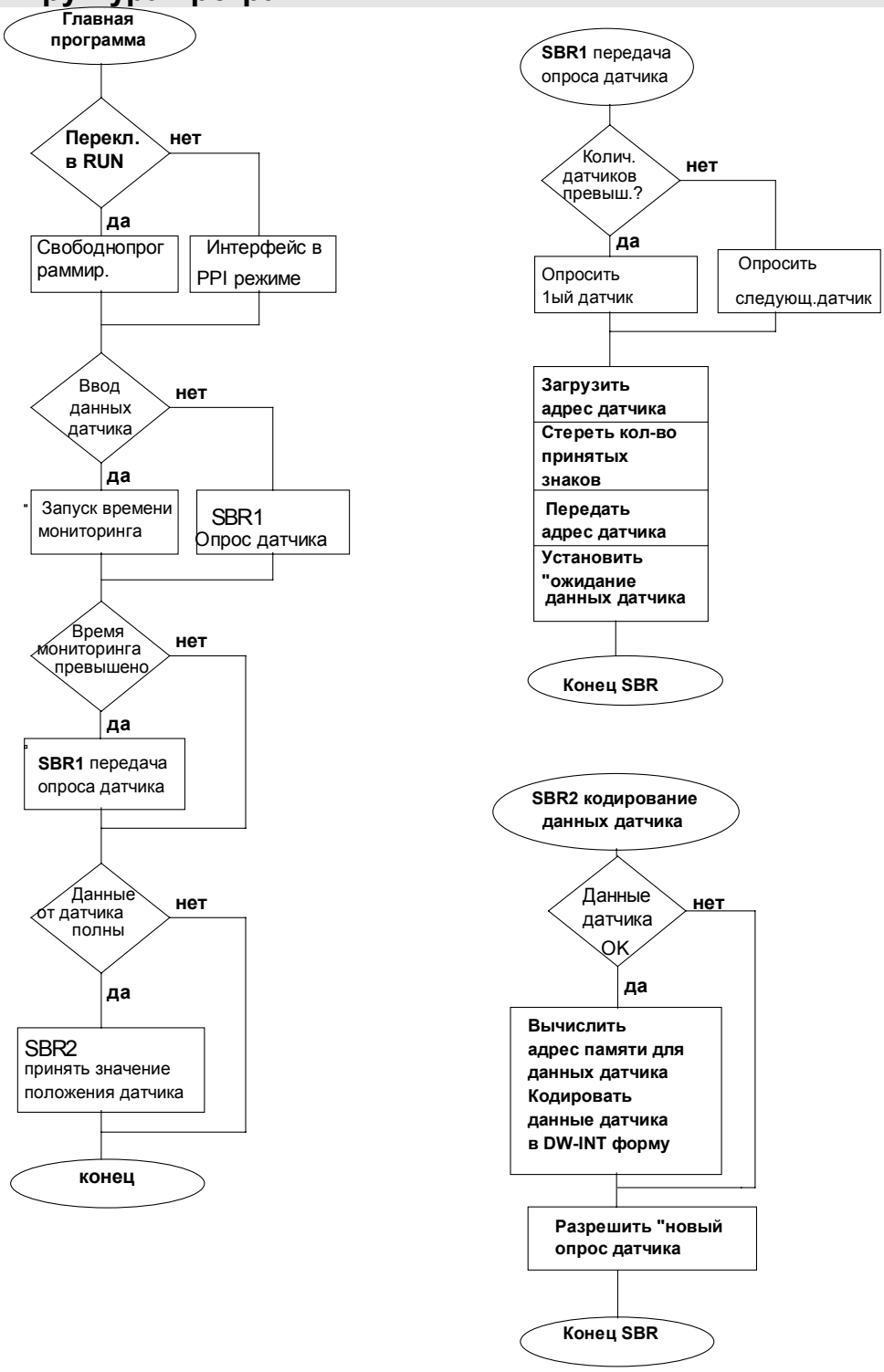
1. Ограничение по току для PPI интерфейса S7-200.
Допутимый ток не должен превышать 120mA, а для датчика Hohner необходимо 140mA. Поэтому датчик питается от отдельного 24V источника питания.
Возможно другое решение с использованием EMC-платы. Данная плата изолирует интерфейс RS485 от датчика. Напряжение питания при этом для интерфейса S7-200 не превышает 20%, или < 30mA.

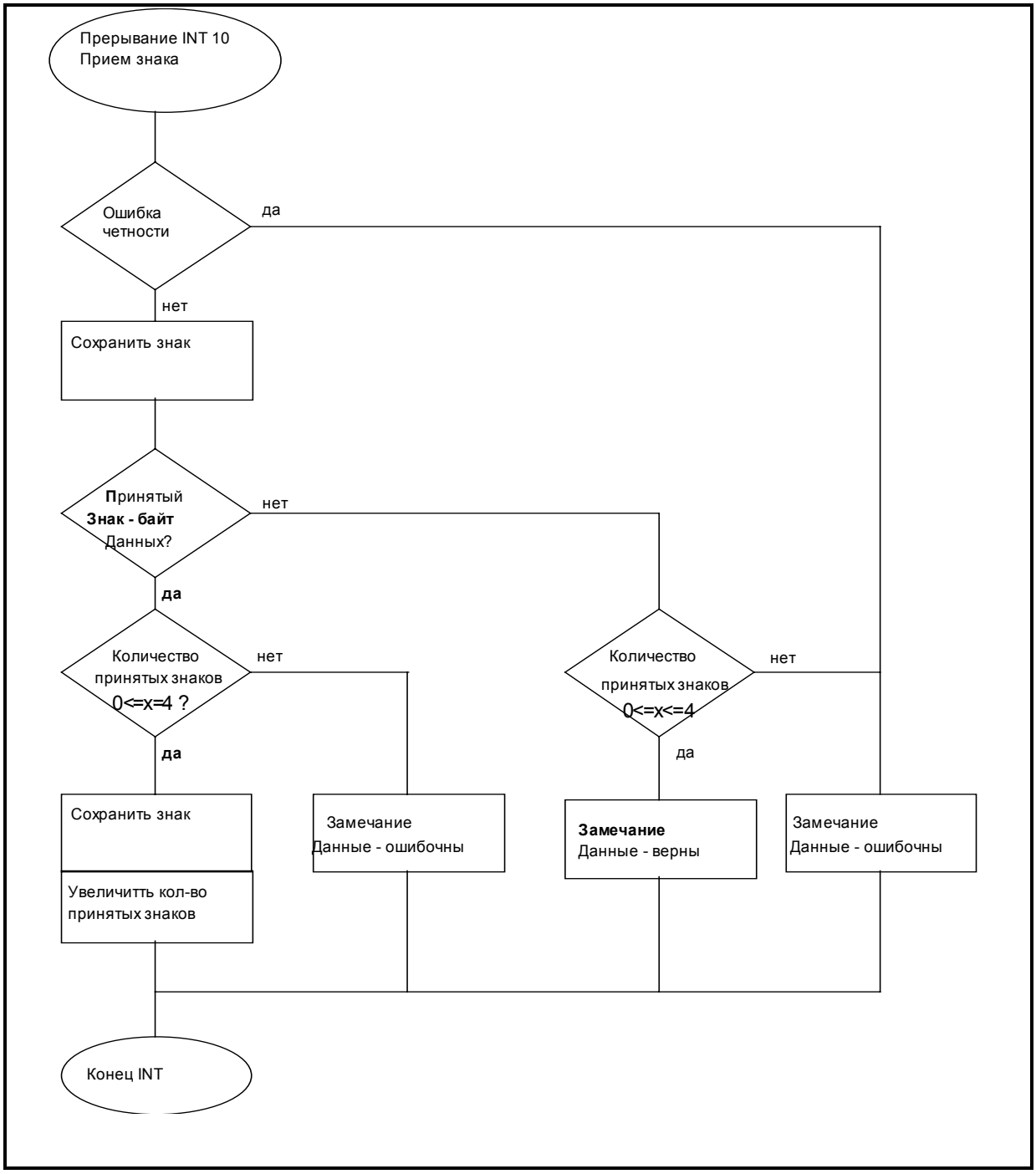
2. Помехи от датчика.
Уровень помех от датчика м.б. уменьшен при использовании терминатора:
6ES7-972-0B00-0XA0, или комбинации резисторов, как показано на рисунке ниже:



3. Слишком короткое время ожидание между двумя передаваемыми словами.
Время ожидания между двумя словами для датчика Hohner равно только 1 бит времени. Это м.б. слишком маленьким промежутком времени для CPU 212 или CPU 214 для того, чтобы сбросить блок UART. UART не будет готов для приема следующего знака после 2 бит времени и может возникнуть ошибка четности. Данная проблема м.б. решена путем увеличения времени ожидания датчика до 2 бит времени.

Структура программы




Прерывание INT 10
Прием знака
Ошибка четности
да
нет
Сохранить знак
Принятый Знак - байт Данных?
нет
да
Количество принятых знаков 0<=x=4 ?
нет
да
Количество принятых знаков 0<=x<=4
нет
да
Сохранить знак
Увеличиттть кол-во принятых знаков
Замечание Данные - ошибочны
Замечание Данные - верны
Замечание Данные - ошибочны
Конец INT

Описание программы

// TITLE = КОММУНИКАЦИЯ

// Инициализация:

Во время запуска - Подпрограмма 0 - устанавливаются параметры для PPI интерфейса, скорость передачи и кадр символа. Дополнительно устанавливается адрес датчика RS485.

Используемые данные:

Тип	Адрес	Описание
VD	200	Указатель на буфер передачи
VB	204	Количество байт данных для передачи (1)
VB	205	Передаваемый байт
VB	210	Количество принятых переданных байт
VB	211	Текущий принятый Байт 1
VB	212	Текущий принятый Байт 2
VB	213	Текущий принятый Байт 3
VB	214	Текущий принятый Байт 4
VB	215	Текущий Номер Датчика
VB	216	Максимальное количество датчиков
VB	220	Шинный Адрес Первого Датчика
VB	221	Чтение Значения в MSB
VB	222	Чтение Значения в
VB	223	Чтение Значения в
VB	224	Чтение Значения в LSB
VB	225	Шинный Адрес Второго Датчика
VB	226	Чтение Значения в MSB
VB	227	Чтение Значения в
VB	228	Чтение Значения в
VB	229	Чтение Значения в LSB
..		
VB	255	Шинный Адрес Восьмого Датчика
VB	256	Чтение Значения в MSB
VB	257	Чтение Значения в
VB	258	Чтение Значения в
VB	259	Чтение Значения в LSB
M	0.0	Чтение данных датчика активно
M	0.1	Данные верны
M	0.2	Принятые ошибочные данные <1 байта, >4 байт и Ошибка Четности
M	0.3	Вспомогательный Меркер <1 байта, >4 байт и Ошибка Четности
T	33	Время мониторинга; датчик не откликается 0.1 сек.
SM	0.0	Всегда ‘‘1’’
SM	0.1	Завершить перезапуск
SM	0.7	Переключатель в положении ‘‘RUN’’
SMB	2	Принятый байт последовательного интерфейс
SM	3.0	Ошибка Четности
SM	4.0	Переполнение очереди
SM	4.4	Разрешить прерывания
SM	4.5	Передатчик в ожидании
SMB	30	Контрольный Байт Интерфейса RS485

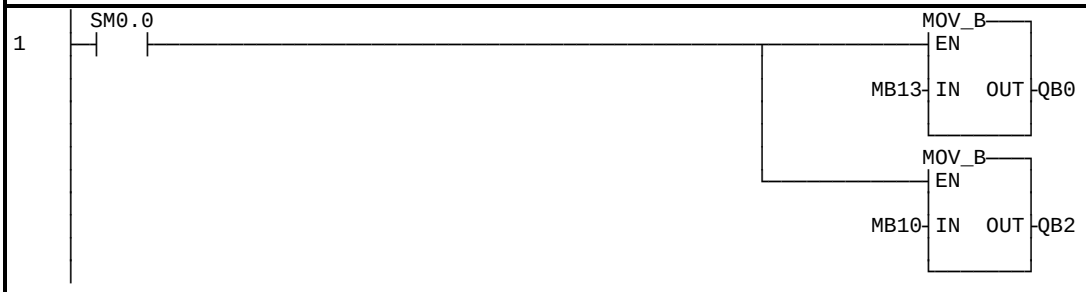
SM	SM	SM	Значение	Описание
30.7	30.6		00	Без контроля четности
			01	Контроль на четность
			10	Без контроля четности
			11	Контроль на нечетность
30.5			0	8 бит на знак
			1	7 бит на знак
30.4	30.3	30.2	000	38.4KB (214) / 19,2KB (212)
			001	19.2KB
			010	9,6 KB
			011	4.8 KB
			100	2,4 KB
			101	1.2 KB
			110	0.6 KB
			111	0.3 KB
30.1	30.0		00	PPI протокол
			01	Свободнопрограммируемый протокол
			10/11	PPI протокол

Тип	Прерывание	EVENT	Описание

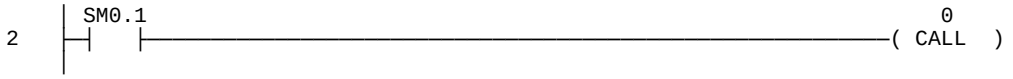
INT	--	9	Прерывание передачи данных
INT	10	8	Прием Данных

Тип	Адрес	Описание
SBR	0	Инициализация
SBR	1	Опрос передающего датчика
SBR	2	Сохранить данные датчика

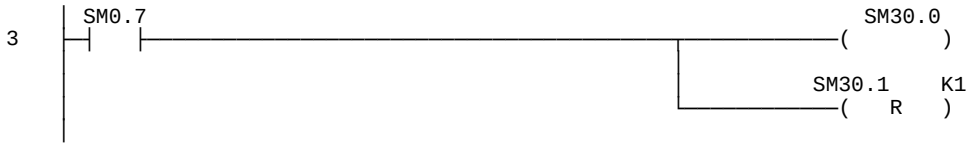
```
// Подключен датчик Hohner-RS485 с разрешением 13 и 25 бит
// данных на байт, 7 бит используются как биты данных.
// Выходы Q0.0 - Q1.1 установлены на 13 бит разрешения
//
// Данные проверяются на:
// - Сумму установленных битов, получаемых с контрольным байтом
// - Ошибка Четности
```



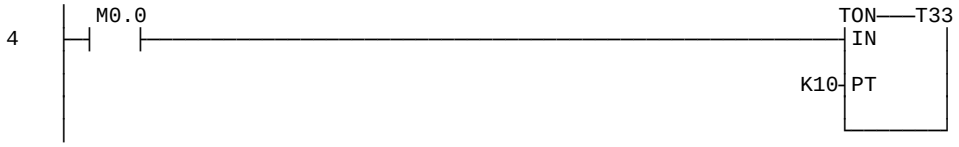

```
LD      SM0.0                   // Каждый цикл
MOVB    MB13    QB0             // Скопировать байт памяти13 в Бит выходов 0
MOVB    MB10    QB2             // Скопировать байт памяти10 в Бит выходов 2
```




```
// Инициализация коммуникации
LD      SM0.1                   // 1ый цикл программы
CALL    0                       // Вызов Подпрограммы инициализации
```




```
// Установить интерфейс PPI или свободнопрограммируемый интерфейс
LD      SM0.7                   //  Если RUN, установить свободнопрограммируемый
                                //  интерфейс
=       SM30.0                  //  Если TERM, установить PPI
R       SM30.1,1                //  00 = PPI ; 01 = свободнопрограммируемый
                                //  интерфейс
```




```
// Запуск таймера
LD      M0.0                    //Повторить опрос датчика, если он
TON     T33,10                  //не отвечает в течении 0.1 сек
```

```
5   |  M0.0    M0.1                                                  1
    |-| / |---| / |-------------------------------------------( CALL )
    |                |
    |  T33           |
    |-|   |----------|
    |
```

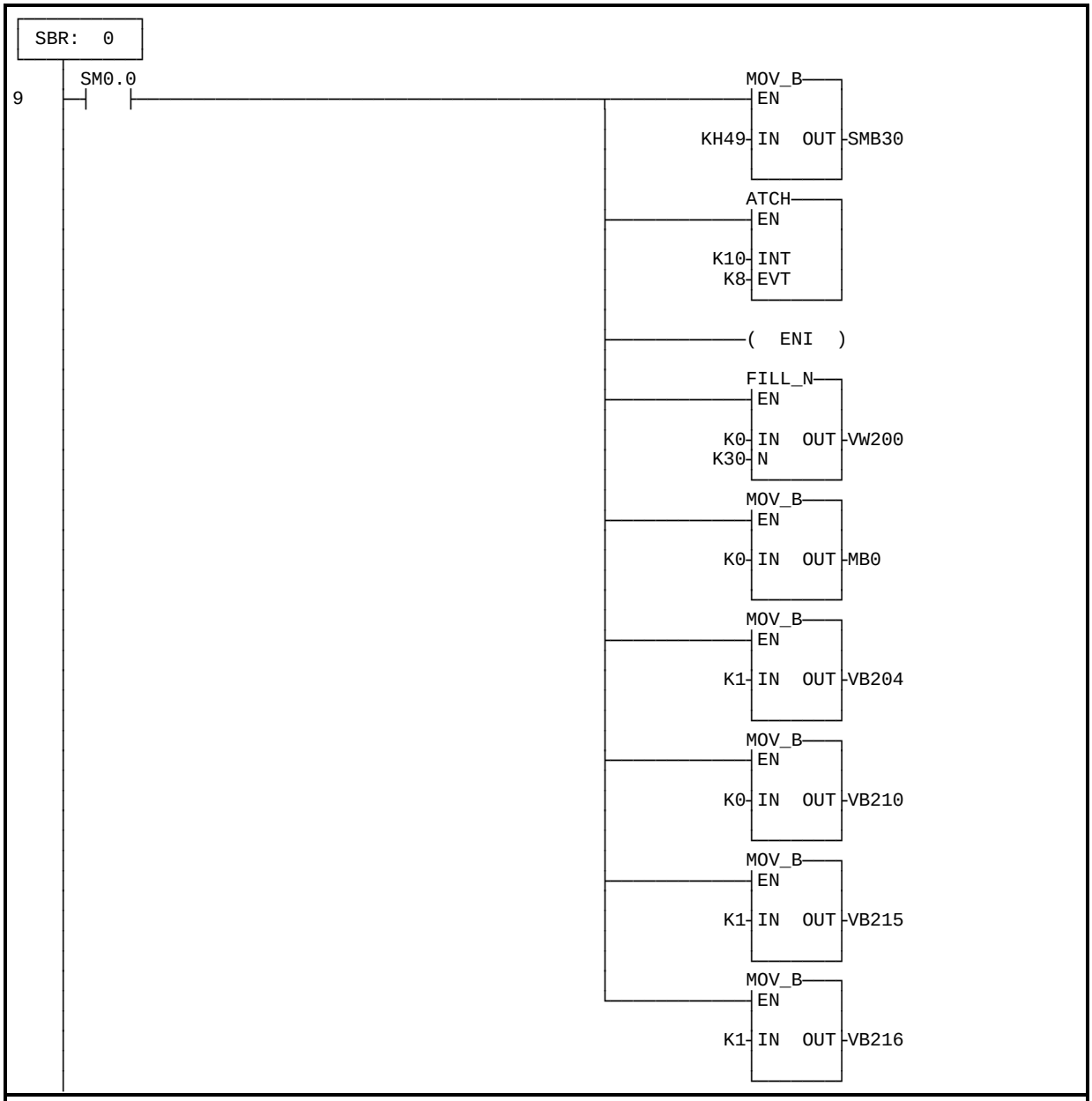
```
// Передача
LDN   M0.0                    // не M 0.0 - прочитанные данные датчика активны
O     T33                     // ИЛИ T33 активен
AN    M0.1                    // НЕ  M 0.1 - данные датчик верны
CALL  1                       // заполнить и передать буфер передачи
```

```
6   |  M0.1                                                   2
    |-|   |-------------------------------------------( CALL  )
    |
```

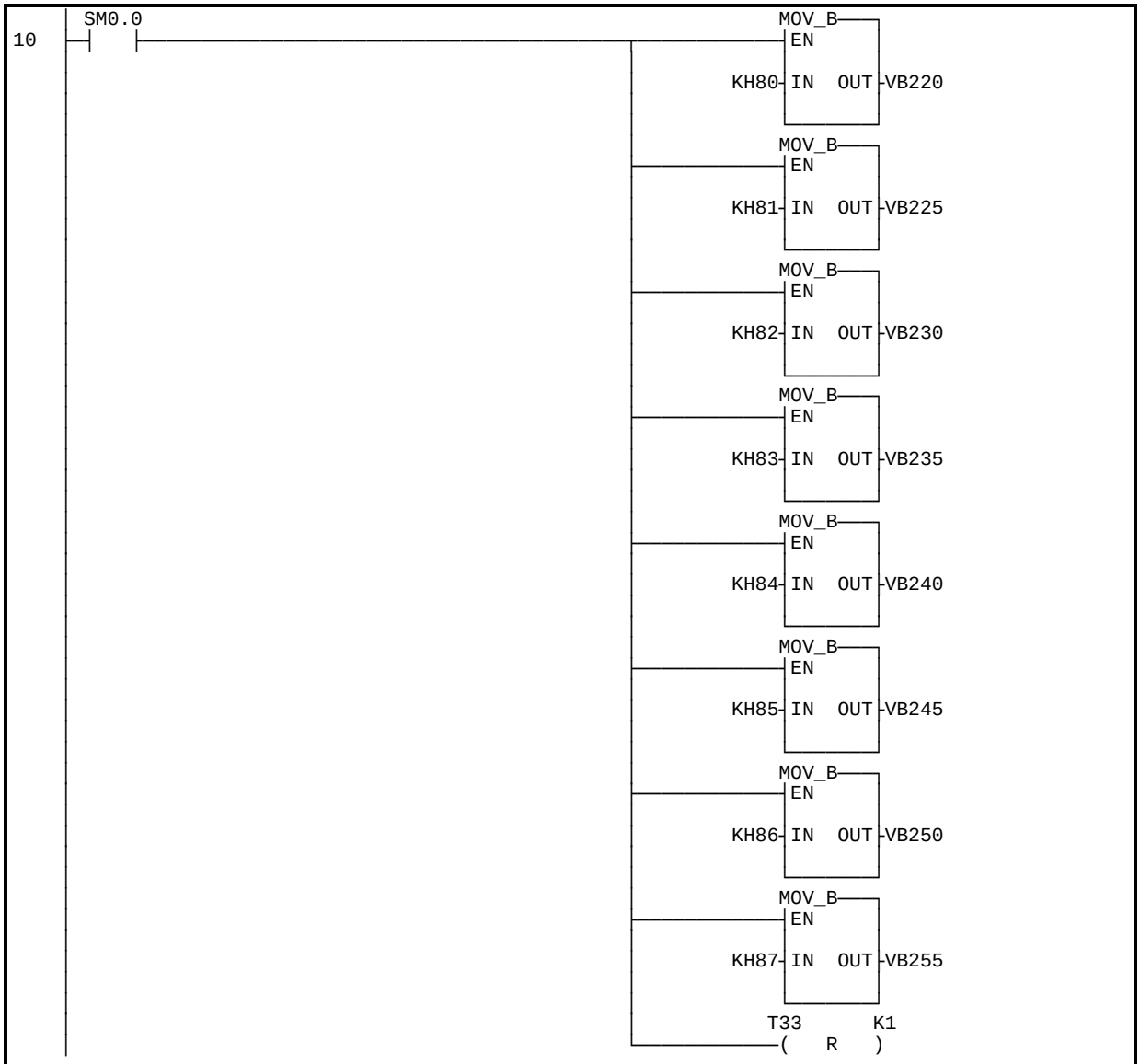
```
// Кодирование принятых днные
LD    M0.1                    // Данные датчика верны
CALL  2                       // Сохранить принятые данные
```

```
7   |-------------------------------------------------( MEND  )
    |
```

```
MEND                          // ***** Конец главной программы ******
```




```
// Инициализация передачи
SBR      0                    // ПОДПРОГРАММА 0
LD       SM0.0                // RLO=1
MOVB  16#49,SMB30             // 9600 Бод, с контролем четности, 8бит/знак
                              // Бит 7/6    01 - с контролем четности
                              // Бит 5      0 - 8 бит/знакг
                              // Бит 4/3/2 = 010 -  9.6 kB
                              // Бит 4/3/2 = 001 - 19.2 kB
                              // Бит 1/0 =  01 - свободнопрограммируемый интерфейс
ATCH  10,8                    // Знак приема INT 10 на EVENT 8
ENI                           // Разрешить прерывания
FILL  0,VW200,30              // Очистить VW200-VW258
MOVB  0,MB0                   // Очистить Меркер
MOVB  1,VB204                 // Количество переданных байт = 1
MOVB  0,VB210                 // Знак счетчика CC = 0
MOVB  1,VB215                 // Текущий Указатель Датчика = 1
MOVB  1,VB216                 // Номер Датчика      = 1
```




```
LD      SM0.0
                                  // Ввести Адрес для 1го Датчика
MOVB  16#80,VB220                 // 1ый Номер Датчика = 0
                                  // Ввести Адрес для 2го Датчика
MOVB  16#81,VB225                 // 2ой Номер Датчика = 1
                                  // Ввести Адрес для 3го Датчика
MOVB  16#82,VB230                 // 3ий Номер Датчика = 2
                                  // Ввести Адрес для 4го Датчика
MOVB  16#83,VB235                 // 4ый Номер Датчика = 3
                                  // Ввести Адрес для 5го Датчика
MOVB  16#84,VB240                 // 5ый Номер Датчика = 4
                                  // Ввести Адрес для 6го Датчика
MOVB  16#85,VB245                 // 6ой Номер Датчика = 5
                                  // Ввести Адрес для 7го Датчика
MOVB  16#86,VB250                 // 7ой Номер Датчика = 6
                                  // Ввести Адрес для 8го Датчика
MOVB  16#87,VB255                 // 8ой Номер Датчика= 7
R       T33,1                     // Сбросить таймер
```

```
11 |
   |-------------------------------------------------------------( RET )
   |
```

RET // Конец инициализации

```
 SBR:  1
   |
   | SM4.5
13 |-| / |-------------------------------------------------------( CRET )
   |
```

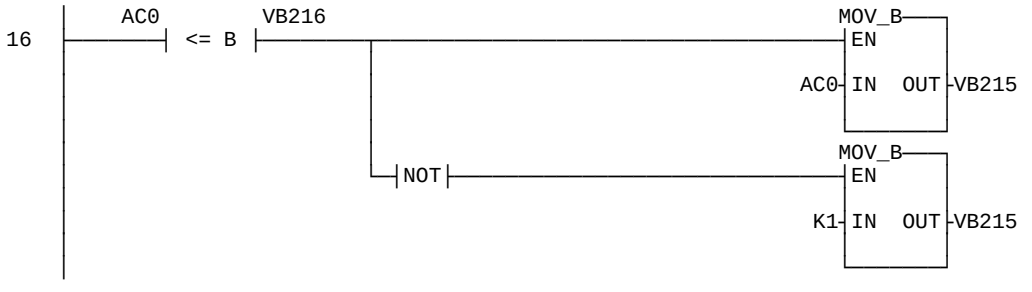
// Передача опроса Датчика
SBR 1 // *** ПОДПРОГРАММА 1 ***
LDN SM4.5 // Обрабатывается передатчик
CRET // Конец

```
   | SM0.0                                                  MOV_DW
14 |-| |-------------------------------------------------------| EN
   |                                                        |
   |                                                        | K0- IN  OUT -AC0
   |                                                        |
   |                                                        |
   |                                                        | MOV_DW
   |                                                        | EN
   |                                                        |
   |                                                        | K0- IN  OUT -AC1
   |                                                        |
   |                                                        |
```

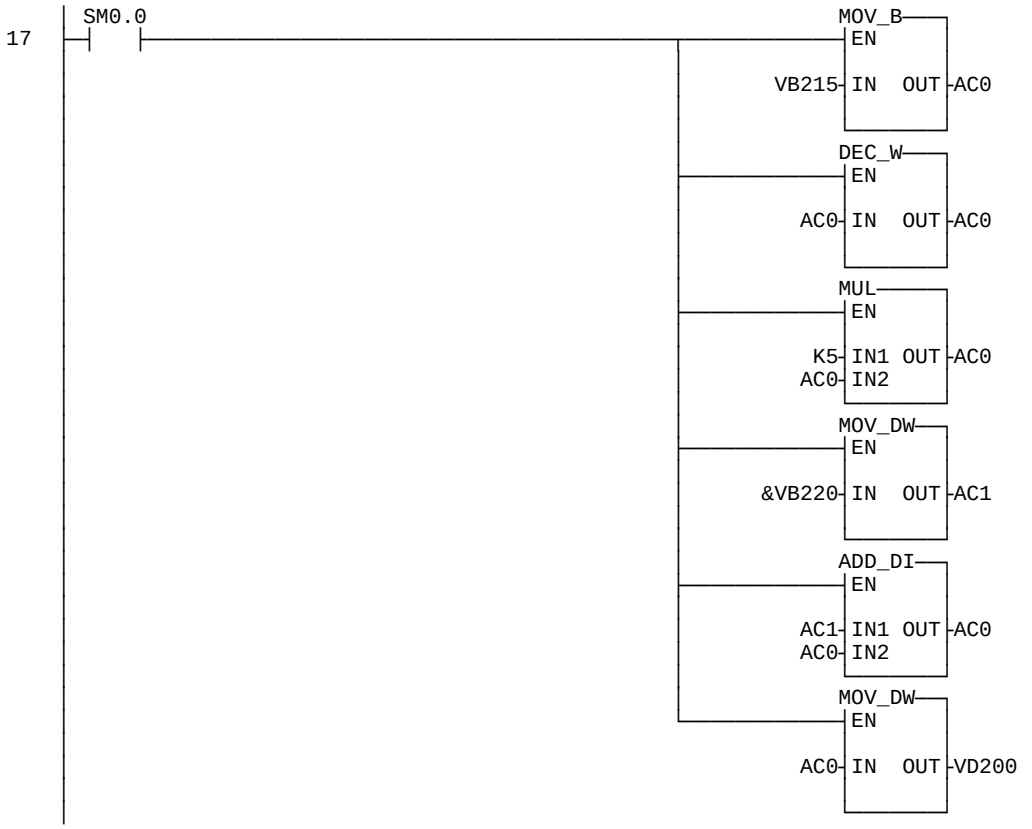
LD SM0.0 // RLO=1
MOVD 0,AC0 // Очистить ACCU0
MOVD 0,AC1 // Очистить ACCU1

```
   | SM0.0                                                  MOV_B
15 |-| |-------------------------------------------------------| EN
   |                                                        |
   |                                                        | VB215- IN  OUT -AC0
   |                                                        |
   |                                                        |
   |                                                        | INC_W
   |                                                        | EN
   |                                                        |
   |                                                        | AC0- IN  OUT -AC0
   |                                                        |
   |                                                        |
```

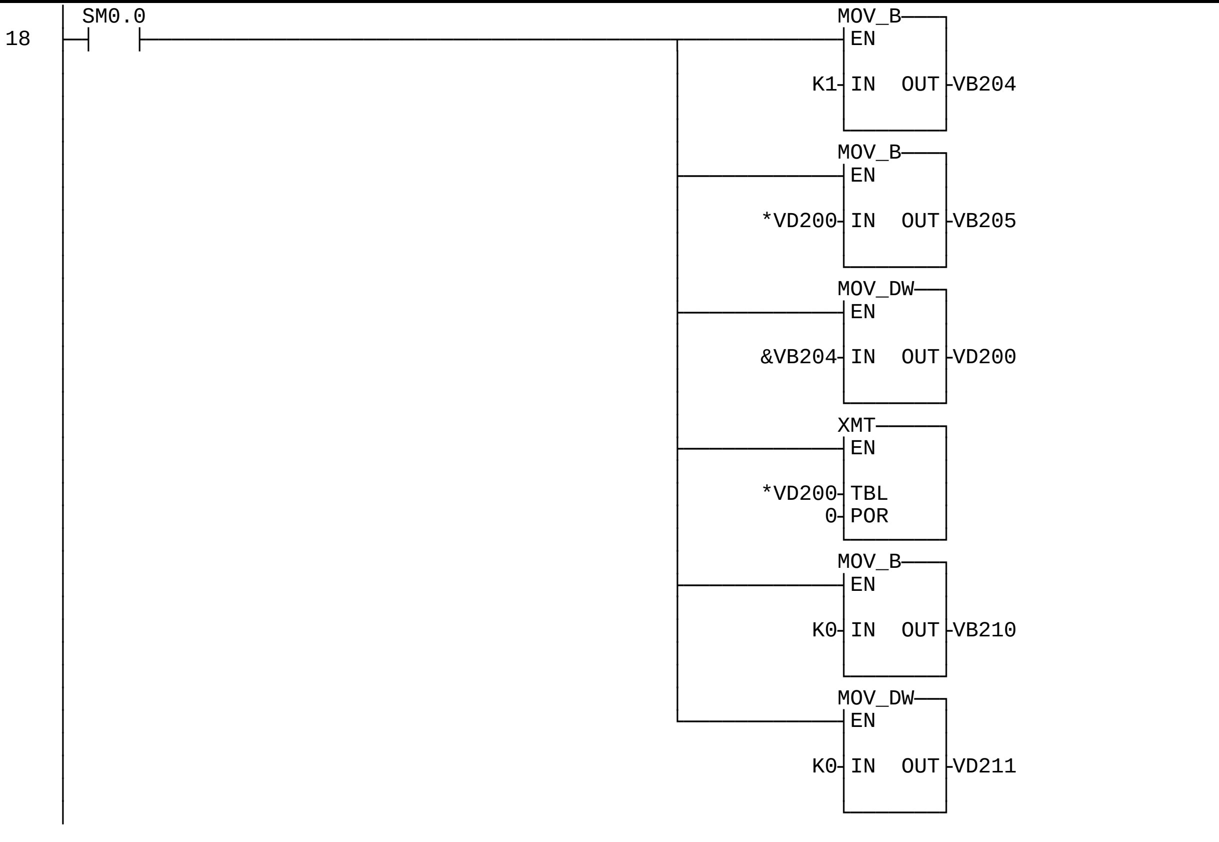
LD SM0.0 // RLO=1
MOVB VB215,AC0 // Увеличить указатель на адрес датчика
INCW AC0



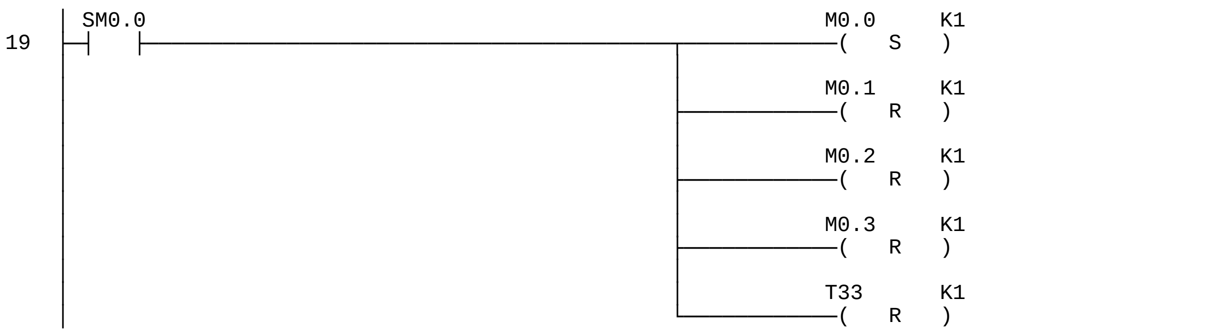

```
LDB<= AC0,VB216          // Номер Датчика превышент ?
MOVB  AC0,VB215                  // Сохранить новый указатель на адрес датчика
NOT
MOVB  1,VB215            // иначе, установить указатель на адрес датчика
                         // на нижнее граничное значение
```




```
// Вычислить указатель на адрес датчика
LD      SM0.0                    // RLO=1
MOVB  VB215,AC0                          // ((тек. № датчика - 1) * 5 ) + базовый адрес (VB220)
DECW  AC0                        //
MUL     5,AC0
MOVD  &VB220,AC1
+D      AC1,AC0
MOVD  AC0,VD200
```




```
// Загрузить переданные данные (адрес датчика)
LD      SM0.0                 // RLO=1
MOVB  1,VB204            // Количество переданных байт = 1
MOVB  *VD200,VB205             // Загрузить переданные данные
MOVD &VB204,VD200              // Установить указатель передачи на буфер передачи
XMT    *VD200,0          // Передача
MOVB   0,VB210           // Сбросить счетчик символов
MOVD   0,VD211           // Очистить буфер приема
```




```
LD      SM0.0        // RLO=1
S       M0.0,1       // Установить "чтение данных датчика активно"
R       M0.1,1       // Сбросить " данные датчика верны"
R       M0.2,1       // Сбросить " данные датчика ошибочны"
R       M0.3,1       // Сбросить "прием OK"
R       T33,1        // Сбросить таймер
```

```
20 ├──────────────────────────────────────────────────────( RET )
```

RET **// Конец ПОДПРОГРАММЫ 1**

```
SBR: 2
    SM0.0                                                  MOV_DW
22 ├─┤   ├──────────────────────────────────────────────┬──┤EN
                                                        K0-IN  OUT-AC0
                                                        └──────────┘
                                                        MOV_DW
                                                        ├──┤EN
                                                        K0-IN  OUT-AC1
                                                        └──────────┘
```

// Сохранить данные датчика

// Первым приходит Младший Байт, затем Старший Байт
// В каждом байте используются Биты 0-6

//V211.0....V214.3 = 25 Биты Датчик

```
SBR    2                    // ПОДПРОГРАММА 2

LD     SM0.0                // RLO = 1
MOVD   0,AC0                // Очистить ACCU 0
MOVD   0,AC1                // Очистить ACCU 1
```

```
    M 0.2                                                  MOV_B
23 ├─┤ / ├──────────────────────────────────────────────┬──┤EN
                                                        VB215-IN  OUT-AC0
                                                        └──────────┘
                                                        DEC_W
                                                        ├──┤EN
                                                        AC0-IN  OUT-AC0
                                                        └──────────┘
                                                        MUL
                                                        ├──┤EN
                                                        K5-IN1 OUT-AC0
                                                        AC0-IN2
                                                        MOV_DW
                                                        ├──┤EN
                                                        &VB221-IN  OUT-AC1
                                                        └──────────┘
                                                        ADD_DI
                                                        ├──┤EN
                                                        AC1-IN1 OUT-AC0
                                                        AC0-IN2
                                                        MOV_DW
                                                        ├──┤EN
                                                        AC0-IN  OUT-VD200
                                                        └──────────┘
```

```
LDN    M0.2                    // Данные датчика верны?
                               // Подсчитать адрес памяти
MOVB  VB215,AC0                // ((тек. № датчика - 1) * 5 ) + Базовый Адрес (VB221)
DECW  AC0
MUL    5,AC0
MOVD  &VB221,AC1
+D       AC1,AC0
MOVD  AC0,VD200
```

```
    | SM0.0                                                    MOV_DW
24  |-|   |-------------------------------------------------------|EN
    |                                                           |
    |                                                       K0-|IN   OUT|-MD10
    |                                                           |
```

```
LD     SM0.0                 // RLO=1
MOVD  0,MD10     // Очистить Вспомогательный Меркер
```

```
    | V211.0                                                    M13.0
25  |-|   |-------------------------------------------------------(       )
    |
```

```
LD   V211.0                    // "Скомпоновать" данные датчика
=    M13.0
```

```
    | V211.1                                                    M13.1
26  |-|   |-------------------------------------------------------(       )
    |
```

```
LD   V211.1
=    M13.1
```

```
    | V211.2                                                    M13.2
27  |-|   |-------------------------------------------------------(       )
    |
```

```
LD   V211.2
=    M13.2
```

```
    | V211.3                                                    M13.3
28  |-|   |-------------------------------------------------------(       )
    |
```

```
LD   V211.3
=    M13.3
```

```
    | V211.4                                                    M13.4
29  |-|   |-------------------------------------------------------(       )
    |
```

```
LD   V211.4
=    M13.4
```

```
30  | V211.5                                                  M13.5
    |-| |-------------------------------------------------------( )
```

LD V211.5
= M13.5

```
31  | V211.6                                                  M13.6
    |-| |-------------------------------------------------------( )
```

LD V211.6
= M13.6

```
32  | V212.0                                                  M13.7
    |-| |-------------------------------------------------------( )
```

LD V212.0
= M13.7

```
33  | V212.1                                                  M12.0
    |-| |-------------------------------------------------------( )
```

LD V212.1
= M12.0

```
34  | V212.2                                                  M12.1
    |-| |-------------------------------------------------------( )
```

LD V212.2
= M12.1

```
35  | V212.3                                                  M12.2
    |-| |-------------------------------------------------------( )
```

LD V212.3
= M12.2

```
36  | V212.4                                                  M12.3
    |-| |-------------------------------------------------------( )
```

LD V212.4
= M12.3

```
37  | V212.5                                                  M12.4
    |-| |-------------------------------------------------------( )
```

LD V212.5
= M12.4

```
    │
    │ V212.6                                                  M12.5
38  ├─┤   ├──────────────────────────────────────────────────(       )
    │
```

LD V212.6
= M12.5

```
    │ V213.0                                                  M12.6
39  ├─┤   ├──────────────────────────────────────────────────(       )
    │
```

LD V213.0
= M12.6

```
    │ V213.1                                                  M12.7
40  ├─┤   ├──────────────────────────────────────────────────(       )
    │
```

LD V213.1
= M12.7

```
    │ V213.2                                                  M11.0
41  ├─┤   ├──────────────────────────────────────────────────(       )
    │
```

LD V213.2
= M11.0

```
    │ V213.3                                                  M11.1
42  ├─┤   ├──────────────────────────────────────────────────(       )
    │
```

LD V213.3
= M11.1

```
    │ V213.4                                                  M11.2
43  ├─┤   ├──────────────────────────────────────────────────(       )
    │
```

LD V213.4
= M11.2

```
    │ V213.5                                                  M11.3
44  ├─┤   ├──────────────────────────────────────────────────(       )
    │
```

LD V213.5

= M11.3

```
45  | V213.6                                                  M11.4
    |-| |-------------------------------------------------------( )
    |
```

LD V213.6
= M11.4

```
46  | V214.0                                                  M11.5
    |-| |-------------------------------------------------------( )
    |
```

LD V214.0
= M11.5

```
47  | V214.1                                                  M11.6
    |-| |-------------------------------------------------------( )
    |
```

LD V214.1
= M11.6

```
48  | V214.2                                                  M11.7
    |-| |-------------------------------------------------------( )
    |
```

LD V214.2
= M11.7

```
49  | V214.3                                                  M10.0
    |-| |-------------------------------------------------------( )
    |
```

LD V214.3
= M10.0

```
50  | M0.2                                                  MOV_DW
    |-| / |-------------------------------------------------|EN
    |                                                       |
    |                                                 MD10-|IN  OUT|-*VD200
    |                                                       |
    |                                                       |
```

LDN M0.2 **// Данные Датчика верны ?**
MOVD MD10,*VD200 **// Сохранить данные датчика**

```
51  | SM0.0                                                 M0.0     K1
    |-| |-------------------------------------------------------( R )
    |                                                 |
    |                                                 |
    |                                                 |
    |                                                 |
    |                                                 |
```

// Разрешить новый опрос датчика

```
LD    SM0.0                    // RLO=1
R     M0.0,1                   // Установить "чтение данных датчика активно"
R     M0.1,1                   // Сбросить " данные датчика верны"
```

```
52  |-------------------------------------------------------------( RET )
    |
```

```
RET                            // Конец ПОДПРОГРАММЫ 2
```

```
 INT: 10
    |
    | SM4.5
54  |-| / |-------------------------------------------------------( CRETI )
    |     |
    |  M0.0
    |-| / |-
    |
```

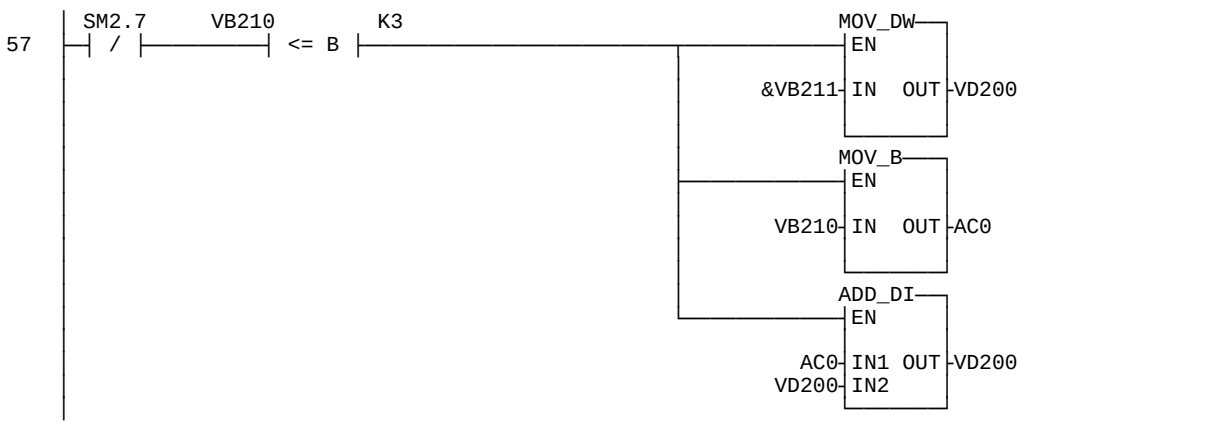
```
// Программа прерывания для приема
INT  10                        // *** INT 10 ***
LDN     SM4.5                  // Обрабатывается Передача
ON      M0.0                   // данные ожидаются ?
CRETI                          // Конец
```

```
    | SM0.0                                                  T33      K1
55  |-|   |-------------------------------------------------------( R )
    |                                                 |
    |                                                 |      MOV_DW
    |                                                 |-----------| EN
    |                                                            |
    |                                                        K0- IN  OUT -AC0
    |                                                            |
    |                                                            |
```

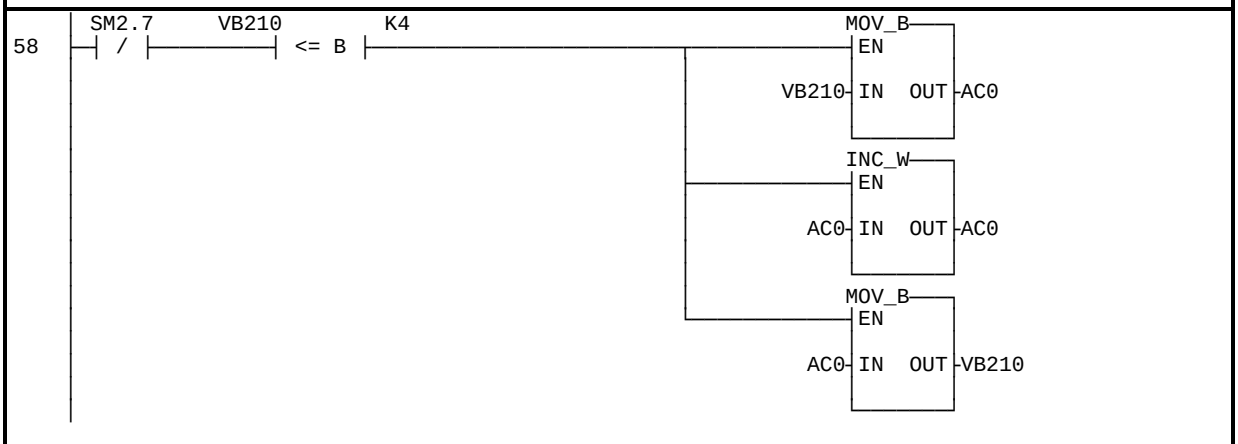
```
LD      SM0.0                  // RLO = 1
R       T33,1                  // Сбросить таймер
MOVD  0,AC0                    // Очистить ACCU
```

```
    | SM3.0                                                  MOV_B
56  |-|   |-------------------------------------------------------| EN
    |                                                            |
    |                                                        K5- IN  OUT -VB210
    |                                                            |
    |                                                            |
```

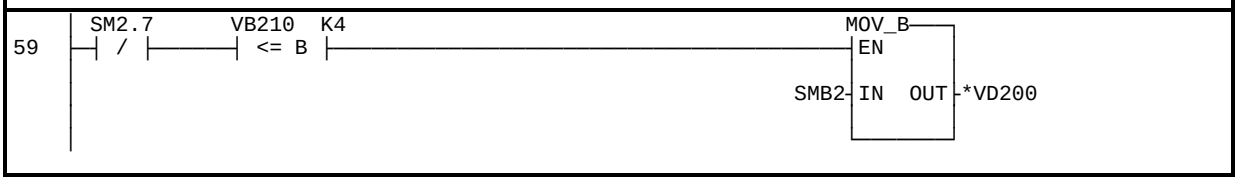
```
LD      SM3.0                  // Ошибка Четности
MOVB  5,VB210                  // Установить счетчик символов
                               // в недействительное значение
```




```
LDN     SM2.7                   // СЛОВО ДАННЫХ датчика
AB<=    VB210,3                 // Счетчик Знаков <= 3
MOVD    &VB211,VD200            // Загрузить адрес 1го знак буфера
MOVB    VB210,AC0               // Счетчик Знаков в ACCU
+D      AC0,VD200               // Подсчитать указатель
```




```
LDN     SM2.7                   // СЛОВОДАННЫХ от Датчика
AB<=    VB210,4                 // Счетчик Знаков <= 4 ?
MOVB    VB210,AC0               // Увеличить Счетчик Знаков
INCW    AC0
MOVB    AC0,VB200
```




```
LDB<=   VB210,4                 // Больше, чем 4 знака
AN      SM2.7                   // Слово Данных от датчика
MOVB    AC0,VB210               // Загрузить символ в буфер приема
```

```
    |   VB210        K1         VB210        K4                    M0.3
60  |--------| >= B |-------------------| <= B |-------------------------(        )
    |
```

```
LDB>= VB210,1                    // Проверка  >= 1 байт достигнуто
AB<=  VB210,4                    // Проверка <= 4 байт достигнуто
=     M0.3                       // Прием OK
```

```
    | SM2.7    M0.3                                                  M0.0     K1
61  |--|   |----|   |-------------------------------------------------(   R   )
    |                                                  |
    |                                                  |             M0.1     K1
    |                                                  |-------------(   S   )
    |                                                  |
    |                                                  |             M0.2     K1
    |                                                  |-------------(   R   )
    |
```

```
LD    SM2.7                      // Если контрольное значение (всех прочитаных данных)
A     M0.3                       // >=1 и <=4 байт данных присутствует,
R     M0.0,1                     // сбросить чтение данных активно
S     M0.1,1                     // Установить Данные верны
R     M0.2,1                     // Сбросить Ошибка Приема Данных
```

```
    | SM2.7    M0.3                                                  M0.0     K1
62  |--|   |----| / |-------------------------------------------------(   R   )
    |                                                  |
    |                                                  |             M0.1     K1
    |                                                  |-------------(   S   )
    |                                                  |
    |                                                  |             M0.2     K1
    |                                                  |-------------(   S   )
    |
```

```
LD    SM2.7                      // Если контрольное значение (всех прочитаных данных)
AN    M0.3                       // >=1 и <=4 байт данных НЕ присутствует,
R     M0.0,1                     // сбросить чтение данных активно
S     M0.1,1                     // Установить Данные верны
S     M0.2,1                     // Сбросить Ошибка Приема Данных
```

```
    |
63  |-------------------------------------------------------------------( RETI  )
    |
```

RETI

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все права защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC
S7-200 Примеры

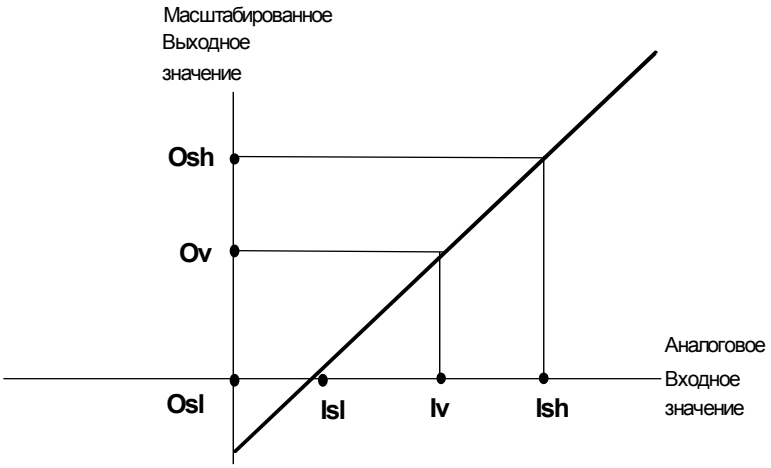
Группа	Тема
6	**Масштабирование аналоговых значений**

Краткое описание

Данная программа считывает аналоговое значение с входа одного из аналогового входного канала S7-200 и выдает выходное значение XQ в область масштабированную пользователем. Все необходимые параметры получают значения в сегменте инициализации программы. В формуле масштабирования используются следующие пременные:

Ov	=	масштабированное выходное значение
Iv	=	аналоговое входное значение
Osh	=	верхний предел шкалы масштабированного выходного значения
Osl	=	нижний предел шкалы масштабированного выходного значения
Ish	=	верхний предел шкалы аналогового входного значения
Isl	=	нижний предел шкалы аналогового входного значения

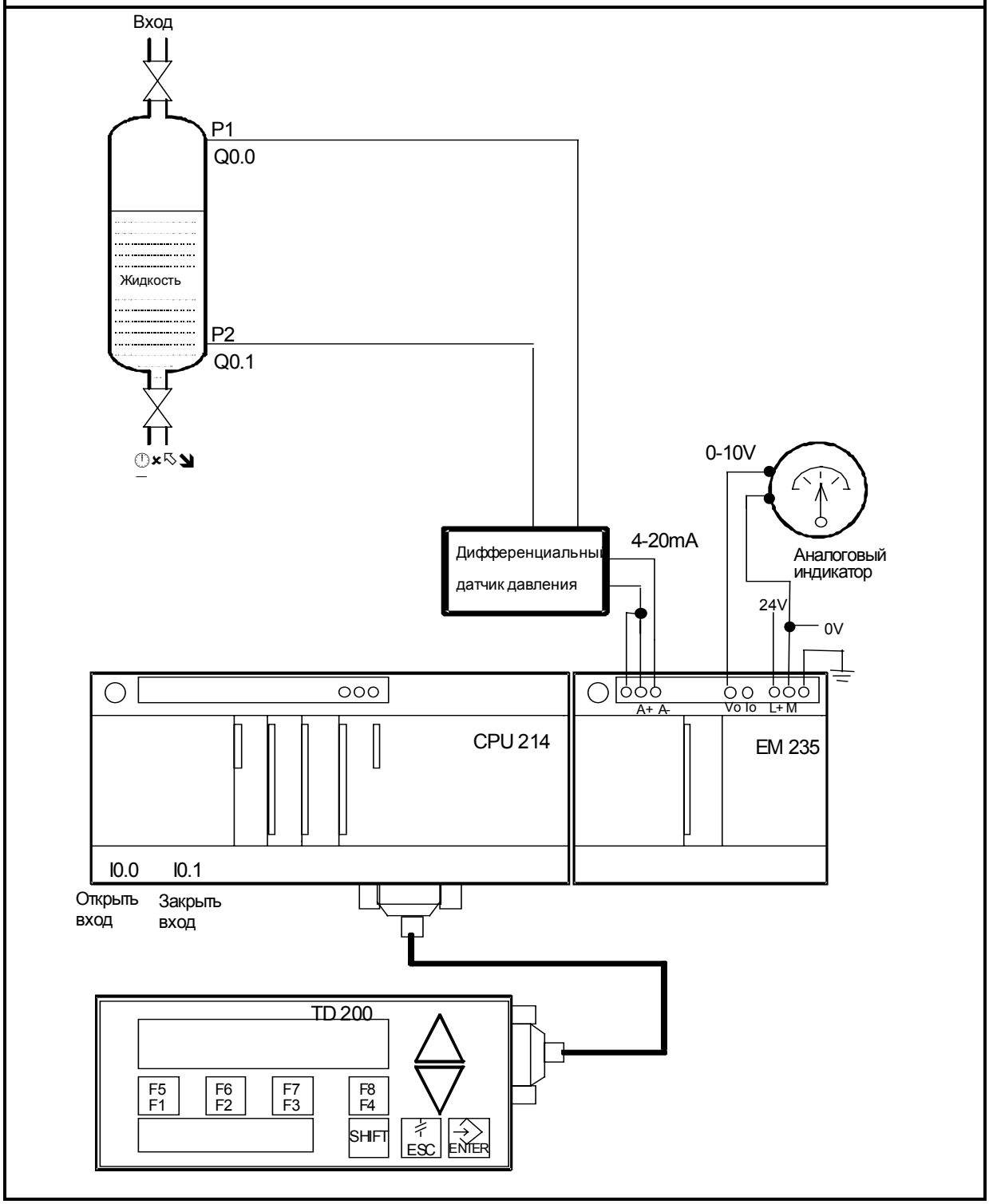
Схема масштабирования м.б. представлена следующей диаграммой:



Из диаграммы м.б. выведена следующая формула для расчета масштабированного значения:

Ov = [(Osh - Osl) * (Iv - Isl) / (Ish - Isl)] + Osl

Блок-схема

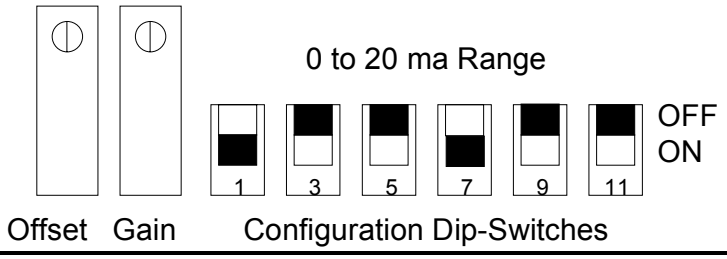

Вход
P1
Q0.0
Жидкость
P2
Q0.1
0-10V
Дифференциальный
датчик давления
4-20mA
Аналоговый
индикатор
24V
0V
A+ A-
Vo Io
L+ M
CPU 214
EM 235
I0.0
I0.1
Открыть
вход
Закрыть
вход
TD 200
F5
F1
F6
F2
F7
F3
F8
F4
SHIFT
ESC
ENTER

Структура программы

Аппаратные требования

1. Задание входного диапазона аналогого модуля EM235

6 DIP переключателей в нижней части модуля EM235 служат для задания рабочего диапазона входа по напряжению или току. Потенциометры ‘’СМЕЩЕНИЕ’’ и ‘’УСИЛЕНИЕ’’ для калибрирования аналогого входа модуля.



Описание программы

Закрытый бункер содержит жидкость. Текущий уровень жидкости должен все время отображаться на индикаторе. Программа следит за уровнем жидкости и заданными пределами, и если достигнут максимальный или минимальный предел, то это сигнализируется на TD200, и входной или выходной вентиль закрывается.

- Дифференциальный датчик давления выдает ток (4 - 20mA) пропорционально уровню жидкости на вход модуля аналоговых входов.
- Аналоговый модуль EM235 откалибрирован так, что при уровне 10м, аналоговое значение тока = 20mA преобразуется в цифровое значение 32000; а при 0м (4mA) в значение 3200. Программа масштабирует преобразованное цифровое значение в высоту (в метрах).
- Программа пользователя проверяет не выходит ли принятое значение за максимальный предел 10м. Если данный предел достигается, она выдает на TD200 сообщение "Макс. Предел Достигнут". Одновременно устанавливается бит выходов для закрытия входного вентиля.
- Уровень жидкости не должен опускаться ниже предела 1м. Если данный предел достигается, она выдает на TD200 сообщение "Мин. Предел Достигнут". и активизирует выход для закрытия выходного вентиля.
- Напряжение для индикатора уровня жидкости генерируется на модуле аналоговых выходов. Данное напряжение генерируется путем записи соответствующего цифрового значения в AQW, или слово аналогового выхода.
- Модуль аналогого выхода выдает уровень жидкости (между 0м и 10м) на измерительный прибор в виде напряжения в диапазоне 0 -10V. Аналоговый дисплей измерительного прибора реагирует на напряжение отклонением указателя пропорционально уровню жидкости.

Формула масштабирует любое заданное значение между любыми заданными максимальными и минимальными точками шкалы. Данная программа, в частности, принимает слово аналогового входа (AIW) и масштабирует его для TD200 и модуля аналоговых выходов. В начале, программа считывает AIW, имеющее значение от 4 до 20 mA и масштабирует его в значение между 3200 и 32000. Данное значение затем масштабирует для AQW, которое имеет диапазон от 0 до 32000. После этого, AIW масштабирует для TD200, который имеет минимальное значение шкалы = 0 и максимальное значение шкалы = 10.

Copyright ©1996 by SIEMENS

Листинг программы

// ЗАГОЛОВОК = Мониторинг уровня жидкости с масштабированными аналоговыми значениями

```
     SM0.0                                                    MOV_DW
1   ─┤   ├──────────────────────────────────────────────────┬─┤EN
                                                             │
                                                         K0-┤IN   OUT├-AC1
                                                             │
                                                             MOV_W
                                                             ├─┤EN
                                                             │
                                                       AIW0-┤IN   OUT├-AC1
                                                             │
                                                             DI_REAL
                                                             ├─┤EN
                                                             │
                                                        AC1-┤IN   OUT├- VD500
```

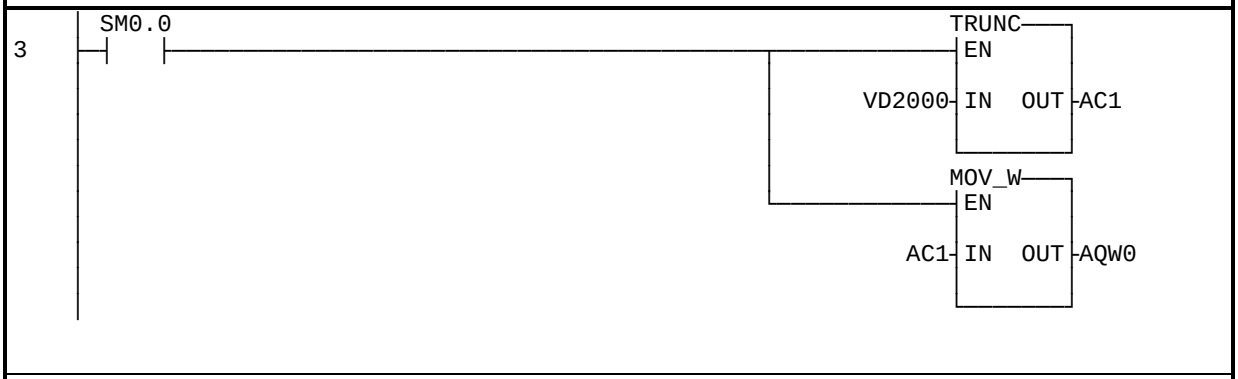
LD	**SM0.0**		**// Всегда**
MOVD	**0**	**AC1**	**// очистить AC1**
MOVW	**AIW0**	**AC1**	**// поместить слово аналогового входа 0 в AC1**
DTR	**AC1**	**VD500**	**// преобразовать значение в AC1 из десятичного в реальное**

```
     SM0.0                                                    MOV_DW
2   ─┤   ├──────────────────────────────────────────────────┬─┤EN
                                                             │
                                                     VD500-┤IN   OUT├-VD1000
                                                             │
                                                             MOV_DW
                                                             ├─┤EN
                                                             │
                                                     VD200-┤IN   OUT├-VD1004
                                                             │
                                                             MOV_DW
                                                             ├─┤EN
                                                             │
                                                     VD204-┤IN   OUT├-VD1008
                                                             │
                                                             MOV_DW
                                                             ├─┤EN
                                                             │
                                                     VD208-┤IN   OUT├-VD1012
                                                             │
                                                             MOV_DW
                                                             ├─┤EN
                                                             │
                                                     VD212-┤IN   OUT├-VD1016
                                                             │
                                                                0
                                                             ─( CALL   )
```

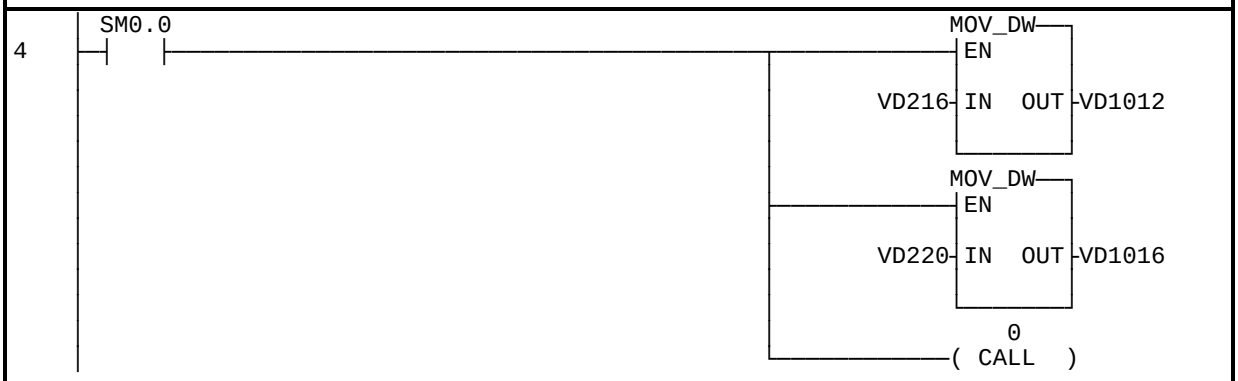
// Максимальные и минимальные масштабированные значения для AIW0, AQW0, и TD200
// д.б. помещены во временную область памяти для использования в подпрограмме 0.

//			
LD		**SM0.0**	**// Всегда**
MOVD	**VD500**	**VD1000**	**// поместить AIW0 в VD1000**
MOVD	**VD200**	**VD1004**	**// поместить AIW макс_масштаб значение (32000) в VD1004**

```
MOVD    VD204    VD1008    // поместить AIW миним_масштаб значение (3200) в VD1008
MOVW    VW208    VW1012    // поместить AQW макс_масштаб значение (32000) в VD1012
MOVD    VD212    VD1016    // поместить AQW миним_масштаб значение (0) в VD1016
CALL    0                  // вызвать подпрограмму 0
```




```
LD      SM0.0              // Всегда
TRUNC   VD2000   AC1       // округлить значение, посланное из SBR 0,
                           // и поместить его в AC1
MOVW    AC1      AQW0      // поместить значение в слово аналогового выхода
```




```
LD      SM0.0              // Всегда
MOVD    VD216    VD1012    // поместить TD200 макс_масштаб значение (10) в VD1012
MOVD    VD220    VD1016    // поместить TD200 миним_масштаб значение (0) в VD1016
CALL    0                  // вызвать подпрограмму 0
```

```
     | SM0.0                                                    MUL_R
5    |-| |-------------------------------------------------------|EN
     |                                                          |
     |                                                     VD2000-IN1 OUT-AC1
     |                                                  KR+100.000-IN2
     |                                                          |
     |                                                          TRUNC
     |                                                          |EN
     |                                                       AC1-IN  OUT-AC1
     |                                                          |
     |                                                          MOV_W
     |                                                          |EN
     |                                                       AC1-IN  OUT-VW116
     |                                                          |
     |                                                          V10.7
     |                                                          (     )
```

LD	**SM0.0**		**// Всегда**
MOVR	**VD2000**	**AC1**	**// поместить значение, посланное назад из SBR 0, в AC1**
***R**	**KR+100.000 AC1**		**// умножить значение на 100.00**
TRUNC	**AC1**	**AC1**	**// округлить значение в AC1**
MOVW	**AC1**	**VW116**	**// поместить значение в AC1 в встроенное слово**
			// значений дисплея TD200
=	**V10.7**		**// разрешить сообщение TD200**

```
     | VD2000        VD224                                        V10.6
6    |-------| >= R |-------------------------------------------(      )
     |
```

LDR>=	**VD2000**	**VD224**	**// Если значение из SBR 0 >= Верхнего предела,**
=	**V10.6**		**// разрешить сообщение “Макс. Предел Достигнут"**

```
     | VD2000        VD228                                        V10.5
7    |-------| <= R |-------------------------------------------(      )
     |
```

LDR<=	**VD2000**	**VD228**	**// Если значение из SBR 0 <= Нижнего Предела,**
=	**V10.5**		**// разрешить сообщение “ Миним. Предел Достигнут”**

```
     | V10.6    I0.0    Q0.1                                      Q0.0
8    |-| / |----|   |----| / |-------------------------------------(      )
     |
```

LDN	**V10.6**	**// Бункер не полон,**
A	**I0.0**	**// и активирован "Открыть Вход",**
AN	**Q0.1**	**// и выходной вентиль не открыт,**
=	**Q0.0**	**// открыть входной вентиль**

Copyright ©1996 by SIEMENS

```
     V10.5    I0.1    Q0.0                                          Q0.1
9   ─┤ / ├──┤    ├──┤ / ├──────────────────────────────────────(        )
```

LDN	**V10.5**	**// Уровень не ниже минимального,**
A	**I0.1**	**// и активирован "Открыть Выход"**
AN	**Q0.0**	**// и входной вентиль не открыт,**
=	**Q0.1**	**// открыть выходной вентиль**

```
10  ──────────────────────────────────────────────────────────────( MEND   )
```

MEND **// Конец**

```
 SBR:  0
     SM0.0                                                    SUB_R
12  ─┤    ├──────────────────────────────────────────────────┬──────┤EN
                                                             │
                                                   VD1012─IN1 OUT─AC1
                                                   VD1016─IN2
                                                             │
                                                             │  SUB_R
                                                             ├──────┤EN
                                                             │
                                                   VD1004─IN1 OUT─AC2
                                                   VD1008─IN2
                                                             │
                                                             │  SUB_R
                                                             └──────┤EN
                                                             │
                                                   VD1000─IN1 OUT─AC3
                                                   VD1008─IN2
```

// Подпрограмма 0:
//
// SBR 0 предназначена для проведения математических операций масштабирования
// аналогого слова входов для TD200 и AQW0.
// Формула следующая:
// Ov = [(Osh - Osl) * (Iv - Isl) / (Ish - Isl)] + Osl
//

SBR	**0**		**// Подпрограмма 0**
LD	**SM0.0**		**// Всегда**
MOVR	**VD1012**	**AC1**	**// поместить макс_масштаб значение в AC1**
-R	**VD1016**	**AC1**	**// и вычесть миним_масштаб значение из него**
MOVR	**VD1004**	**AC2**	**// поместить макс_масштаб значение в AC2**
-R	**VD1008**	**AC2**	**// и вычесть миним_масштаб значение из него**
MOVR	**VD1000**	**AC3**	**// поместить AIW0 в AC3**
-R	**VD1008**	**AC3**	**// и вычесть AIW миним_масштаб значение из него**

```
13  | SM0.0                                                  DIV_R
    |-| |-------------------------------------------------------|EN
    |                                                         |
    |                                                       AC3-IN1 OUT-AC3
    |                                                       AC2-IN2
    |                                                         |
    |                                                         MUL_R
    |                                                         |EN
    |                                                         |
    |                                                       AC3-IN1 OUT-AC3
    |                                                       AC1-IN2
    |                                                         |
    |                                                         ADD_R
    |                                                         |EN
    |                                                         |
    |                                                       AC3-IN1 OUT-VD2000
    |                                                     VD1016-IN2
    |                                                         |
```

LD	**SM0.0**		**// Всегда**
/R	**AC2**	**AC3**	**// разделить значение в AC3 на значение в AC2**
***R**	**AC1**	**AC3**	**// умножить значение в AC1 на AC3**
MOVR	**AC3**	**VD2000**	**// добавить значение в AC3 к VD1016**
+R	**VD1016**	**VD2000**	**// и поместить сумму в VD2000**

```
14  |-------------------------------------------------------------( RET )
```

RET	**// Возврат**

Copyright ©1996 by SIEMENS

```
//
// Блок Данных DB1:
// Блок параметров для TD200 для выдачи сообщений с масштабированными значениями
// измеренного уровня жидкости в бункере, и предупреждений, если достигнуты пределы
//

VB0     'TD'                    // Идентификатор TD200
VB2     16#10                   // Язык - Английский, обновлять так быстро, как возможно
VB3     16#00                   // Дисплей с 20 знаками
VB4     3                       // Число сообщений = 3
VB5     0                       // Маркер MB0 для F - клавиш

VW6     100                     // VB100 = Начало сообщения
VW8     10                      // VB10 = Адрес битов разрешения сообщений
VB100   'Liquid Level'          // Текст1 = 12 знаков Сообщения 1
VB112   '= '                    // '= ' и пробел
VB114   16#00                   // MSB байта формата - без обработки, без подтверждения
VB115   16#12                   // LSB байта формата - 1 слово с 2 цифрами
                                // после запятой и знаком
VB116   16#00                   // значение встроенного слова - MSB
VB117   16#00                   // значение встроенного слова - LSB

VB118   'm'                     // Текст2 = 1 знак и пробел
VB120   'Max. Level Reached  '  // Текст1 = 20 знаков Сообщения-2
VB140   'Min. Level Reached  '  // Текст1 = 20 знаков Сообщения-3
//
//  Инициализация:
//
VD200       32000.0     // AIW макс_масштаб
VD204       3200.0      // AIW миним_масштаб
VD208       32000.0     // AQW макс_масштаб
VD212       0.0         // AQW миним_масштаб
VD216       10.0        // TD200 макс_масштаб
VD220       0.0         // TD200 миним_масштаб
VD224       10.0        // Проверка Верхнего Предела
VD228       1.0         // Проверка Нижнего Предела
//
```

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все права защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC

S7-200 Примеры

Группа	Тема
6	**Измерение и мониторинг температуры при помощи S7-200 с использованием термосопротивления PT100**

Краткое описание

В данном примере обсуждается измерение температуры и мониторинг указанных пределов с использованием аналогого модуля расширения EM235, к одному из каналов которого подключено термосопротивление PT100.

Для того, чтобы преобразовать зависимое от температуры изменение сопротивления PT100 в напряжение, используется аналоговый выход как источник постоянного тока. Выход питает датчик PT100 постоянным током 12.5mA. С помощью этой цепи генерируется линейное входное напряжение 1mV/°C. EM235 преобразует данное напряжение в цифровое значение, которое циклически читается программой. Программа вычисляет температуру [(C], используя следующую формулу:

$$T[°C] = \frac{te - до}{t_1}$$

te = прочитанное и оцифрованное аналоговым модулем аналоговое значение измеряемой температуры, хранящееся в AIWx (x=0,2,4).
до = цифровое значение, измеренное при 0°C (0°C Смещение).

t_1 = изменение значения при увеличении температуры на 1°C.

Программа вычисляет значение до первой десятичной запятой и записывает результат в встроенную переменную Сообщения 1: ''Температура = xxx.x°C'', которое отображается на TD200. В сегменте инициализации программы пользователь может ввести верхний и нижний пределы температуры. Программа контролирует измеряемое значение и отображает предупреждение на TD200, если измеренная температура вышла из указанного диапазона. Если измеренная температура превысила вернюю границу температуры, то на второй строке дисплея TD200 появляется Сообщение 2: ''Температура > xxx.x°C''; если измеренная температура меньше нижнего предела температуры, то отображается Сообщение 3: ''Температура < xxx.x°C".

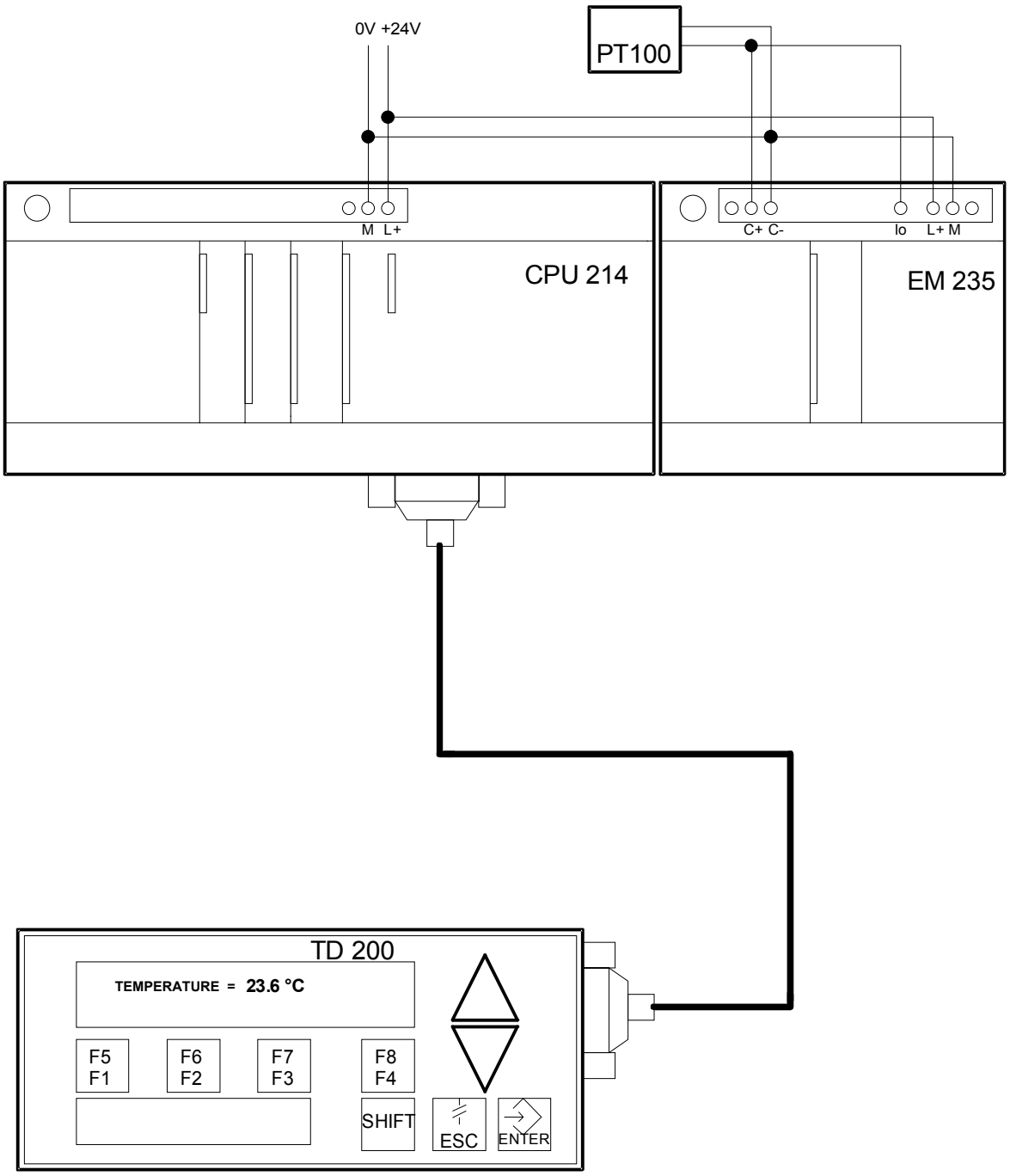

0V +24V
PT100
M L+
CPU 214
C+ C-
Io L+ M
EM 235
TD 200
TEMPERATURE = 23.6 °C
F5
F1
F6
F2
F7
F3
F8
F4
SHIFT
ESC
ENTER

Рисунок 39.1

Описание программы

Датчик температуры:

Использованный в данном примере термосопротивление Pt100 соответствуюет DIN IEC 751, предназначено для применения в диапазоне от -200 до +100°C. Температурная характеристика PT100 показанная на Рисунке 39.2 не является полностью линейной. Она отличается от прямой линии и отклоняется больше всего на границах температурного диапазона.

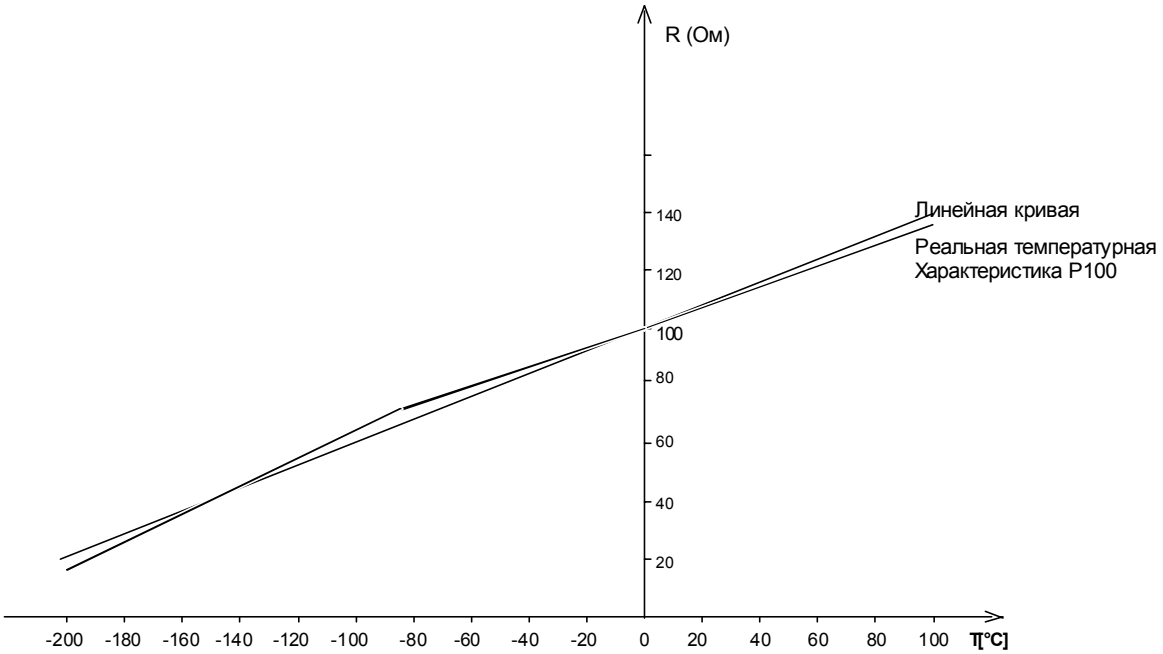


Рисунок 39.2

В диапазоне температур от -200°C до -130°C и от 0°C до 100°C, измеренная температура немного меньше реального значения и д.б. увеличена на соответвующее компенсационное значение. В диапазоне от -130°C до 0°C, измеренная температура немного больше реального значения и д.б. уменьшена на соответвующее компенсационное значение.

На рисунке 39.3 показано отклонение сопротивления в зависимости от температуры. В данном случае, весь температурный диапазон был поделен на 30 сегментов по 10°C каждый и было взято среднее значение отклонения для сегмента. В результате получено 30 значений отклонения, которые м.б. использованы в программе для линеаризации в качестве компенсационных значений для соответствующего температурного диапазона.

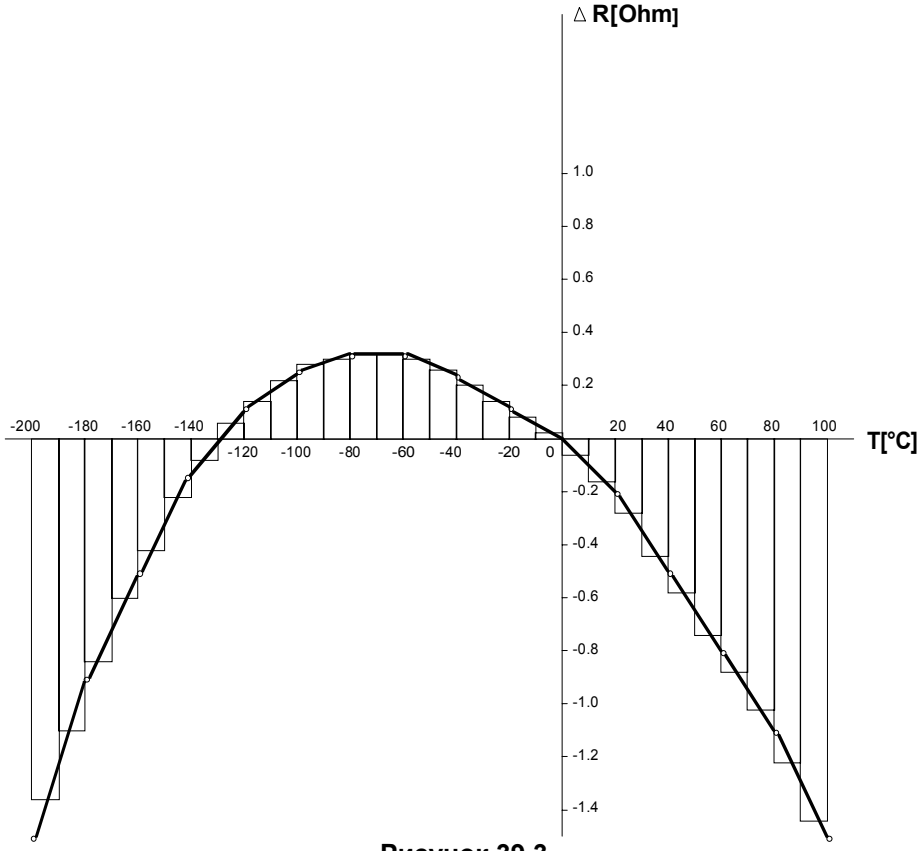


Рисунок 39.3

Т.к. изменение сопротивления на 0.4 Ома соответствует изменению температуры на 1°C, компенсационное значение м.б. преобразовано в °C и непосредственно добавлено к значению температуры, рассчитанное программой. На рисунке 39.4 приведены компенсационные значения для всех 30 температурных диапазонов.

ТемпДиапазон[°C]	-200 до -190	-190 до -180	-180 до -170	-170 до -160	-160 до -150
КомпЗначение [°C]	3.5	2.6	2.0	1.5	1.0
ТемпДиапазон[°C]	-150 до -140	-140 до -130	-130 до -120	-120 до -110	-110 до -100
КомпЗначение[°C]	0.6	0.1	-0.2	-0.4	-0.5
ТемпДиапазон[°C]	-100 до -90	-90 до -80	-80 до -70	-70 до -60	-60 до -50
КомпЗначение[°C]	-0.6	-0.7	-0.8	-0.8	-0.7
ТемпДиапазон[°C]	-50 до -40	-40 до -30	-30 до -20	-20 до -10	-10 до 0
КомпЗначение[°C]	-0.6	-0.5	-0.4	-0.2	-0.1
ТемпДиапазон[°C]	0 до 10	10 до 20	20 до 30	30 до 40	40 до 50
КомпЗначение[°C]	0.1	0.3	0.6	1.0	1.4
ТемпДиапазон[°C]	50 до 60	60 до 70	70 до 80	80 до 90	90 до 100
КомпЗначение[°C]	1.6	2.1	2.5	3.0	3.5

Рисунок 39.4

Компенсационные значения запоминаются во время инициализации в переменных и потом добавляются в программе к измеренным значениям теипературы. Для того, чтобы правильно выбрать компенсационное значение, программа вначале должна определить к какому из 30 диапазонов принадлежит измеренное значение.

Вычисление потребляемого тока для PT100:

PT100 при температуре 0C° имеет сопротивление 100 Ом. Сопротивление с ростом температуры изменяется линейно - около 0.4 Ом на градус Цельсия, таким образом тепловой коэффициент PT100 равен 0.4 Ома/°C. Для генерации коэффициента напряжения 1mV/°C, необходим источник тока 2.5mA. В выбранном диапазоне напряжения от 0 до 1V текущее разрешение аналогого выхода равно 10uA/Число. A count value of 250 is required до get 2.5mA. Т.к. формат слова данных из AQW сдвигается на 4 бита влево, вычисленное значение д.б. умножено на 16. В результате для инициализации тока I_0 2.5mA на аналоговом выходе в AQW0 заносится 4000.

Формула: (32000/20mA * 2.5mA = 4000)

EM235 Цепь:

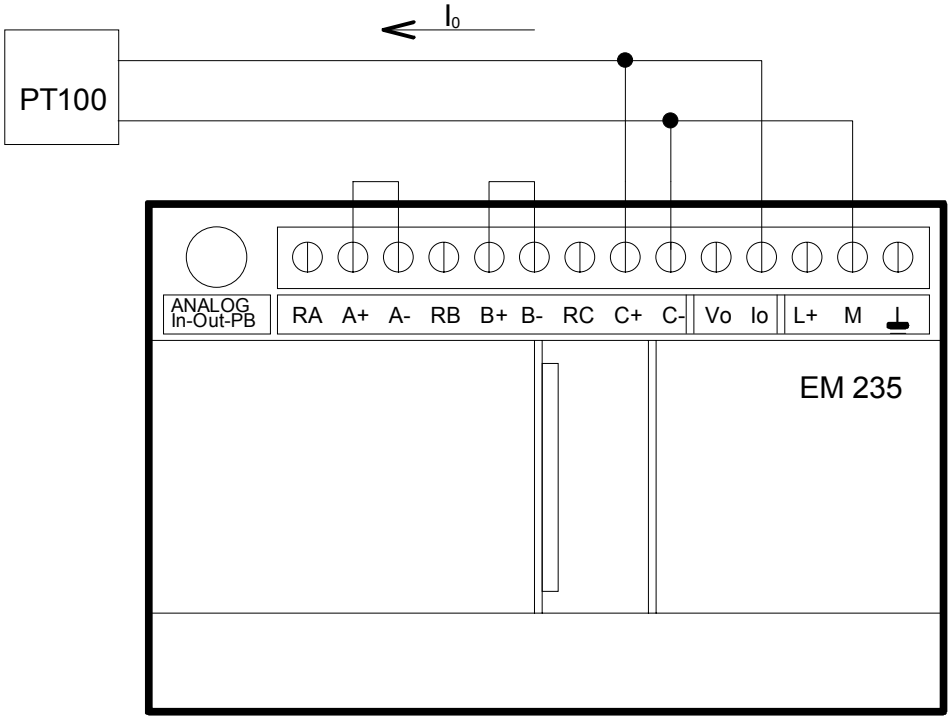


Рисунок 39.5

На модуле EM235 при помощи конфигурационных переключателей выбран диапазон напряжения 0..10V:

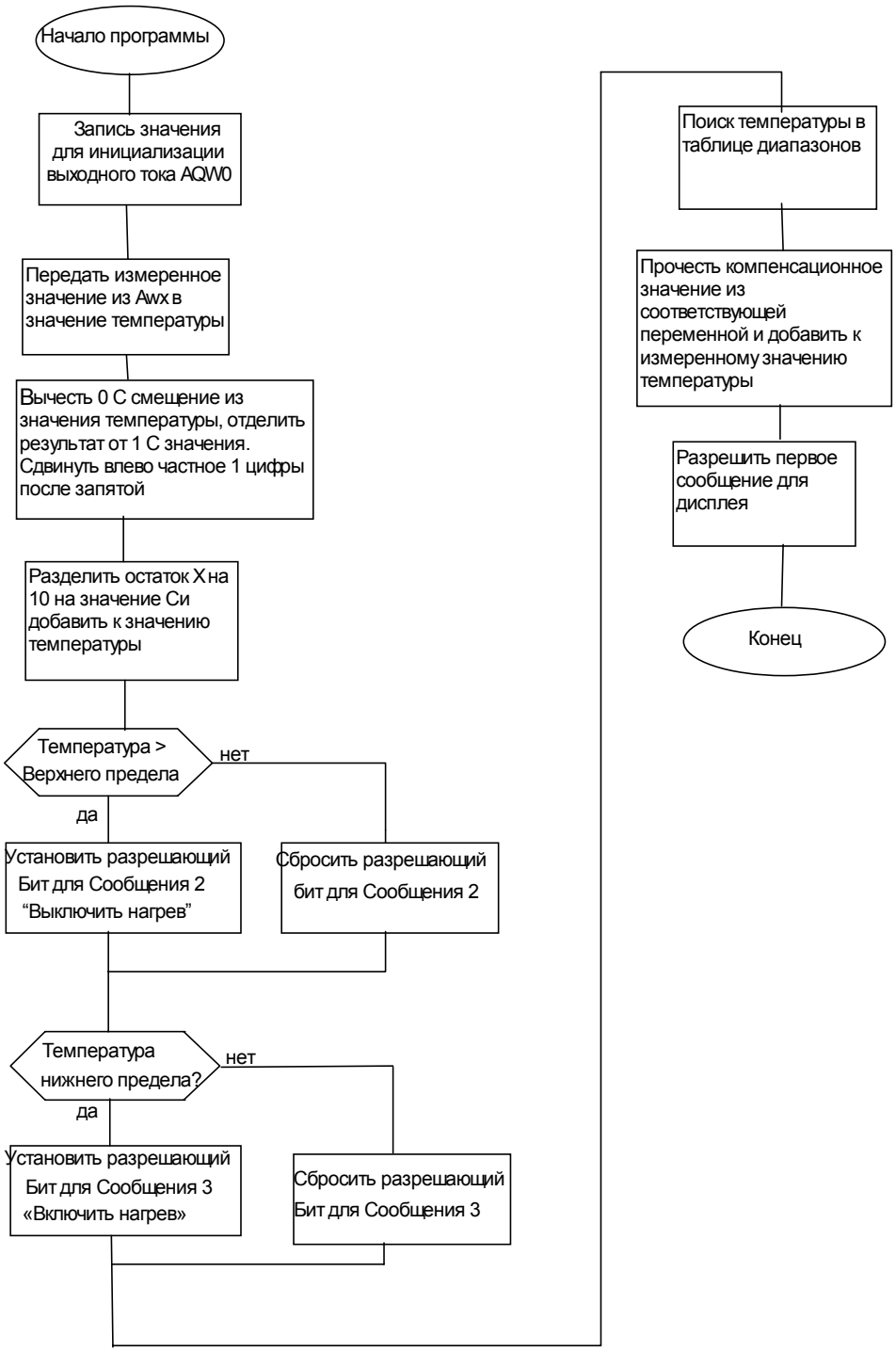
Конфигурационный переключатель №:	1	3	5	7	9	11
	ON	OFF	ON	OFF	ON	OFF

В программе, в зависимости от номера канала на EM235 д.б. использован соответствующий адрес AI слова:

AIW0 для входного канала 1, AIW2 для входного канала 2,
AIW4 для входного канала 3, AQW0 для входного канала 1

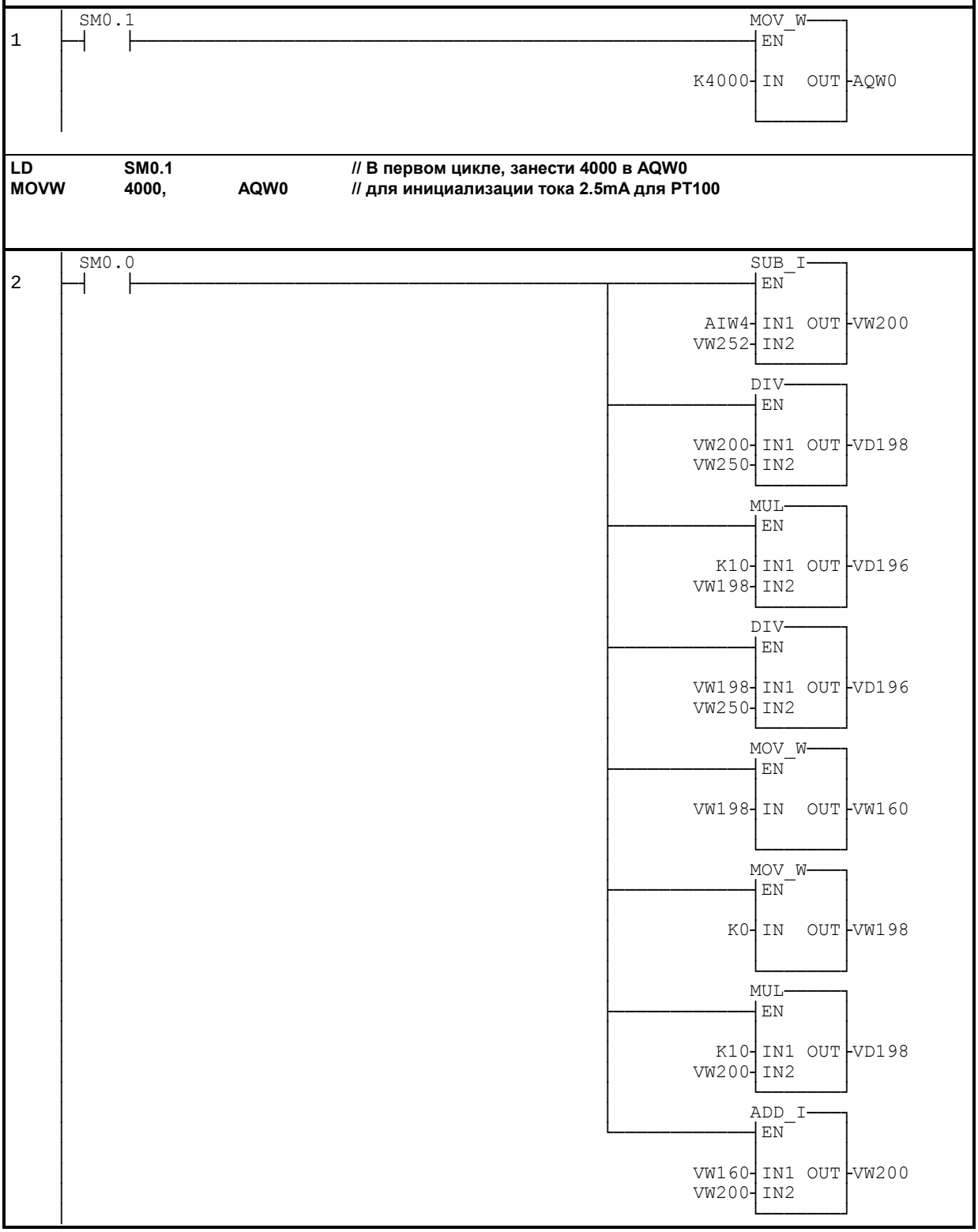
Неиспользуемые входы EM 235 д.б. закорочены.

Структура программы



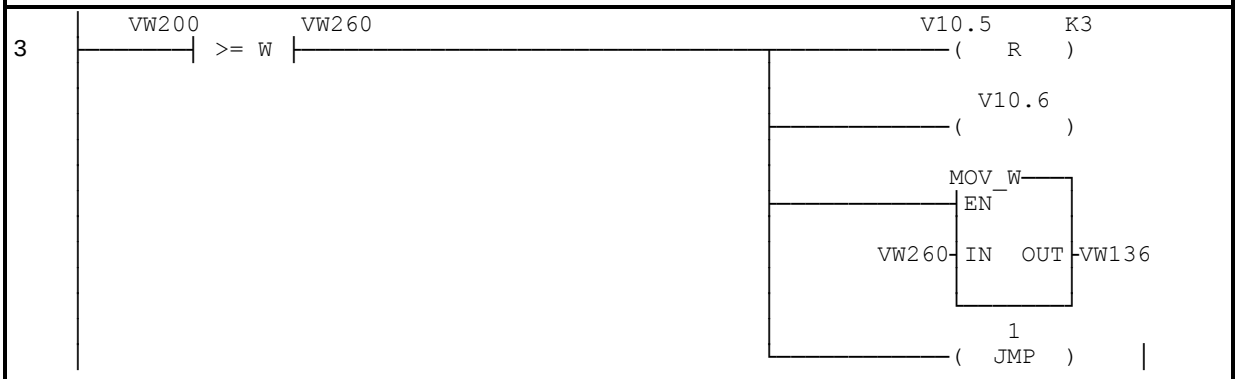
Описание программы

// ЗАГОЛОВОК: Измерение температуры с линеаризованным Pt100

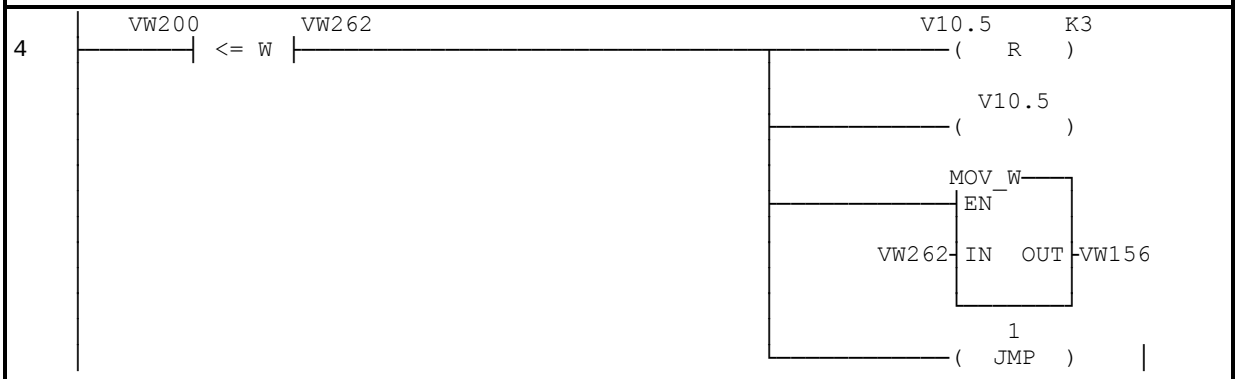



```
LD      SM0.1          // В первом цикле, занести 4000 в AQW0
MOVW    4000,    AQW0  // для инициализации тока 2.5mA для PT100
```

LD	SM0.0		// Всегда загружать измеренное
MOVW	AIW4,	VW200	// значение в VW200
-I	VW252,	VW200	// Вычесть 0(C смещение
DIV	VW250,	VD198	// Разделить значение на (C
MUL	10,	VD196	// Остаток умножить на 10. . .
DIV	VW250,	VD196	// Разделить на значение 1(C = 1 цифра после запятой
MOVW	VW198,	VW160	// Поместить VW198 во временную ячейку VW160
MOVW	0,	VW198	// Очистить VW198
MUL	10,	VD198	// Температуру VW200 x 10
+I	VW160,	VW200	// Температура x 10 + 1 цифра после запятой = температура



LDW>=	VW200,	W260	// Проверка, выходит ли значение за верхний предел
R	V10.5,	3	// Сбросить все три сообщения TD200
=	V10.6		// Разрешить сообщение TD200, “Температура>“
MOVW	VW260,	VW136	// Поместить верхний предел в переменную дисплея TD200
JMP	1		// Перейти к MEND



LDW<=	VW200,	VW262	// Проверка, выходит ли значение за нижний предел
R	V10.5,	3	// Сбросить все три сообщения TD200
=	V10.5		// Разрешить сообщение TD200, “Температура <“
MOVW	VW262,	VW156	// Поместить нижний предел в переменную дисплея TD200
JMP	1		// Перейти к MEND

```
     SM0.0                                                    MOV_DW
5  ──┤  ├──────────────────────────────────────────────────┬──┤EN
                                                            │
                                                            │  K0-IN  OUT-AC1
                                                            │
                                                            │   TBL_FIND
                                                            ├──┤EN
                                                            │
                                                            │ VW398-SRC
                                                            │ VW200-PTN
                                                            │   AC1-IDX
                                                            │     4-CMD
                                                            │
                                                            │   MOV_DW
                                                            ├──┤EN
                                                            │
                                                            │&VB300-IN  OUT-AC2
                                                            │
                                                            │   MUL
                                                            ├──┤EN
                                                            │
                                                            │  K2-IN1 OUT-AC1
                                                            │ AC1-IN2
                                                            │   ADD_DI
                                                            ├──┤EN
                                                            │
                                                            │ AC1-IN1 OUT-AC2
                                                            │ AC2-IN2
                                                            │   ADD_I
                                                            ├──┤EN
                                                            │
                                                            │ *AC2-IN1 OUT-VW116
                                                            │ VW200-IN2
                                                            │
                                                            │ V10.7     K1
                                                            └──────( S )
```

LD	**SM0.0**		**// Всегда**
MOVD	**0,**	**AC1**	**// загружать начальный адрес для поиска в AC1**
FND>	**VW398,**	**VW200**	**AC1 // команда поиска в таблице**

// Данная команда начинает поиск в Таблице B с VW398 пока не найдет значение,
// хранящееся в VW200. Затем она помещает смещение или индекс в AC1.

MOVD	**&VB300,**	**AC2**	**// Загрузить начальный адрес Таблицы A в AC2**
MUL	**2,**	**AC1**	**// Область хранения длиной два байта. Умножить индекс на 2**
+D	**AC1,**	**AC2**	**// Добавить индекс к начальному адресу для**
			// получения абсолютного адреса
MOVW	***AC2,**	**VW116**	**// Поместить корректировачное значение в VW116**
+I	**VW200,**	**VW116**	**// Добавить корректировачное значение к температуре для**
			// получения реального значения
S	**V10.7,**	**1**	**// Разрешить сообщение TD200, “Температура=“**

```
 LBL:  1
 ┌──────┐
 │
7 ├──────────────────────────────────────────────────────────( MEND  )
```

LBL	**1**	**// Метка 1**
MEND		**// Конец**

Блок Данных DB1:

```
// Блок параметров для TD200 для отображения сообщений со встроенным значением
// измеренной температуры и предупреждений со встроенными температурными пределами.

VB0     'TD'                // Идентификатор TD200
VB2     16#10               // Язык - Английский, обновлять так быстро, как возможно
VB3     16#00               // Дисплей с 20 знаками
VB4     3                   // Число сообщений = 3
VB5     0                   // Маркер MB0 для F - клавиш
VB6     0
VB7     100                 // VB100 = Начало сообщения
VB8     0
VB9     10                  // VB10 = Адрес битов разрешения сообщений

VB100   'Temperature='      // Текст1 = 12 знаков Сообщения 1
VB112   ' '                 // Два пробела
VB114   16#00               // MSB байта формата - без обработки, без подтверждения
VB115   16#11               // LSB байта формата - 1 слово с 1 цифрой после запятой
VB116   16#00               // значение встроенного слова - MSB
VB117   16#00               // значение встроенного слова - LSB
VB118   16#DF               // Текст2 = 2 знака
VB119   'C'
VB120   'Temperature>'      // Text1 = 12 знаков Сообщения 2
VB132   ' '                 // Два пробела
VB134   16#00               // MSB байта формата - без обработки, без подтверждения
VB135   16#11               // LSB байта формата - 1 слово с 1 цифрой после запятой
VB136   16#00               // значение встроенного слова - MSB
VB137   16#00               // значение встроенного слова - LSB
VB138   16#DF               // Text2 = 2 знака
VB139   'C'

VB140   'Temperature<'      // Text1 = 12 знаков Сообщения 3
VB152   ' '                 // Два пробела
VB154   16#00               // MSB байта формата - без обработки, без подтверждения
VB155   16#11               // LSB байта формата - 1 слово с 1 цифрой после запятой
VB156   16#00               // значение встроенного слова - MSB
VB157   16#00               // значение встроенного слова - LSB
VB158   16#DF               // Текст2 = 2 знака
VB159   'C'
//
// Инициализация
VD196       0               // Очистить VW196 и VW198
VW250       32              // Загрузить 1mV/°C = 32 в VW250
VW252       8000            // 0°C смещение = 8000 в VW252
VW260       1000            // Верхний предел температуры = +100°C
VW262       -2000           // Нижний предел температуры = -200°C
```

```
// Таблица A: Компенсация температуры для указанных диапазонов

VW300     35  // -200°C до -190°C
VW302     26  // -190°C до -180°C
VW304     20  // -180°C до -170°C
VW306     15  // -170°C до -160°C
VW308     10  // -160°C до -150°C
VW310      6  // -150°C до -140°C
VW312      1  // -140°C до -130°C
VW314     -2  // -130°C до -120°C
VW316     -4  // -120°C до -110°C
VW318     -5  // -110°C до -100°C
VW320     -6  // -100°C до -90°C
VW322     -7  // -90°C до -80°C
VW324     -8  // -80°C до -70°C
VW326     -8  // -70°C до -60°C
VW328     -7  // -60°C до -50°C
VW330     -6  // -50°C до -40°C
VW332     -5  // -40°C до -30°C
VW334     -4  // -30°C до -20°C
VW336     -2  // -20°C до -10°C
VW338     -1  // -10°C до 0°C
VW340      1  //  0°C до 10°C
VW342      3  //  10°C до 20°C
VW334      6  //  20°C до  30°C
VW346     10  //  30°C до 40°C
VW348     14  //  40°C до 50°C
VW350     16  //  50°C до 60°C
VW352     21  //  60°C до 70°C
VW354     25  //  70°C до 80°C
VW356     30  //  80°C до 90°C
VW358     35  //  90°C до 100°C
//

// Таблица B: таблица поиска для подсчета индекса

VW400          -1900          // -190°C
VW402          -1800          // -180°C
VW404          -1700          // -170°C
VW406          -1600          // -160°C
VW408          -1500          // -150°C
VW410          -1400          // -140°C
VW412          -1300          // -130°C
VW414          -1200          // -120°C
VW416          -1100          // -110°C
VW418          -1000          // -100°C
VW420          -900           // -90°C
VW422          -800           // -80°C
VW424          -700           // -70°C
VW426          -600           //  -60°C
VW428          -500           // -50°C
VW430          -400           // -40°C
VW432          -300           // -30°C
VW434          -200           // -20°C
VW436          -100           // -10°C
VW438          0              // 0°C
VW440          100            // 10°C
VW442          200            // 20°C
VW444          300            // 30°C
VW446          400            // 40°C
VW448          500            // 50°C
VW450          600            // 60°C
VW452          700            // 70°C
VW454          800            // 80°C
VW456          900            // 90°C
VW458          1000           // 100°C
//
```

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все права защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC
S7-200 Примеры

Группа	Пример к теме
4	**Использование S7-200 AC/AC/AC со входами ~ 220В для управления двигателя крана**

Обзор

Данный пример показывает возможность, mittels der Standard-110VAC-Eingdnge der S7-200 CPU 220VAC-Eingdnge einzusetzen (Внимание: Обратите внимание на предупреждение). В данном примере S7-214 CPU версия /AC/AC/AV управляет асинхронным двигателем крана посредством выходов ~ 220В. (Указание: S7-212 тоже может быть использован.)

Предупреждение

Die S7-200 выполнен по Категории 2 для перенапряжения при степени загрязнения 2 для входов ~ 120В. Входы ~220В можно подключать к S7-200 без опасений. Однако не существует простого, общего описания по установке, которое бы удовлетворяло названным нормам в любой ситуации.

Некорректная установка ПЛК может привести к смерти телесным повреждениям и/или порче имущества.

Если вы используете ~ 220В, то Вы должны или уменьшить степень загрязнения при установке до степени 1 (т.е., чистая, сухая установка по IEC 664), или категорию перенапряжения окружающей среды снизить до категории 1. Это означает, что приходящий сигнал (~220В) имеет положительно-транзиентное ограничение, для того чтобы быть уверенным, что молниезащита / защита от напряжения коммутации ограничена 1360В с замыканием на землю.

Схема включения

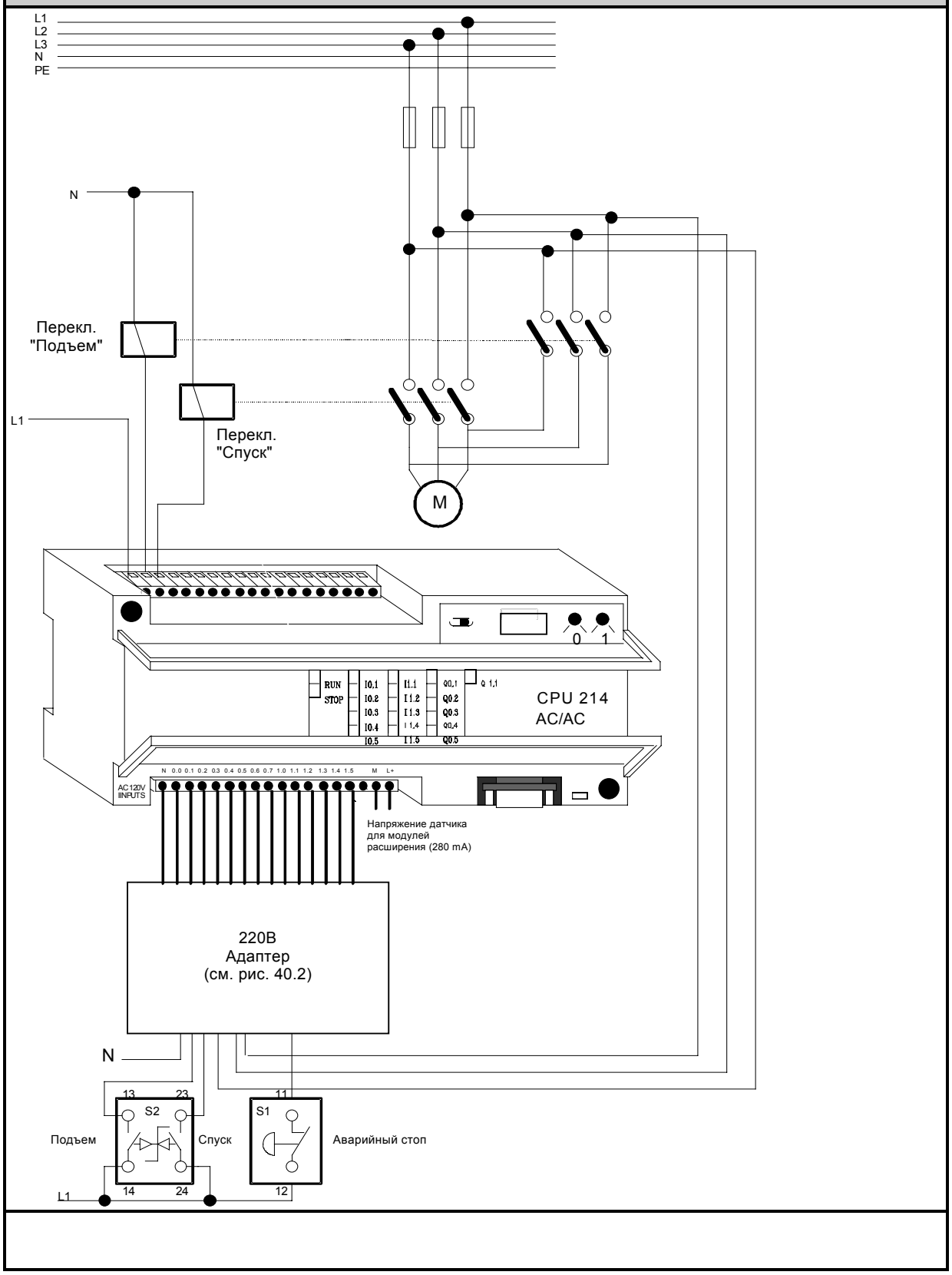


Рис. 40.1

Указания по подключению входов ~ 220В

Подключение S7-200 AC/AC/AC со входами ~ 220В

AC-версия S7-200 была разработана для американского рынка и не может быть использована в Европе просто так. Причина этого в том что входы CPU, расчитаны максимально на ~135 В / 47-63Гц. Поэтому европейское напряжение питания ~220-240В / 50Гц не может быть подключено непосредственно на входы.

Так как имеется возможность подать 220В переменного тока на блок питания и выходы, то входы остаются единственным препятствием для подключения к европейскому напряжению питания. Проблему можно решить повысив импеданс со стороны, как это показано на рисунке 40.2.

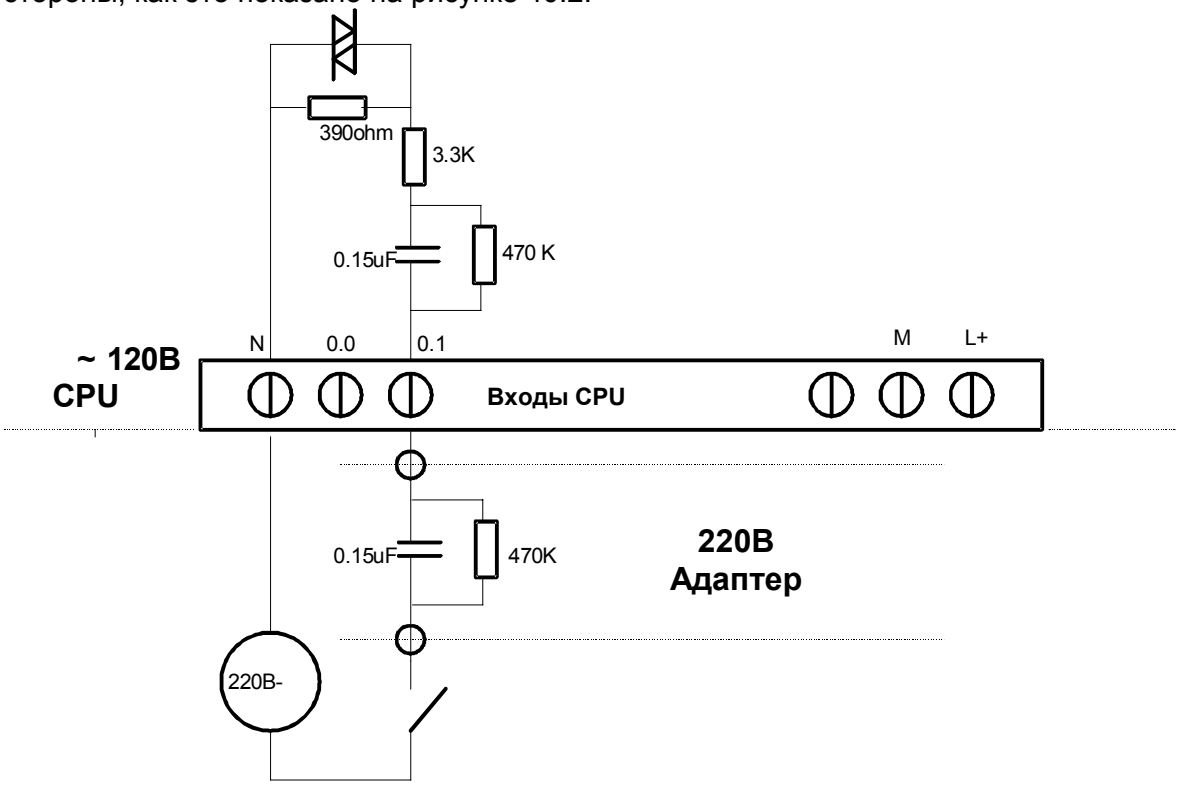
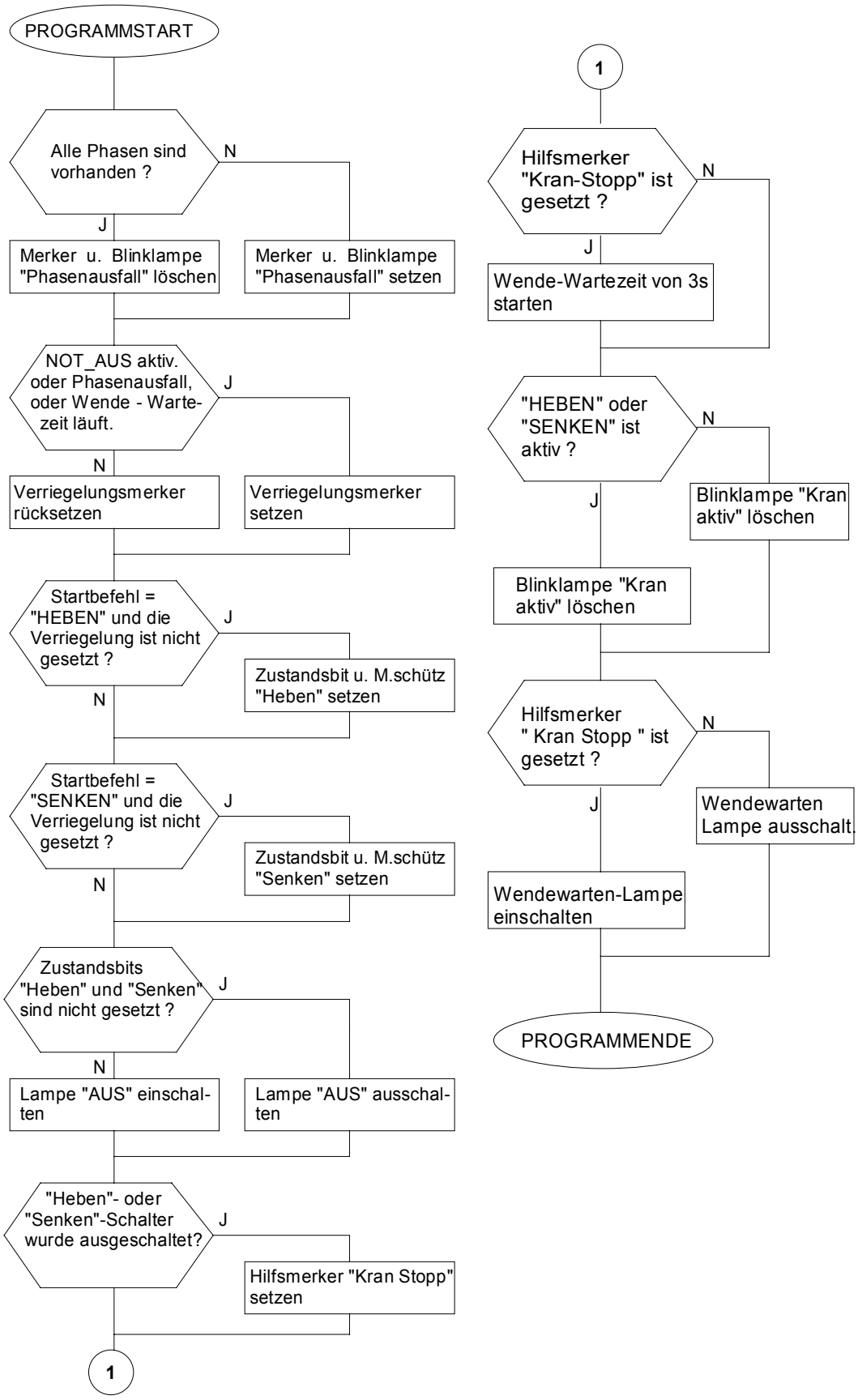


Рис 40.2: ~220В Адаптер

На каждый вход подключается элемент крепящийся на стандартную профильную шину и состоящий из конденсатора 0,15мкФ/400В и сопротивления 470К/0,25Вт включенных параллельно. При50 Гц сопротивление обоих компонентов почти также велико, как внутреннее сопротивление; это означает, что разделенное переменное напряжение на входе CPU составит около 115В. Крепежные блоки MKS-M с клеммами под винт, изготавливаемые фирмой Murr Electronics, удовлетворяют всем требованиям безопасности по DIN VDE.

Структура программы



Описание программы вкл. листинг

Движение крана индицируется мигающей лампой. Направления движения ‘’Подъем’’ и ‘Спуск’’ индицируются двумя светодиодами (LED). Привод крана реализован на асинхронном двигателе с двумя переключателями. Двигатель управляется реверсивной кнопкой, связанной со входами E0.0 und E0.1. ‘’Подъем’’ производится по положительному фронту входа E0.0, ‘’Спуск’’ по положительному фронту на входе E0.1. При переключении направления вращения двигателя отрицательный фронт на E0.0 или на E0.1 запускает таймер ожидания 3 секунды. В течение этого времени не может быть включено движение в противоположную сторону, так что двигатель имеет время на торможение.

Программа контролирует также фазы L1, L2 L3 непосредственно на входе двигателя. Если одна из трех фаз выпала, то питание двигателя немедленно отключается CPU. Немедленно отключить двигатель можно также нажатием кнопки Аварийного останова на E0.7.

Операционный блок OB1:

// TITEL = Управление двигателем крана

// Контроль переменного тока

```
LDN     E0.2     // Фаза L1 пропала
ON      E0.3     // Фаза L2 пропала
ON      E0.4     // Фаза L3 пропала
=       M2.7     // Пропадание фазы
```

```
     │  E0.2                                                    M2.7
1    ├─┤ / ├──────────────────────────────────────────────────(      )
     │      │
     │  E0.3 │
     ├─┤ / ├─┤
     │      │
     │  E0.4 │
     ├─┤ / ├─┘
     │
```

// Активирование лампы "Пропадание фазыl"

```
LD      M2.7     // Установлен меркер "Пропадание фазы"
U       SM0.5    // Мигающий меркер 1сек
=       A0.5     // Мигание лампы " Пропадание фазы"
```

```
     │  M2.7   SM0.5                                          A0.5
2    ├─┤   ├───┤   ├──────────────────────────────────────────(      )
     │
```

// Блокировка

```
LDN     E0.7     // Активирован аварийный стоп
O       M2.3     // или работает таймер ожидания реверса
O       M2.7     // или пропадание фазы
=       M2.0     //Установить блокировку
```

```
3   |  E0.7                                                          M2.0
    |-| / |-------------------------------------------------------(        )
    |      |
    |  M2.3 |
    |-|   |-|
    |      |
    |  M2.7 |
    |-|   |-|
    |
```

```
// Запуск подъема
LD          E0.0                    // Команда "Подъем"
UN          M2.0                    // и нет блокировки
UN          A0.1
=           A0.0                    // Активирован переключатель "Подъем"
```

```
4   |  E0.0    M2.0   A0.1                                    A0.0
    |-|   |---| / |--| / |-----------------------------------(        )
    |
```

```
// Запуск спуска
LD          E0.1                    //Команда "Спуск"
UN          M2.0                    // и нет блокировки
UN          A0.0
=           A0.1                    // Активирован переключатель "Спуск"
```

```
5   |  E0.1    M2.0   A0.0                                          A0.1
    |-|   |---| / |--| / |-----------------------------------------(        )
    |
```

```
// Лампа ВЫКЛ
LDN         A0.0                    // Нет состояния "Подъем"
UN          A0.1                    // Нет состояния "Спуск"
=           A0.7                    // Активирована лампа "ВЫКЛ"
```

```
6   |  A0.0    A0.1                                                A0.7
    |-| / |---| / |-------------------------------------------------(        )
    |
```

```
// Установка вспомогательного меркера при выключении
LD          E0.0                    // "Подъем" остановлен
ED
LD          E0.1                    // или "Спуск"
ED                                  // остановлен
OLD
S           M2.3,1                  // Установка вспомогательного меркера
```

```
     E0.0                                                          M2.3     K1
7   ├─┤   ├──┤ N ├──┬──────────────────────────────────────────────(   S   )
     E0.1            │
    ├─┤   ├──┤ N ├──┘
```

// Запуск таймера ожидания при изменении направления движения

LD	**M2.3**	**// Установка вспомогательного меркера**
TON	**T33,300**	**// Запуск T33**
U	**T33**	**// Если таймер завершил работу,**
R	**M2.3,1**	**// сбросить вспомогательный меркер**

```
     M2.3                                                          TON──T33
8   ├─┤   ├──┬──────────────────────────────────────────────────┤IN      │
                                                                 K300-PT    │
                                                                 └──────┘
              T33                                                M2.3     K1
             └─┤   ├──────────────────────────────────────────────(   R   )
```

//Установка мигающей лампы "Кран работает"

LD	**A0.0**	**// "Подъем" активен**
O	**A0.1**	**// или "Спуск" активен**
U	**SM0.5**	**// Мигающий меркер 1 сек**
=	**A0.3**	**// Установить мигающую лампу**

```
     A0.0   SM0.5                                                    A0.3
9   ├─┤   ├──┬─┤   ├──────────────────────────────────────────────(       )
     A0.1    │
    ├─┤   ├──┘
```

//Активирование лампы "Ожидание реверса"

LD	**M2.3**	**// Установлен вспомогательный меркер**
=	**A0.6**	**// Установить лампу реверса**

```
     M2.3                                                            A0.6
10  ├─┤   ├──────────────────────────────────────────────────────────(       )
```

MEND

```
11  ├──────────────────────────────────────────────────────────────( MEND  )
```

Указания по преобразованию

Для того чтобы преобразовать TOOLITE2 AWL в S7-Micro/DOS AWL

- Установите ‘K’ перед каждым числом, не являющимся 16-ричной константой (напр. 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех 16-ричных констант (напр. 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS AWL в KOP-форму нужно начинать каждый сегмент словом ‘NETWORK’ и номером. Каждый сегмент в этом примере имеет свой номер на диаграмме KOP. Используйте NWENFG в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведённого здесь.

За правильную работу системы заказчик несёт ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC

S7-200 Примеры

Группа	Тема
3	**Modbus RTU Slave для S7-214**

Краткое описание

Данный пример включает группу подпрограмм и программ обработки прерываний для использования S7-214 в режиме свободнопрограммируемого интерфейса в качестве Modbus RTU slave. Данная программа поддерживает следующие функции Modbus:

1 Чтение выходов
2 Чтение входов
3 Чтение управляющих регисторов (V память)
4 Чтение входных регисторов
5 Запись одного выхода
6 Запись одного управляющего регистра
15 Запись нескольких выходов
16 Запись нескольких управляющих регистров

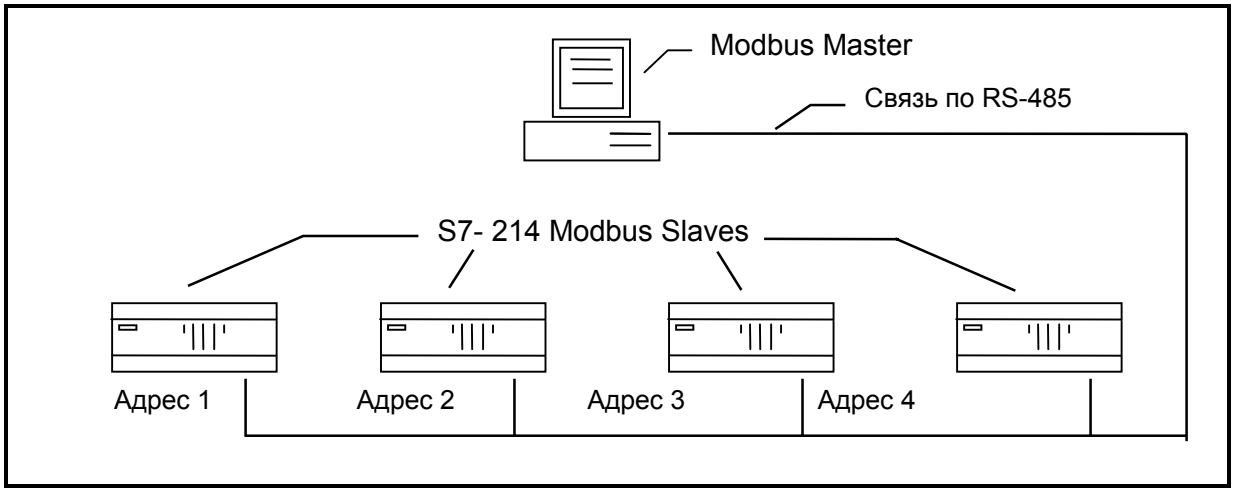


Рисунок 41.1

Структура программы

Драйвер протокола Modbus состоит из группы подпрограмм и программ обработки прерываний, которые инициализируют и обрабатывают запросы Modbus. Программа пользователя состоит из двух сегментов. Один сегмент инициализирует в первом цикле драйвер Modbus. Другой сегмент контролирует М-бит и обрабатывает возникающие запросы Modbus. Этот второй сегмент д.б. помещен в конце программы пользователя (непосредственно перед MEND), так чтобы данные изменялись только в конце цикла.

Список используемых подпрограмм и прерываний:

SBR	50	Инициализация драйвера Modbus RTU
SBR	51	Обработка запроса Modbus и передача ответа
SBR	52	Обработка функции Modbus 1 и 2
SBR	53	Обработка функции Modbus 3 и 4
SBR	54	Обработка функции Modbus 5
SBR	55	Обработка функции Modbus 6
SBR	56	Обработка функции Modbus 15
SBR	57	Обработка функции Modbus 16
SBR	61	Генерация ошибки отклика 2
SBR	62	Инициализация таблицы CRC
SBR	63	Подсчет CRC
INT	120	Обработка тайм-аута свободной линии
INT	121	Обработка символов пока ожидание тайм-аута свободной линии
INT	122	Прием первого символа (поле адреса) запроса
INT	123	Прием остальной части запроса
INT	124	Прерывание запроса после тайм-аута свободной линии
INT	125	Сброс поиска свободной линии после завершения передачи

Описание программы

Данная программа позволяет подключить один или больше S7-214 к главной станции Modbus. Программа использует функции свободнопрограммируемого интерфейса S7-214 для реализации протокола Modbus RTU. Modbus RTU - протокол master-slave; это означает, что структура сети состоит из одной активной станции (главной ЭВМ) и одной или нескольких пасивных станций. Каждая пасивная станция имеет уникальный адрес. Активная станция посылает запрос оной из пасивных станций и ожидает ответа от нее. Пасивная станция отвечает на полученный запрос или возникает ошибка. Если запрос был принят неверно, то возникает ошибка передачи, такая как ошибка четности или неверная CRC (контрольная сумма); пассивная станция не отвечает и активная станция должна повторить запрос после соответствующего времени ожидания.

Modbus RTU - двоичный протокол. Начало телеграммы обозначается временем свободной линии для 3.5 байт на текущей скорости обмена. Конец телеграмы так же обозначается тем же временем свободной линии. Т.к. время свободной линии - складывается из времени передачи символов, оно сильно зависит от скорости передачи. Программа, описанная ниже устанавливает время свободной линии равное значению соответствующему при 9600 Бод. Если скорость обмена меняется, то время свободной линии также д.б. изменено. Это описано в SBR 50.

Протокол Modbus RTU передает данные в 8 битном двоичном коде. Каждый символ также включает один стартовый бит, один или два стоповых бита (S7-214 поддерживает один стоповый бит) и опционально бит четности. Программа, описанная ниже, использует S7-214 в режиме 9600 Бод с четностью. Эти значения м.б. изменены в установках порта в SBR 50.

Протокол Modbus RTU использует CRC (Cyclical Redundancy Check - контроль при помощи циклического избыточного кода) для обнаружения ошибок. Данный пример использует таблицу со значениями CRC для ускорения вычисления CRC при проверки полученной телеграммы и

передачи ответа. Таблица CRC создается в верхней V памяти во время инициализации драйвера Modbus и требует около 700 милисекунд. Это происходит только во время первого цикла.

Для поддержки функций Modbus 1, 2, 3, 4, 5, 6, 15 и 16 используются соответствующие подпрограммы. Если конкретная активная станция не использует всех этих функций, их можно убрать, чтобы увеличить память доступную программе пользователя. Для этого необходимо убрать соответствующую подпрограмму и ее вызов. Все вызовы находяться в SBR 51.

Поддерживаются следующие функции:

1 Чтение состояния одного/нескольких выходов
- Возвращает состояние включено/выключено любого числа выходов (Q). Максимальное число выходов м.б. задано пользователем (см. ниже).

2 Чтение состояния одного/нескольких входов
- Возвращает состояние включено/выключено любого числа входов (I). Максимальное число входов м.б. задано пользователем (см. ниже).

3 Чтение одного/нескольких управляющих регистров
- Возвращает содержимое V памяти. Подразумевается, что управляющий регистр д.б. переменной для Modbus типа слово. Эта область начинается с V0. Размер (в словах) входов м.б. задан пользователем (см. ниже).

4 Чтение одного/нескольких входных регистров
- Чтение аналоговых входов. Эта функция возвращает содержимое области V памяти, отделенное от управляющих регистров. Пользователь должен добавить команды чтения слов AI и скопировать их, если необходимо, в V память. Область V памяти, используемая для этих команд, м.б. задана пользователем (см. ниже).

5 Управление одним выходом
- Запись регистра отображения выхода (Q). Выход, в действительности, не управляется, а только записывается.

6 Запись одного управляющего регистра
- Запись слова в V память.

15 Управление несколькими coil (выходами)
- Запись нескольких выходов (Q). Область выходов должна начинаться на границе байта (например, Q0.0 или Q2.0) и число записываемых выходов д.б. кратно восьми. Это не требования Modbus, а сделано для простоты реализации. Выхода, в действительности, не управляются, а только записывается регистр отображения выходов.

16 Запись нескольких управляющих регистров
- Запись нескольких слов в V память. За один запрос м.б. записано до 60 слов.

Следующие области памяти используются для конфигурирования драйвера Modbus. Эти области инициализируются в SBR 50 и м.б. изменены пользователем, чтобы изменить размер области памяти, доступной главной ЭВМ через драйвер Modbus.

VW3290 Максимальное число входов/выходов, доступных через Modbus.
Имеет значение для функций 1, 2, 5 и 15. Значение по умолчанию - 64 (I0.0 - I7.7 и Q0.0 - Q7.7).

VW3294 Максимальное число входных регистров, доступных через Modbus функцию 4. Значение по умолчанию - 16.

VW3296 Адрес в V памяти, куда будут при ответе считаны AI для Modbus функции 4. Значение по умолчанию - VB2000.

VB4095 Modbus адрес пассивной станции. Значение по умолчанию - адрес 1.

Следующие области V памяти используются драйвером Modbus и *не* должны изменяться пользователем.

M31.7 Меркер, используемый для сигнализации, что был получен запрос Modbus.

VB3300 - VB3559 Буфер драйвера

VB3560 - VB3575 Различные области рабочей памяти для Modbus

VB3580 - VB4091 Таблица данных CRC

Подпрограммы 50 - 63 зарезервированы для драйвера Modbus.

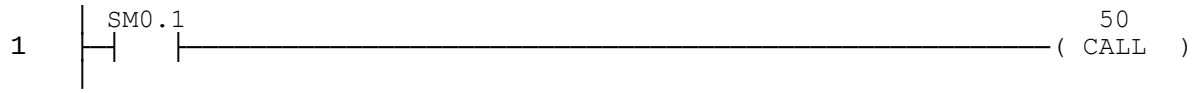
Прерывания 120 - 127 зарезервированы для драйвера Modbus.

Метки 254 и 255 зарезервированы для драйвера Modbus.

LAD (S7-MicroDOS)	STL (IEC)

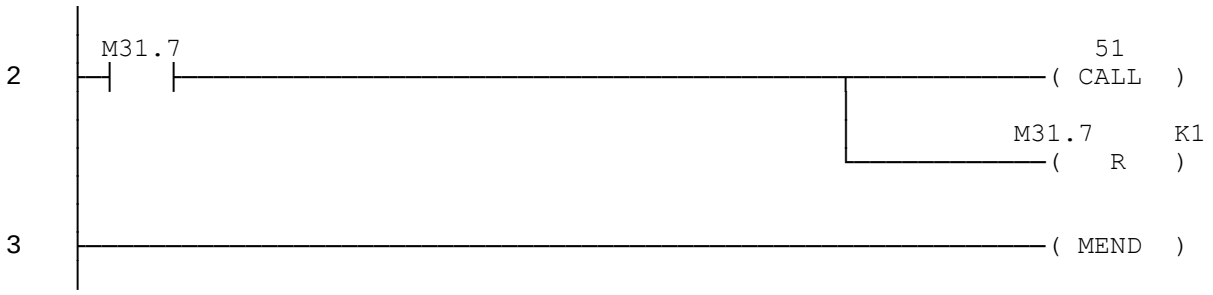
Главная программа

// Вызов подпрограмм инициализации драйвера Modbus во время первого цикла.
// Этот сегмент должен находиться в начале программы.




```
LD      SM0.1      // Если первый цикл,
CALL    50         // инициализация драйвера Modbus
```

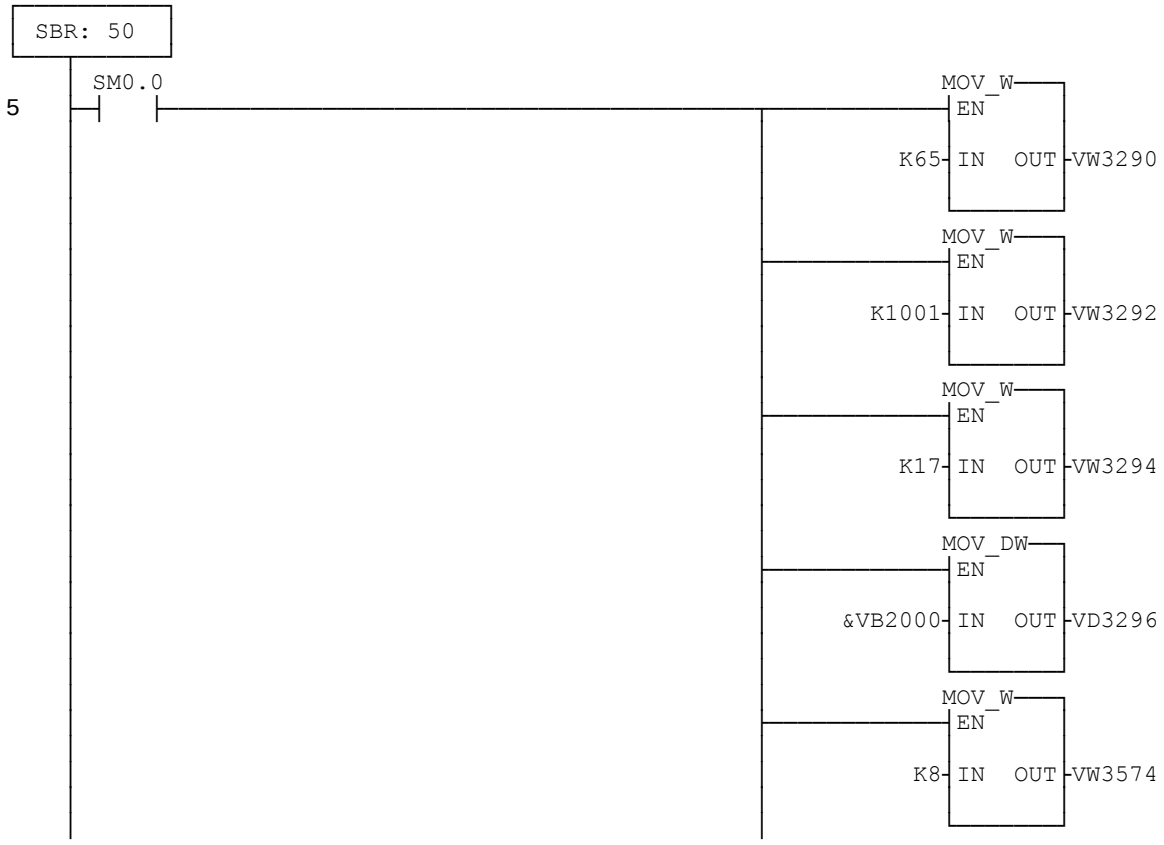
// Проверка получен ли запрос Modbus. Если получен, то программа
// обработки полученного прерывания устанавливает M31.7.
// Этот сегмент должен находиться в конце программы.




```
LD      M31.7      // Если есть Modbus запрос,
CALL    51         // вызов обработчика и
R       M31.7,1    // сброс меркера запроса

MEND
```

Подпрограммы

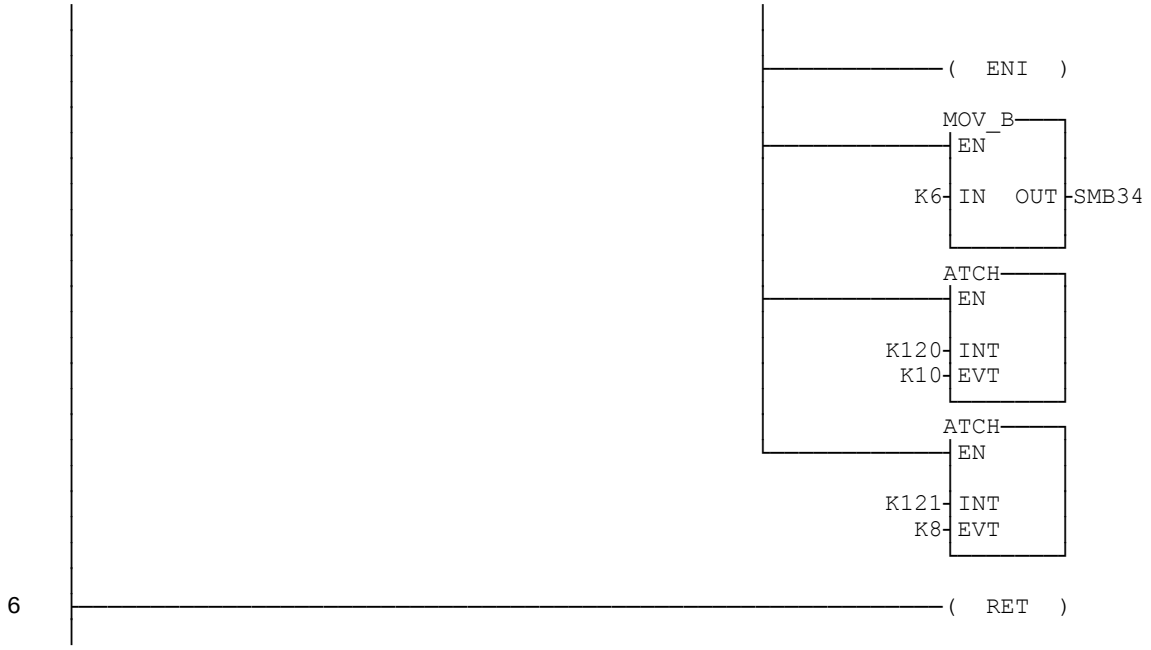
```
// Подпрограмма 50
//
// Инициализация драйвера Modbus для порта 0.
// ЗАМЕЧАНИЕ: Программа инициализации требует для исполнения около 690 милисек.
//  для инициализации таблицы CRC. Не беспокойтесь !
```




```
// Инициализация границ памяти для различных типов данных.
//
// ЗАМЕЧАНИЕ: Максимальное значение в памяти д.б. на единицу больше, чем граничное
//                    значение. Пример: Чтобы разрешить 32 выхода, используйте значение 33.

SBR       50
LD        SM0.0
MOVW      65,VW3290            // макс_регистр_отображения = 64 бита
MOVW      1001,VW3292          // макс. V слов = 2000 байт (1000 слов)
MOVW      17,VW3294            // макс. AI слов = 16 слов
MOVD      &VB2000,VD3296       // начало AI слов в V памяти
MOVW    8, VW3574              // загрузить константу = 8 для использования
                               // в математических операциях
```

```
// Инициализация адреса Modbus.

MOVB      1,VB4095             // Адрес Modbus = 1
```

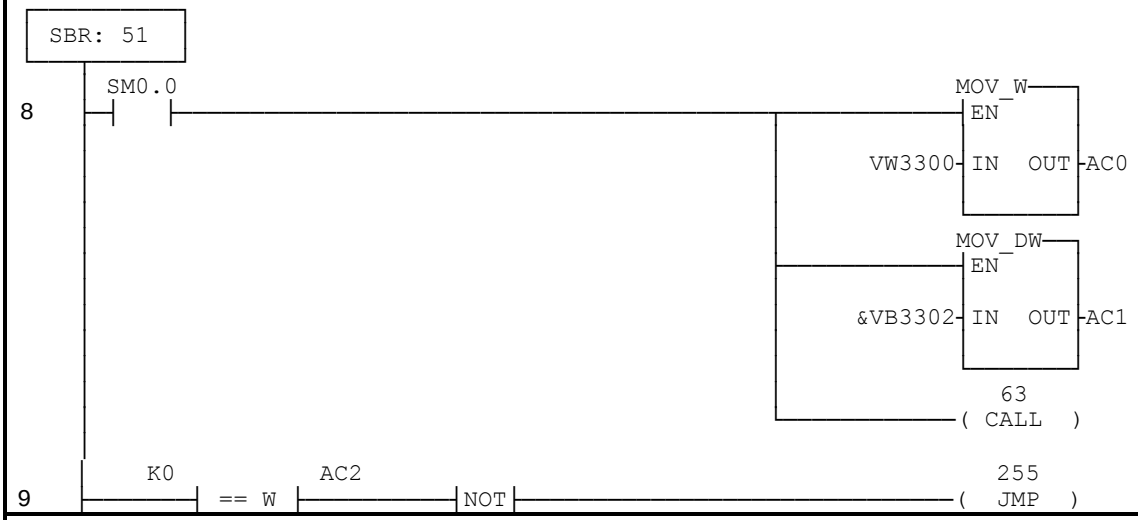
```
// Инициализация порта. См. Руководство по программированию Step-7 Micro
// для других параметров порта (например, скорости передачи, четности). RTU Modbus
// использует 8 битов данных. Скорость передачи и четность м.б. изменены.
//
// ЗАМЕЧАНИЕ: При установке 38.4K Бод функции работают неправильно.

MOVB    16#49,SMB30          // 9600 бод, 8 бит, с четностью

// Инициализация таблицы данных CRC и сброс меркера "продолжающаяся телеграмма".

CALL    62                   // Инициализация таблицы CRC Modbus
R       M31.7,1              // сброс меркера "продолжающаяся телеграмма"
```




```
// Телеграммы Modbus отделяются временем свободной линии для как минимум 3.5 байт,
// которое для 9600 Бод = 4 милисекунд. Время свободной линии устанавливается
// равным 6 милисек., чтобы гарантировать по крайней мере 5 милисек. (4 милисек. -
// время свободной линии + 1 милисек. для приема символа.
//
// ЗАМЕЧАНИЕ: Время тайм-аута д.б. изменено для разных скоростей передачи:
//
//      300 бод          166  милисек.
//      600              84
//      1200             43
//      2400             22
//      4800             12
//      9600             6
//      19.2K            5

ENI                      // разрешение прерываний
MOVB    6,SMB34          // установить таймер свободной линии > 5 милисек.
ATCH    120,10           // начальный поиск свободной линии
```

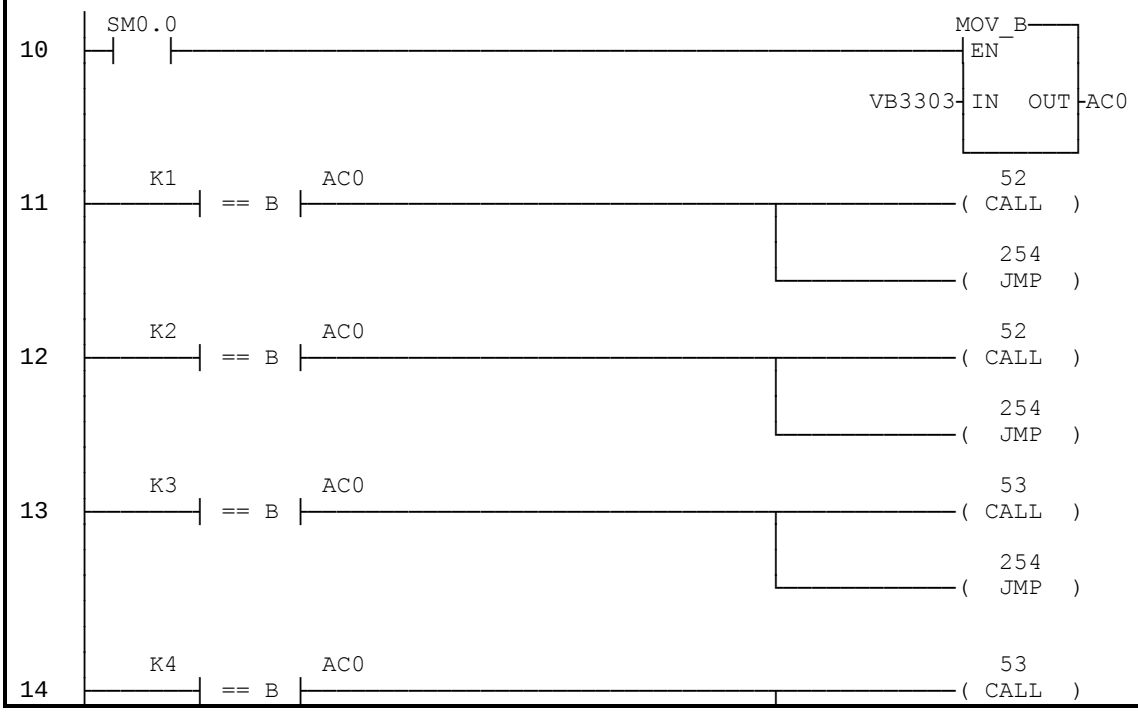
```
ATCH    121,8          // INT 121, если принят символ

RET                    // возврат
```

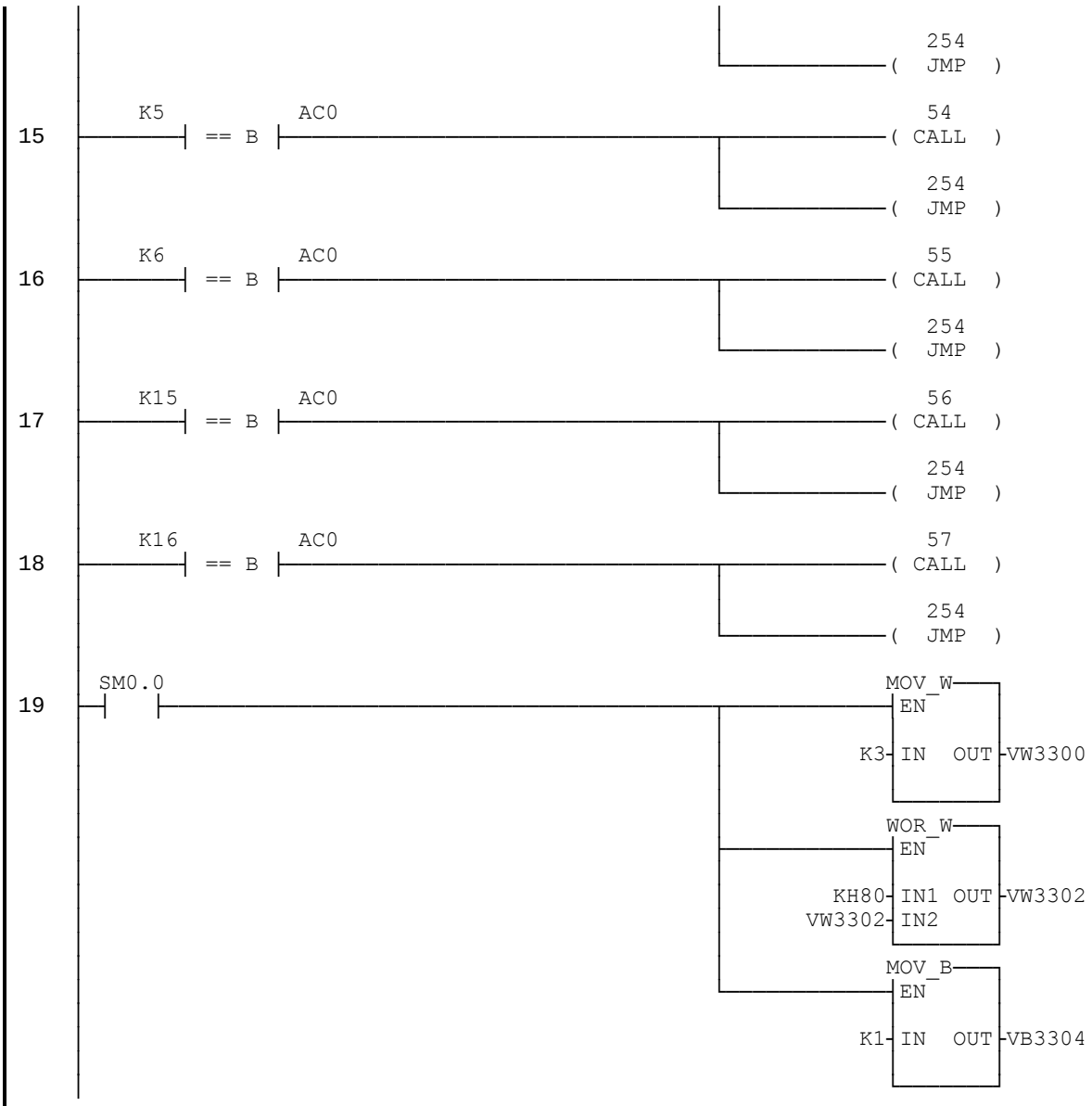
```
// Подпрограмма 51
//
// Эта подпрограмма обрабатывет запросы Modbus во время обычного цикла.
// Расчет CRC полученной телеграммы.  Если CRC, полученной телеграммы,
// включается в расчет, то результат всегда д.б. равен нулю, если
// не было ошибок.
```




```
SBR     51
LD      SM0.0
MOVW    VW3300,AC0          // получить длину буфера
MOVD    &VB3302,AC1         // получить адрес буфера для контроля CRC
CALL    63                  // расчет CRC
LDW=    0,AC2               // Если (расчитанный CRC != 0),
NOT
JMP     255                 // загрузить ошибку
```

```
// Телеграмма выглядит нормально, определяем какая функция Modbus затребована.
// Команда перехода, следующая за вызовом, будет всегда выполняться после обработки
// вызова, т.к. подпрограммы всегда устанавливают TOS в 1 перед возвратом.
```

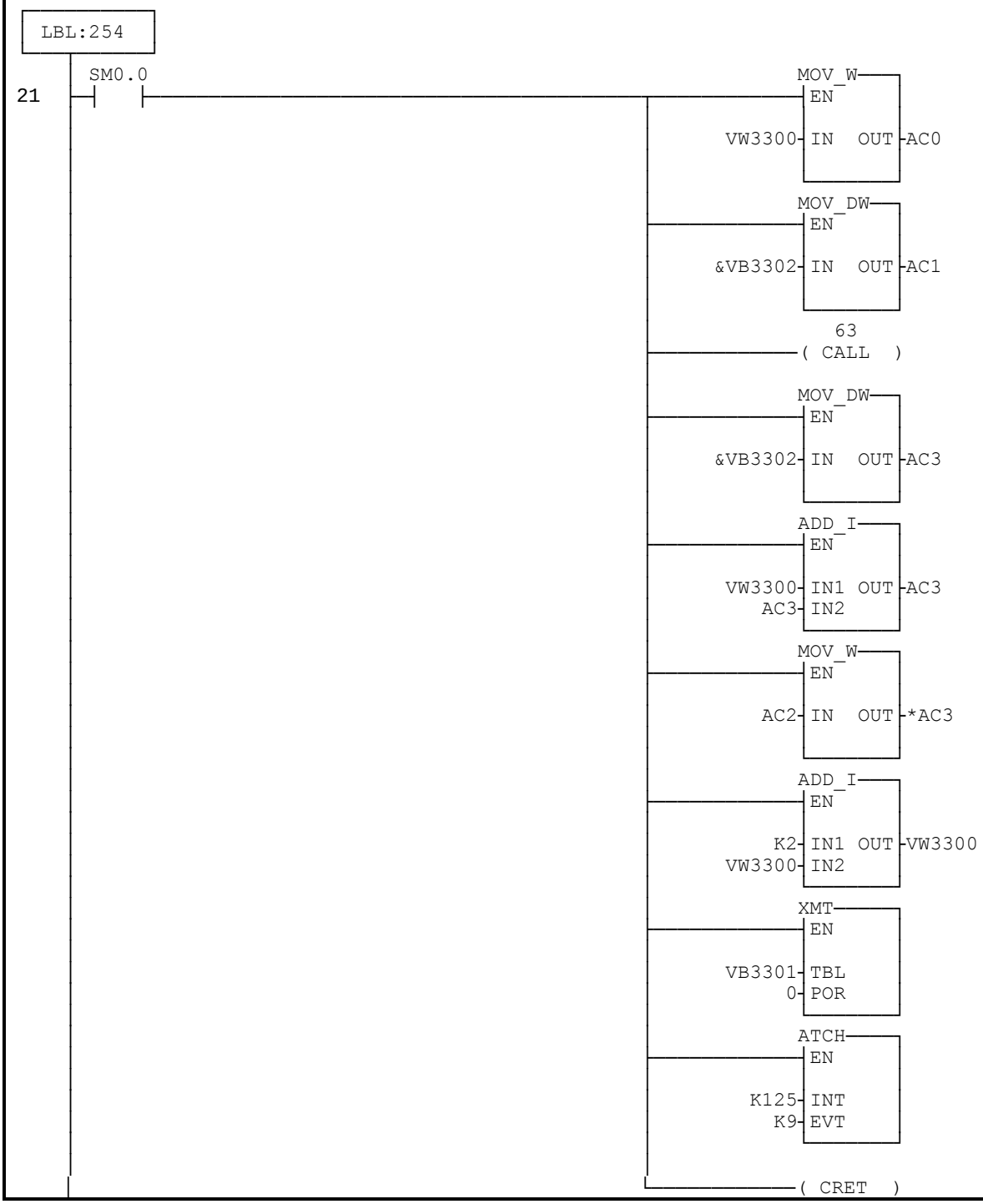





```
LD      SM0.0
MOVB    VB3303,AC0          // получить функцию из буфера приема

LDB=    1,AC0               // это функция 1?
CALL    52                  // ...если да, обработать ее
JMP     254                 //  ...затем перейти в конец

LDB=    2,AC0               // это функция 2?
CALL    52                  // ... ...если да, обработать ее
JMP     254                 //  ...затем перейти в конец

LDB=    3,AC0               // это функция 3?
CALL    53                  // ... ...если да, обработать ее
JMP     254                 //  ...затем перейти в конец

LDB=    4,AC0               // это функция 4
CALL    53                  // ... ...если да, обработать ее
JMP     254                 //  ...затем перейти в конец

LDB=    5,AC0               // это функция 5?
CALL    54                  // ... ...если да, обработать ее
JMP     254                 //  ...затем перейти в конец

LDB=    6,AC0               // это функция 6?
CALL    55                  // ... ...если да, обработать ее
JMP     254                 //  ...затем перейти в конец
```

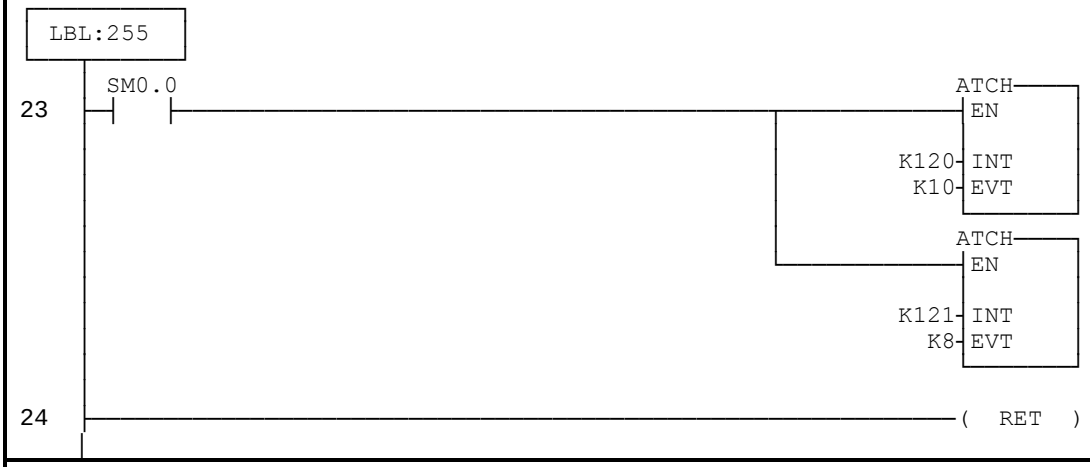
```
LDB=    15,AC0              // это функция 15?
CALL    56                  // ... ...если да, обработать ее
JMP     254                 //  ...затем перейти в конец

LDB=    16,AC0              // это функция 16?
CALL    57                  // ... ...если да, обработать ее
JMP     254                 //  ...затем перейти в конец

LD      SM0.0               // если ни одна из выше перечисленных ...
MOVW    3,VW3300            //   загрузить длину ответа об ошибке
ORW     16#0080,VW3302      //   установвить MSBit функции для показа ошибки
MOVB    1,VB3304            //   загрузить код "функция не поддерживается"
```

```
// Функция предназначена для расчета CRC ответа и  запуска передачи ответа.
// Подобно запросу длина ответа находиться в первом слове буфера.
// Эта длина не включает длину CRC и д.б. увеличена на два
// перед вызовом блока передачи.
```



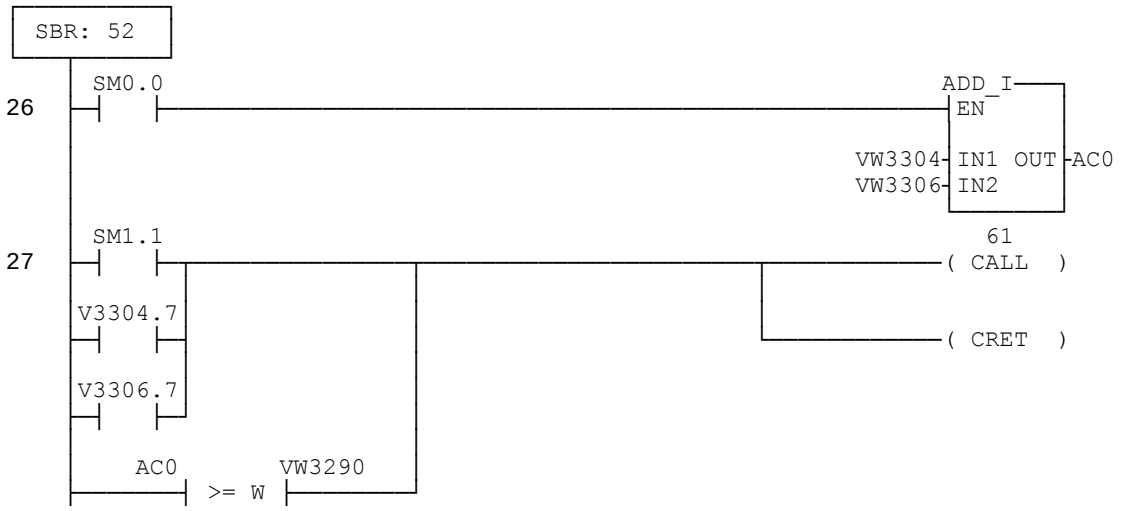
```
LBL     254
LD      SM0.0
MOVW    VW3300,AC0          // получить длину
MOVD    &VB3302,AC1         // получить адрес буфера для контроля CRC

CALL    63                  // рачет CRC

MOVD    &VB3302,AC3         // получить адрес начала буфера
+I      VW3300,AC3          // указатель на конец буфера
MOVW    AC2,*AC3            // поместить CRC в буфер
+I      2,VW3300            // добавить 2 байта к CRC

XMT     VB3301,0            // передача ответа
ATCH    125,9               // переход к INT 125, когда закончена xmit
CRET
```

```
// Обработка ошибок CRC или неверной длины.
//
// На самом деле не так много надо сделать в случае появления ошибки CRC
// или приема недостаточного для обработки количества байт, только
// сбросить драйвер для следующей телеграммы и вынудить активную
// станцию выдать тайм-аут.
```



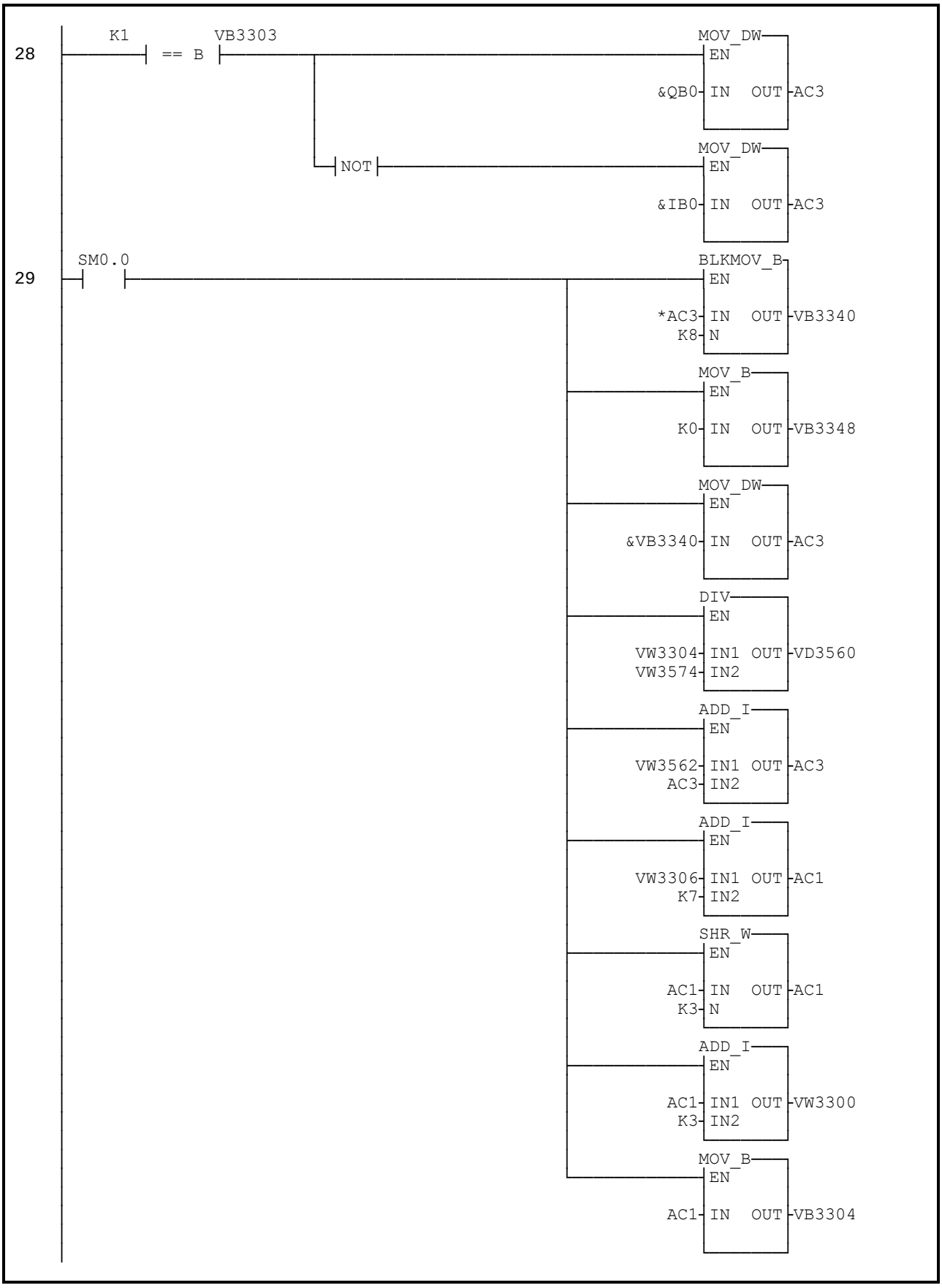

```
LBL     255
LD      SM0.0
ATCH    120,10              // запуск поиска for свободная линия
ATCH    121,8               // INT 121, если символ получен
RET
```

```
// Подпрограмма 52
//
// Эта подпраграмма поддерживает Modbus функции 1 и 2 - чтение состояния одного
// или нескольких выходов или входов. Биты в ответе собраны по восемь бит на байт.
// Первый запрошенный вход/выход находиться в младших битах (LSB) первого байта данных.
// Если число запршенных входов/выходов не делится на восемь, то старшие биты (MSB)
// последнего байта данных заполняются нулямиs, но это происходит не здесь.
//
// Формат запроса:
//   адрес   01   стартовый_бит (MSB,LSB)   количество_битов (MSB,LSB)
//
// стартовый_бит - первый запрошенный вход/выход (базис ноль).
// количество_битов - число запрошенных входов/выходов.
//
// Формат ответа:
//   адрес   01   количество_битов    данные.....
```



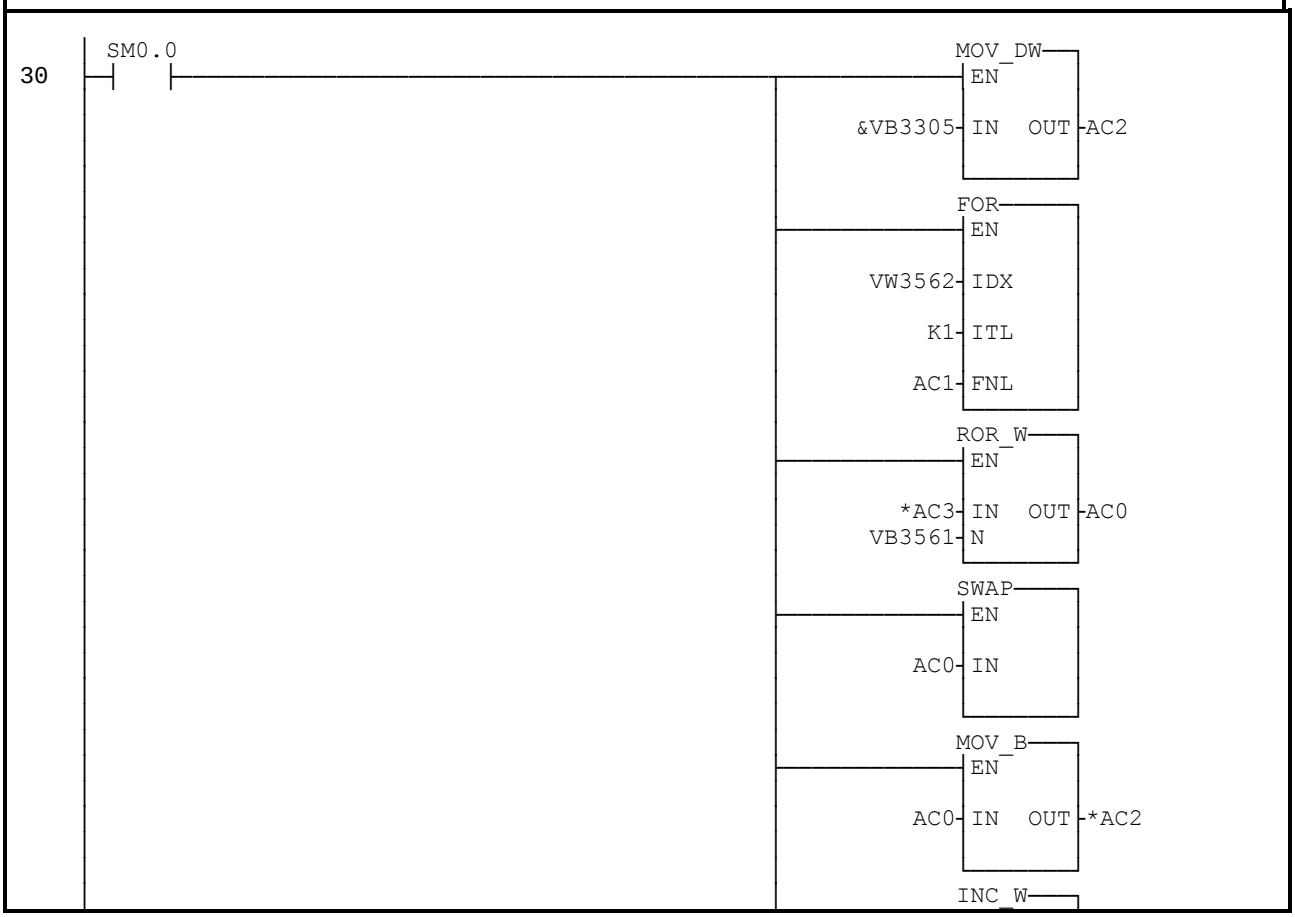

```
// Проверка, превышено ли максимальное количество входов или выходов.

SBR       52
LD        SM0.0
MOVW      VW3304,AC0          // получить значение стартового_бита
+I        VW3306,AC0          // ...и количество_битов

LD        SM1.1               // Если (переполнение) или
O         V3304.7             //   (стартовый_бит < 0) или
O         V3306.7             //   (количество_битов < 0) или
OW>=      AC0,VW3290          //   (последний_номер_бита >
                              //    макс._регистра_отображения)
CALL      61                  //   выдать ошибку
CRET                          //   возврат
```

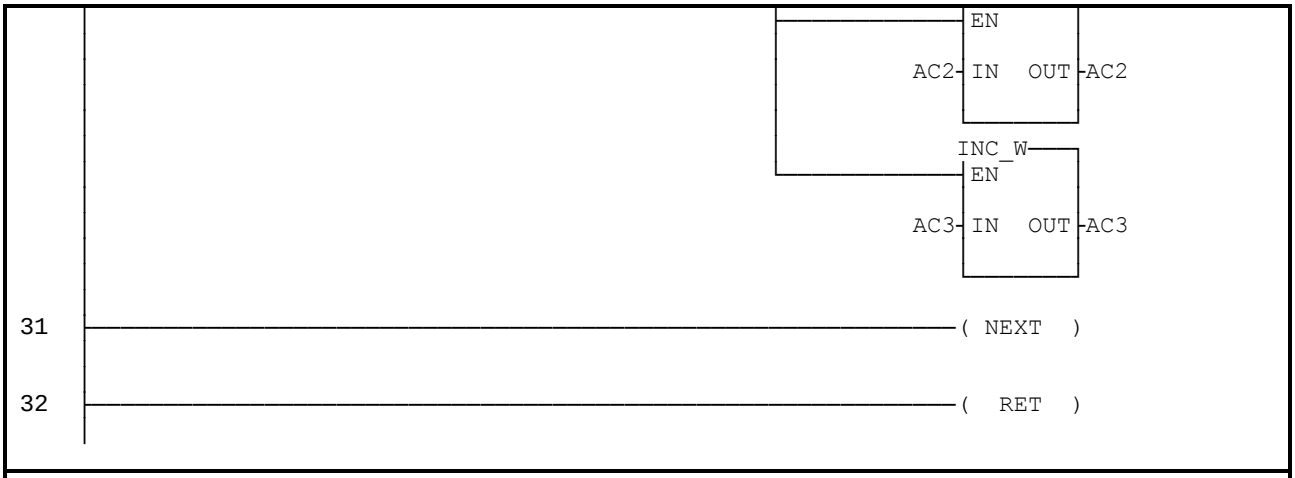
```
// Определение, с какой областью мы работаем, и затем копирование входов или
// выходов в рабочую область буфера обмена (не используемую в этом ответе),
// и установка последующих байтов в ноль. Это дает возможность правильно работать
// методу сдвига, используемому в следующей части.
//
// ЗАМЕЧАНИЕ:   Подразумевается, что размер регистра отображения - 8 байт.  Если размер
//              регистра отображения изменяется и больше 8 байт, то константа в команде
//              перемещения блока д.б. изменена.
//
// Получение номера стартового_бита и определение адреса первого байта для копирования.
// После деления значение в VW3560 - это остаток (количество сдвигов), а
// значение в VW3562 - это частное (смещение байта в входах или выходах).
```


28
K1
== B
VB3303
MOV_DW
EN
&QB0
IN
OUT
AC3
NOT
MOV_DW
EN
&IB0
IN
OUT
AC3
29
SM0.0
BLKMOV_B
EN
*AC3
IN
OUT
VB3340
K8
N
MOV_B
EN
K0
IN
OUT
VB3348
MOV_DW
EN
&VB3340
IN
OUT
AC3
DIV
EN
VW3304
IN1
OUT
VD3560
VW3574
IN2
ADD_I
EN
VW3562
IN1
OUT
AC3
AC3
IN2
ADD_I
EN
VW3306
IN1
OUT
AC1
K7
IN2
SHR_W
EN
AC1
IN
OUT
AC1
K3
N
ADD_I
EN
AC1
IN1
OUT
VW3300
K3
IN2
MOV_B
EN
AC1
IN
OUT
VB3304

```
LDB=    1,VB3303          // если функция 1 (выходы)
MOVD    &QB0,AC3          //    то указатель на выходы
NOT                       // иначе
MOVD    &IB0,AC3          //    указатель на входы

LD      SM0.0
BMB     *AC3,VB3340,8     // копирование входов/выходов в рабочую область
MOVB    0,VB3348          // обнуление "остальных" байтов в рабочей области
MOVD    &VB3340,AC3       // исходный указатель = адрес рабочей области

MOVW    VW3304,VW3562     // получить значение стартового_бита
DIV     VW3574,VD3560     // стартовый_бит / 8
+I      VW3562,AC3        // добавить смещение к исходному указателю

// Получение количества_битов и определение числа байтов, необходимых для передачи
// ответа.

MOVW    VW3306,AC1        // получить количество_битов
+I      7,AC1             // округлить
SRW     AC1,3             // разделить на 8 бит/байт

// Загрузить размер буфера ответа и количество_байт в буфер ответа.

MOVW    AC1,VW3300        // получить количество_байтов
+I      3,VW3300          // добавить три служебных байта
MOVB    AC1,VB3304        // загрузить количество_байтов
```

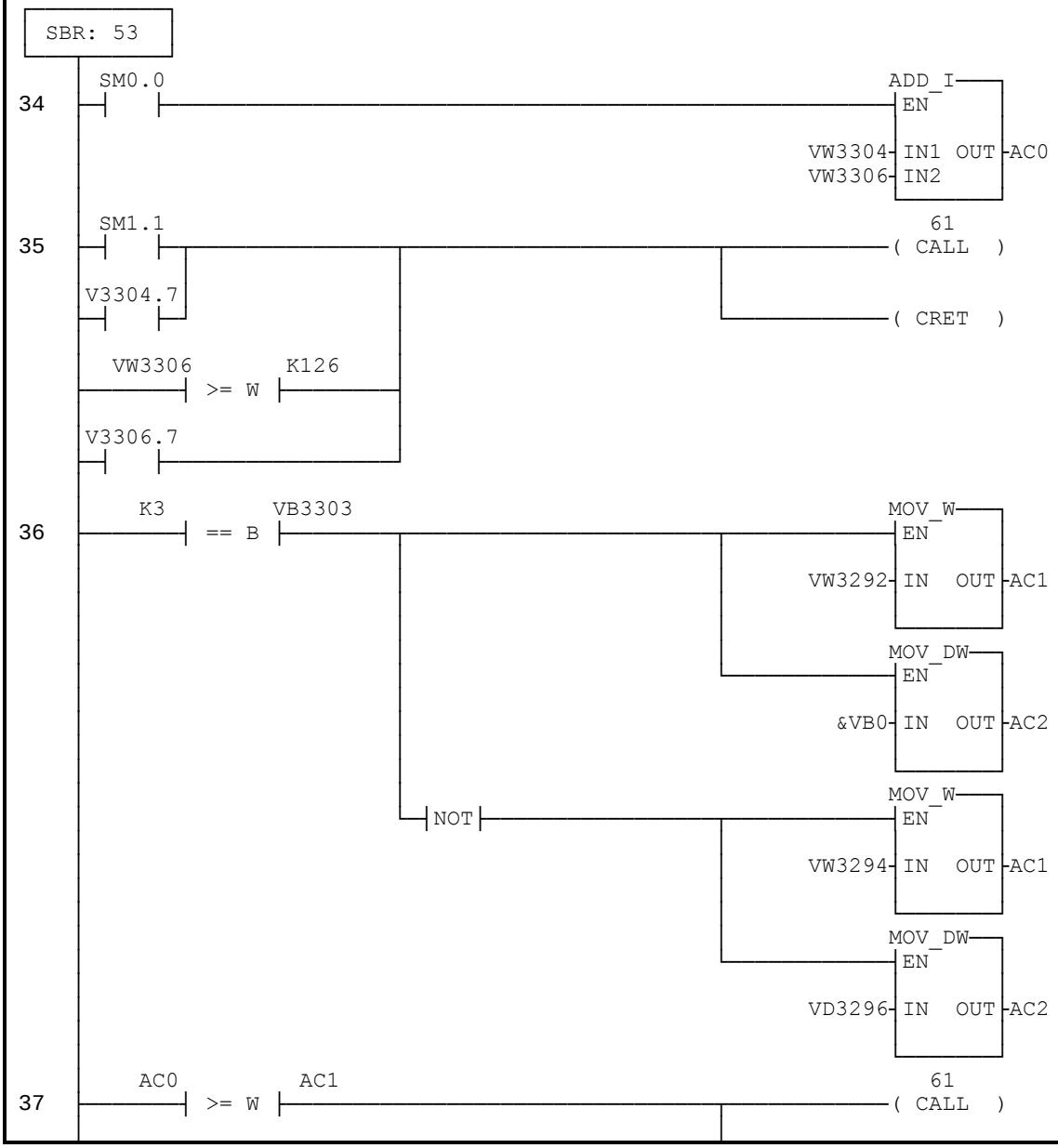
```
// Получение данных из регистра отображения, сдвиг и помещение их в буфер ответа.
//
// Используемый метод: в старший байт слова выбирается слово с необходимым байтом
// из регистра отображения. Слово циклически сдвигается на число бит.
// Таким образом, младший бит "следующего" байта регистра отображения сдвигается
// в старший бит нужного байта. Когда сдвиг закончен, то с помощью команды обмена,
// мы получаем байт в младшем байте слова и затем копируем байт в буфер ответа.
```






```
LD      SM0.0
MOVD    &VB3305,AC2          // загрузить буфер ответа

FOR     VW3562,1,AC1         // для количества_байт байты
MOVW    *AC3,AC0             //   получить слово выходов
RRW     AC0,VB3561           //   сдвиг его вправо
SWAP    AC0                  //   поместить байт в младший байт
MOVB    AC0,*AC2             //   запомнить байт
INCW    AC2                  //   инкрементировать указатель буфера
INCW    AC3                  //   инкрементировать указатель регистра отображения
NEXT                         // конец для
RET
```

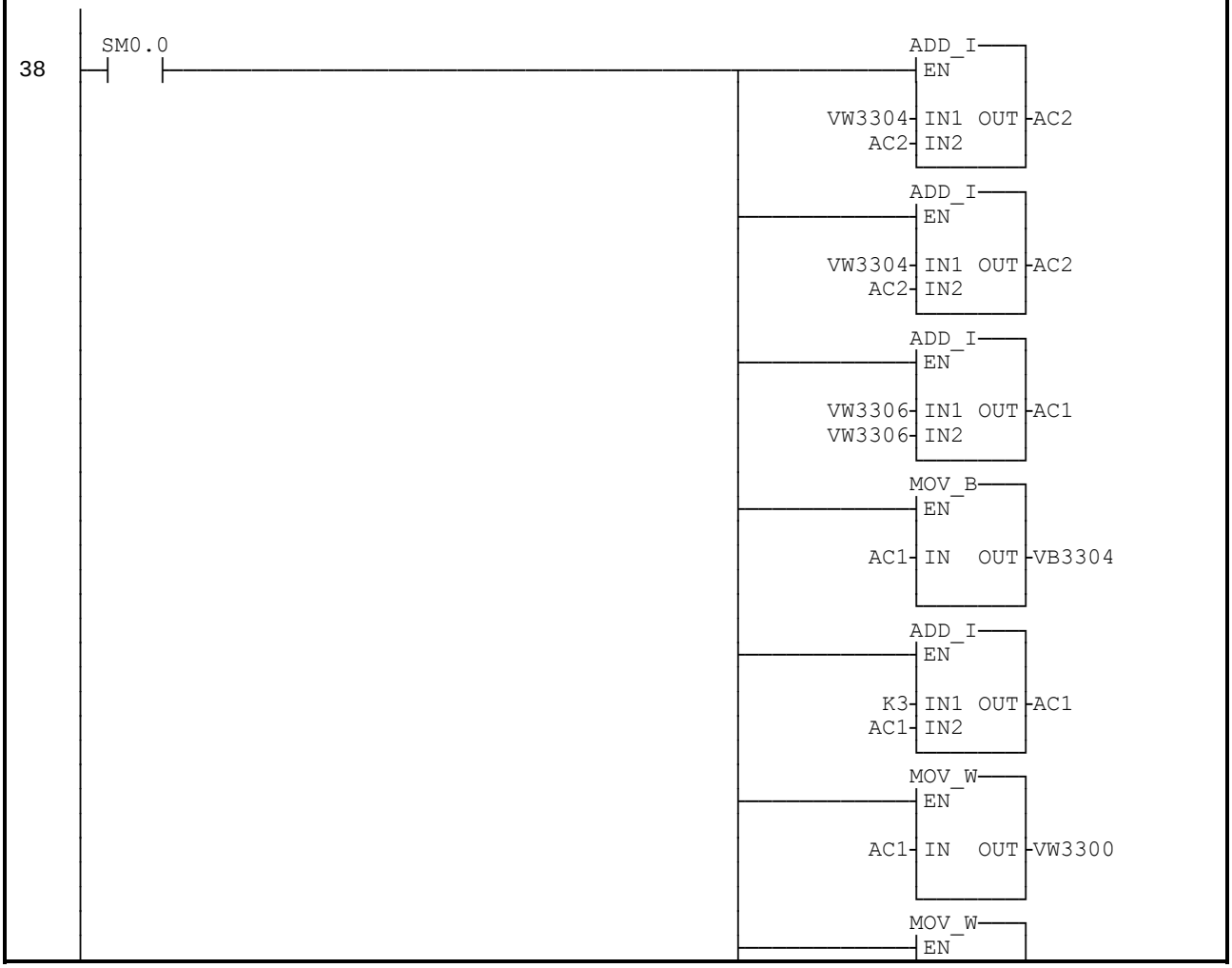
```
// Подпрограмма 53
//
// Эта подпрограмма поддерживает Modbus функцию 3 (чтение регистров выходов /
// управляющих регистров)и функцию 4 (чтение регистров входов). Подразумевается, что
// регистры выходов / управляющие регистры находяться в V памяти в ПЛК.
// Регистры входов - аналоговые входы.
//
// ЗАМЕЧАНИЕ: аналоговые входы не могут напрямую быть прочитаны в данной подпрограмме,
//            т.к. ПЛК не разрешает косвенный доступ к аналоговым значениям. Аналоговые
//            значения читаются из области V памяти, заданной в VD3296. Эта область д.б.
//            задана в подпрограмме инициализации. Копирование аналоговых данных в
//            V память - это обязанность пользователя.
//
// Формат для запроса:
//
//   адрес   03   начальное_слово (MSB,LSB) количество_слов(MSB,LSB)
//
//  начальное_слово - первое запршенное слово (базис ноль).
//  количество_слов - число запрошенных слов.
//
// Формат для ответа:
//
//   адрес   03   количество_байт   данные.....
```

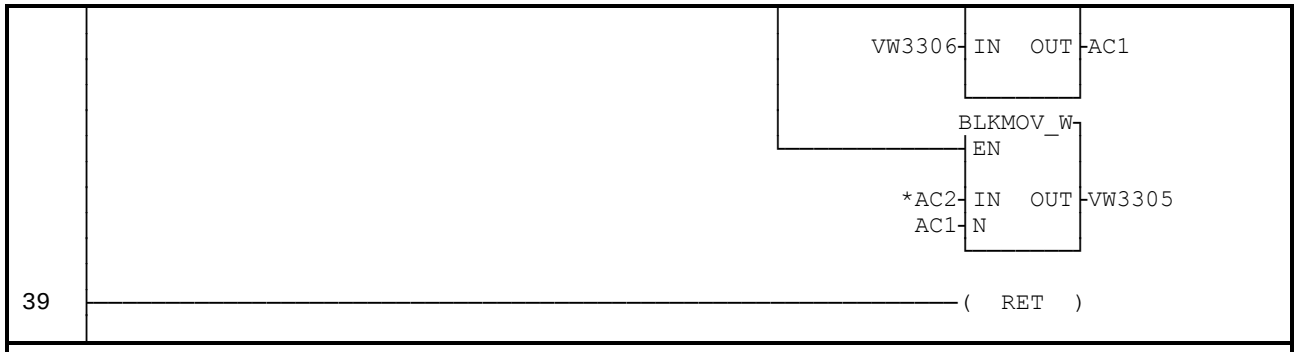


```
                                                    ( CRET  )
```

```
// Проверка, превышена ли макс. граница памяти.

SBR     53
LD      SM0.0
MOVW    VW3304,AC0          // получить значение начального_слова
+I      VW3306,AC0          // ...плюс   количество_слов

LD      SM1.1               // если (переполнение) или
O       V3304.7             //   (начальное_слово < 0) или
OW>=    VW3306,126          //   (количество_слов > 125) или
O       V3306.7             //   (количество_слов < 0)
CALL    61                  //   выдать ошибку отклика
CRET                        //   возврат

LDB=    3,VB3303            // если функция 3 (V память)
MOVW    VW3292,AC1          //   получить макс. размер V памяти
MOVD    &VB0,AC2            //   исходный_указатель = V память
NOT                         // иначе
MOVW    VW3294,AC1          //   получить макс. размер слов AI
MOVD    VD3296,AC2          // исходный_указатель = AI память

LDW>=   AC0,AC1             // если начальное_слово + количество_слов >=
                            // размера памяти
CALL    61                  //   выдать ошибку
CRET                        //   возврат
```

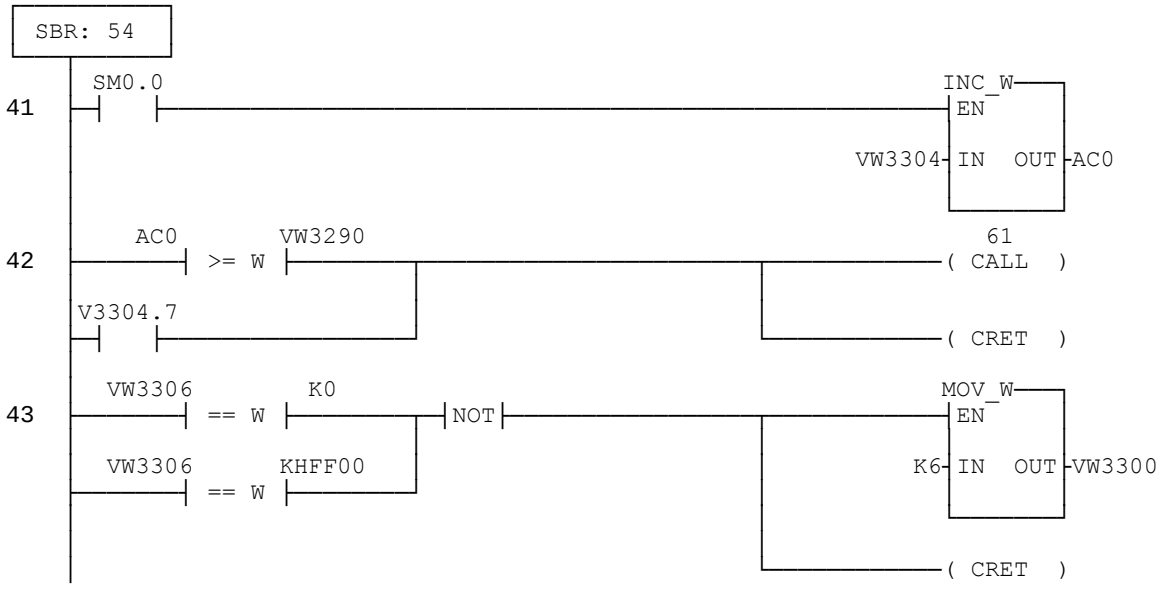
```
// Запрос - OK; получить начальное_слово, удвоить его для смещения байта и
// добавить его к исходному_указателю. Указатель на буфер ответа и
// скопировать данные сюда.
```





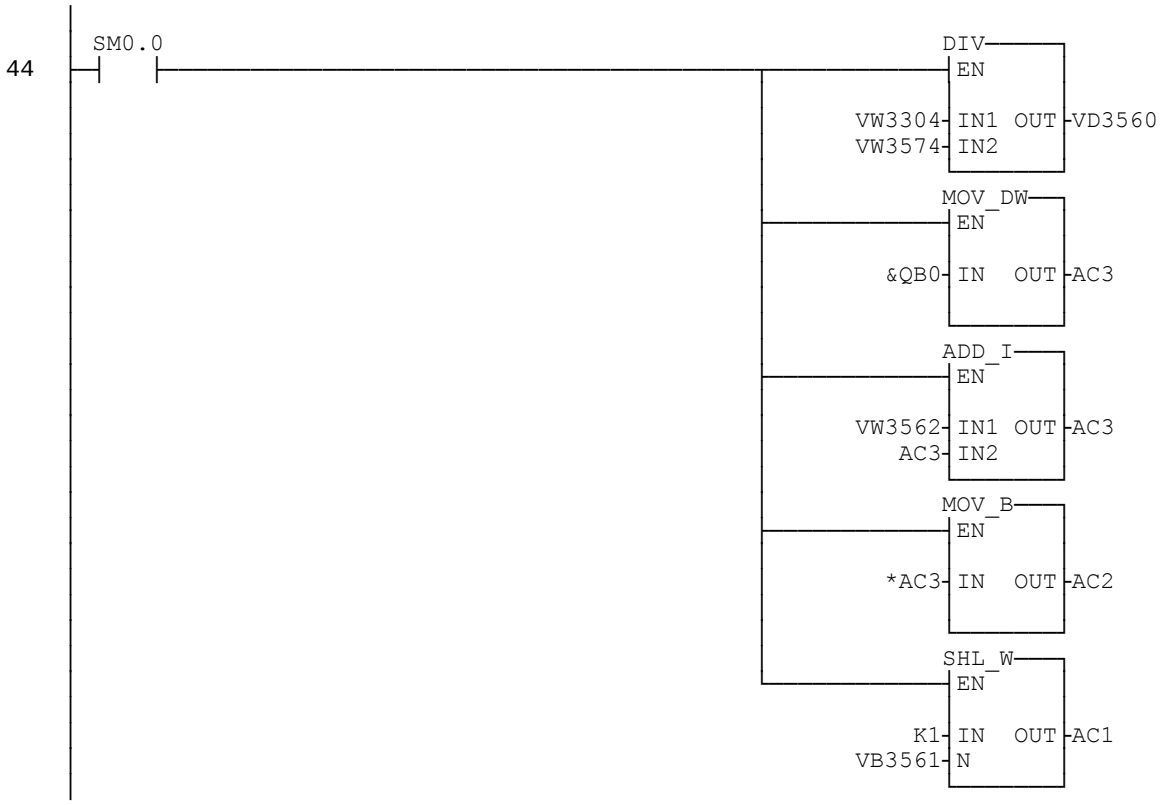

```
LD      SM0.0
+I      VW3304,AC2          // добавить начальное_слово к исходному_указателю
+I      VW3304,AC2          //   ...начальное_слово * 2 к исходному_указателю

MOVW    VW3306,AC1          // получить количество слов
+I      VW3306,AC1          // ...умножить на два для количества байт
MOVB    AC1,VB3304          //   ...запомнить количество байт ответа
+I      3,AC1               // добавить служебные байты
MOVW    AC1,VW3300          //   ...запомнить размер буфера ответа

MOVW    VW3306,AC1          // получить опять количество слов
BMW     *AC2,VW3305,AC1     // скопировать данные из источника в буфер
RET                         // возврат
```

```
// Подпрограмма 54
//
// Даная подпрограмма поддерживает Modbus функцию 5 для управления отдельным выходом: // включения
или выключения.
//
// Формат запроса:
//
//   адрес   05   бит_входов (MSB,LSB)   данные (FF00 или 0000)
//
// Значение данных FF00 включает вход, значение 0000 выключает вход.
// Любое другое значение данных не приводит ни к какому действию.
//
// Ответная телеграмма повторяет телеграмму запроса.
```

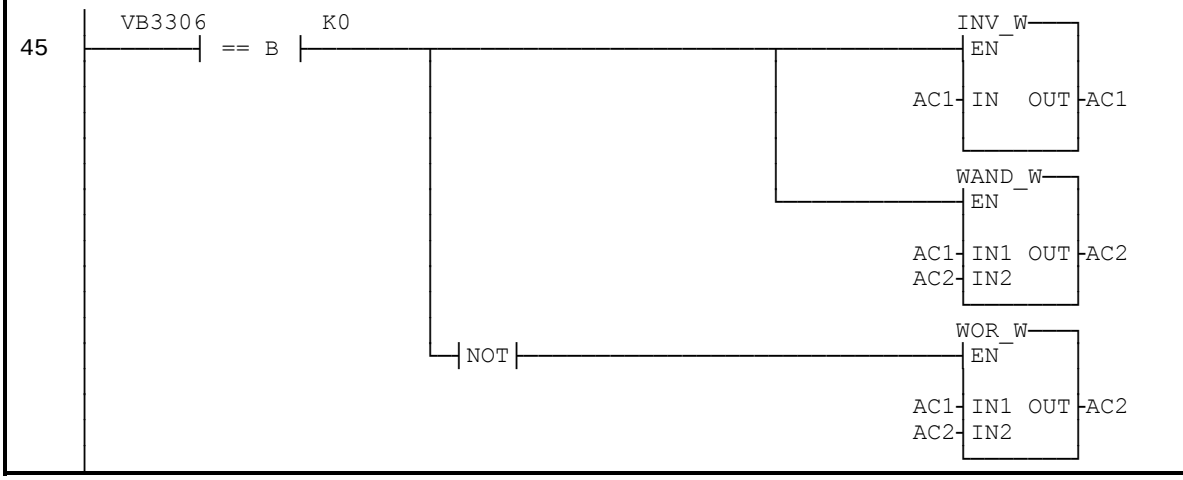



```
SBR       54
LD        SM0.0
MOVW      VW3304,AC0          // получить бит_выходов
INCW      AC0                 // ...и запомнить его для проверки

LDW>=   AC0,VW3290            // если (бит_выходов >
                              // макс_число_регистра_отображения) или
O         V3304.7             //   (бит_выходов < 0)
CALL      61                  //   выдать ошибку
CRET                          //   возврат

// Проверка, равны ли данные FF00 или 0000.  Если данные не совпадают ни с одним
// из этих значений, то не предпринимаются никакие действия, за исключением
// повторения запроса к активной станции.

LDW=      VW3306,0            // если (данные != 0) или
OW=       VW3306,16#FF00      //   (данные != FF00)
NOT
MOVW      6,VW3300            //   установить длину ответа
CRET                          //   возврат
```

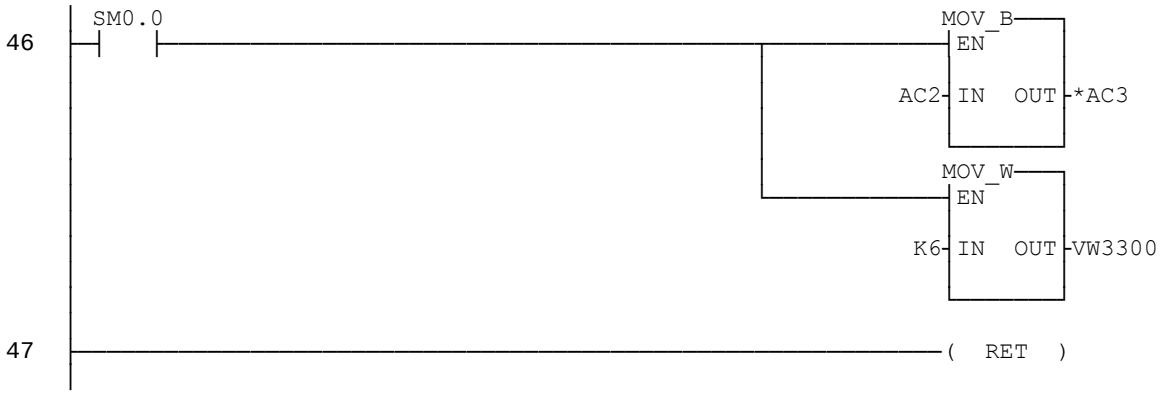
// Определение какой байт выходов и какой бит байта изменились.
// Определяется при помощи деления бита выходов на восемь. Частное - байт
// смещения, а остаток - номер бита.
//
// После деления, значение в VW3560 - остаток (номер бита), а
// значение в VW3562 - частное (смещение в выходах).




```
LD      SM0.0
MOVW    VW3304,VW3562        // загрузить бит_выходов
DIV     VW3574,VD3560        // бит_выходов / 8

MOVD    &QB0,AC3             // указатель на выходы
+I      VW3562,AC3           // сместить указатель на соответствующий байт
MOVB    *AC3,AC2             // получить байт выходов

MOVW    1,AC1                // создать маску
SLW     AC1,VB3561           // ...для соответствующего бита
```

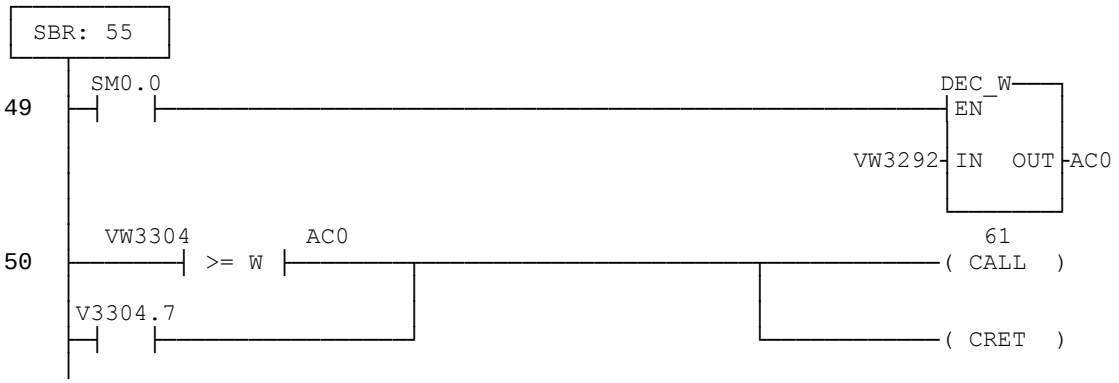


```
LDB=    VB3306,0        // если данные == 0
INVW    AC1             //   образовать маску очистки
ANDW    AC1,AC2         //   очистить бит
NOT                     // иначе
ORW     AC1,AC2         //   установить бит
```



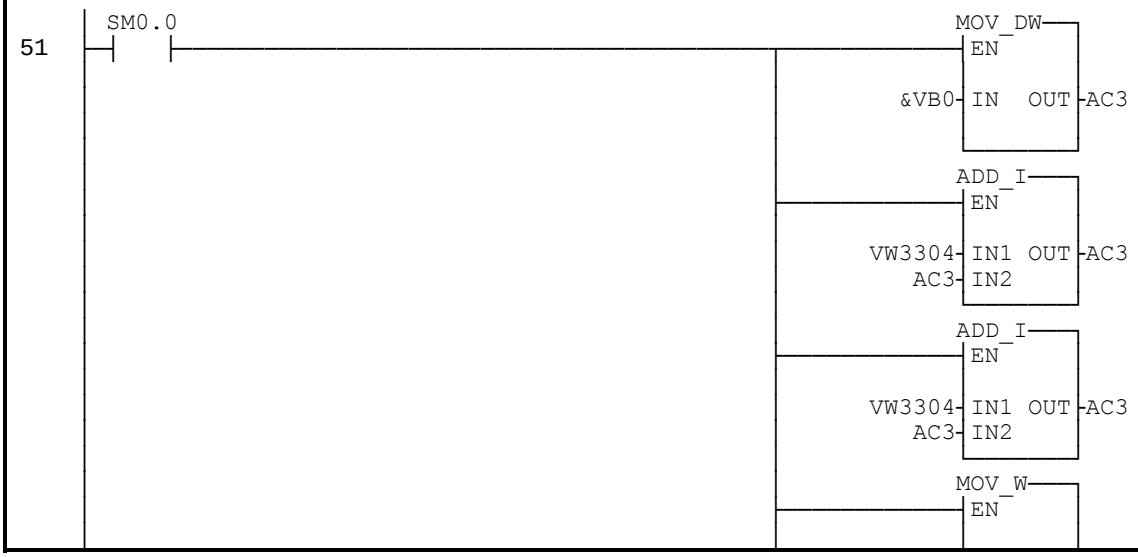

```
LD      SM0.0
MOVB    AC2,*AC3        // восстановить байт выхода
MOVW    6,VW3300        // установить длину ответа
RET
```

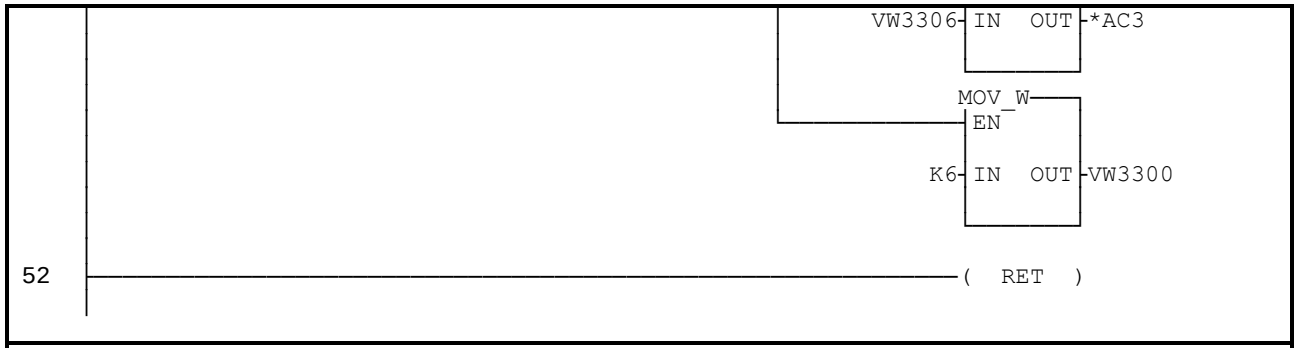
```
// Подпрограмма 55
//
// Данная подпрограмма поддерживает Modbus функцию 6 для записи одного управляющего
// регистра в ПЛК.  Управляющие регистры для данной реализации подразумеваются в
// V памяти.
//
// Формат запроса:
//
//   адрес   06   начальное_слово (MSB,LSB)   данные (MSB,LSB)
//
//  начальное_слово - первое запрошенное слово (базис ноль).
//  данные - слово, записываемое в ПЛК.
//
// Формат ответа такой же, как запроса:
//
//   адрес   06   начальное_слово (MSB,LSB)   данные (MSB,LSB)
```




```
// Проверка, не превышен ли макс. предел памяти.

SBR       55
LD        SM0.0
MOVW      VW3292,AC0          // получить значение размера памяти + 1
DECW      AC0                 // ...и получить из него размер памяти

LDW>=  VW3304,AC0             // если (начальное_слово >= размер памяти) или
O         V3304.7             //   (начальное_слово < 0)
CALL      61                  //   выдать ошибку
CRET                          //   возврат
```

```
// Запрос - OK; получить начальное_слово, удвоить его для смещения байта, и
// добавить его к исходному_указателю. Скопировать данные запроса в V память.
```

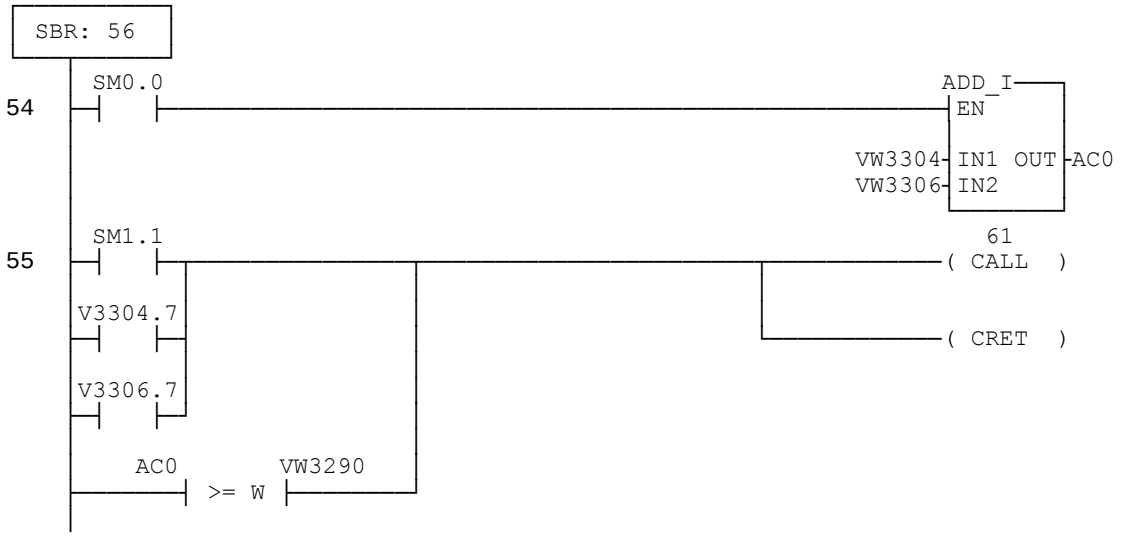





```
LD      SM0.0
MOVD    &VB0,AC3        // исходный_указатель = V память
+I      VW3304,AC3      // добавить смещение...
+I      VW3304,AC3      // ...дважды для смещения байта
MOVW    VW3306,*AC3     // запомнить данные в V память
MOVW    6,VW3300        // установить длину ответа
RET                     // возврат
```

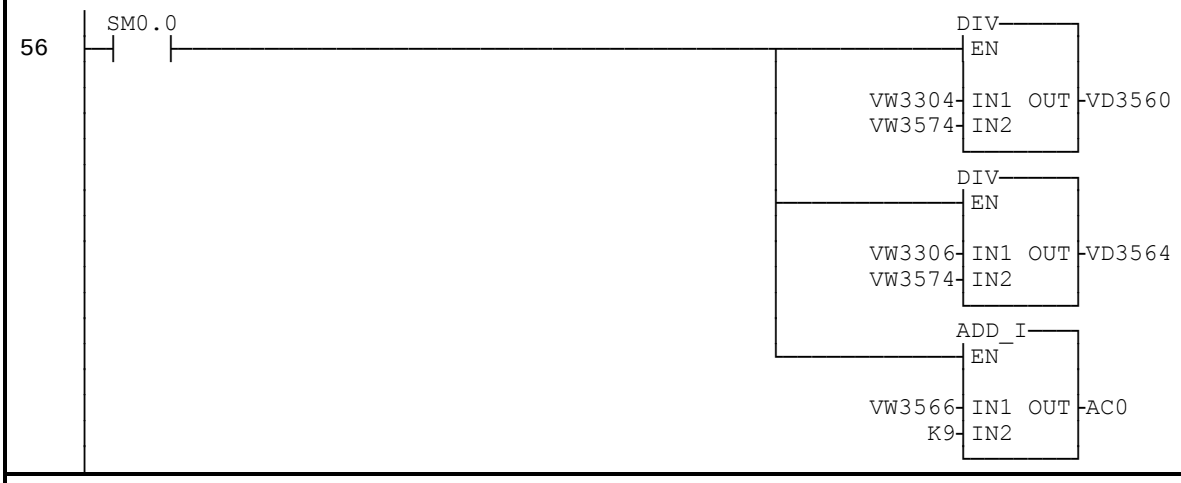
```
// Подпрограмма 56
//
// Данная подпрограмма поддерживает Modbus функцию 15 (управление несколькими
// выходами). Выхода имеют восемь бит на байт. LSБит первого байта данных
// записывается в ячейку начального_бита. Последующие биты записываются
// в следующие ячейки.
//
```

```
// ЗАМЕЧАНИЕ: В этой реализации, если начальный_бит и значение количества_бит
//            не кратны восьми, возвращается ошибка.  В STL или LAD очень трудно
//            обрабатывать начало любого бита.
//
//   Формат запроса:
//
//   адрес  0F  начальный_бит(MSB,LSB)  количество_бит(MSB,LSB)  количество_байт  данные...
//
//   начальный_бит - первый записываемый выход и LSБит первого байта данных.
//   количество_бит - число записываемых выходов.
//   количество_бит - число байтов данных.
//
//   Формат ответа:
//
//   адрес  0F  начальный_бит(MSB,LSB)  количество_бит(MSB,LSB)
```



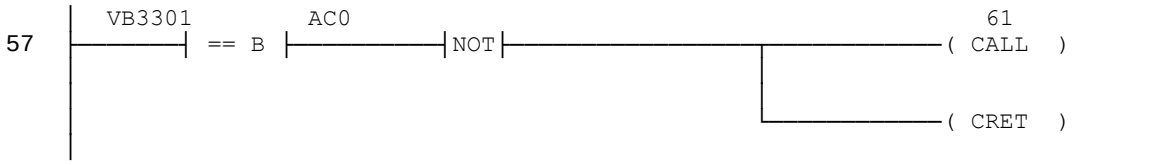

```
// Проверка, превышено ли макс. число выходов.

SBR       56
LD        SM0.0
MOVW      VW3304,AC0          // получить значение начального_бита
+I        VW3306,AC0          // ...плюс количество_бит

LD        SM1.1               // если (переполнение) или
O         V3304.7             //   (начальный_бит < 0) или
O         V3306.7             //   (количество_бит < 0) или
OW>=      AC0,VW3290          //   (последний_адрес > макс_регистра_отображения)
CALL      61                  //   выдать ошибку
CRET                          //   возврат
```

// Определение смещения байта и бита в регистре отображения выхода, и полная или
// частичная запись байтов.
//
// После деления, значение в VW3560 - остаток (количество сдвигов), а
// значение в VW3562 - частное (байт смещения в выходах).
//
// После деления, значение в VW3564 - остаток, а значение в
// VW3566 - число полных записанных байт.



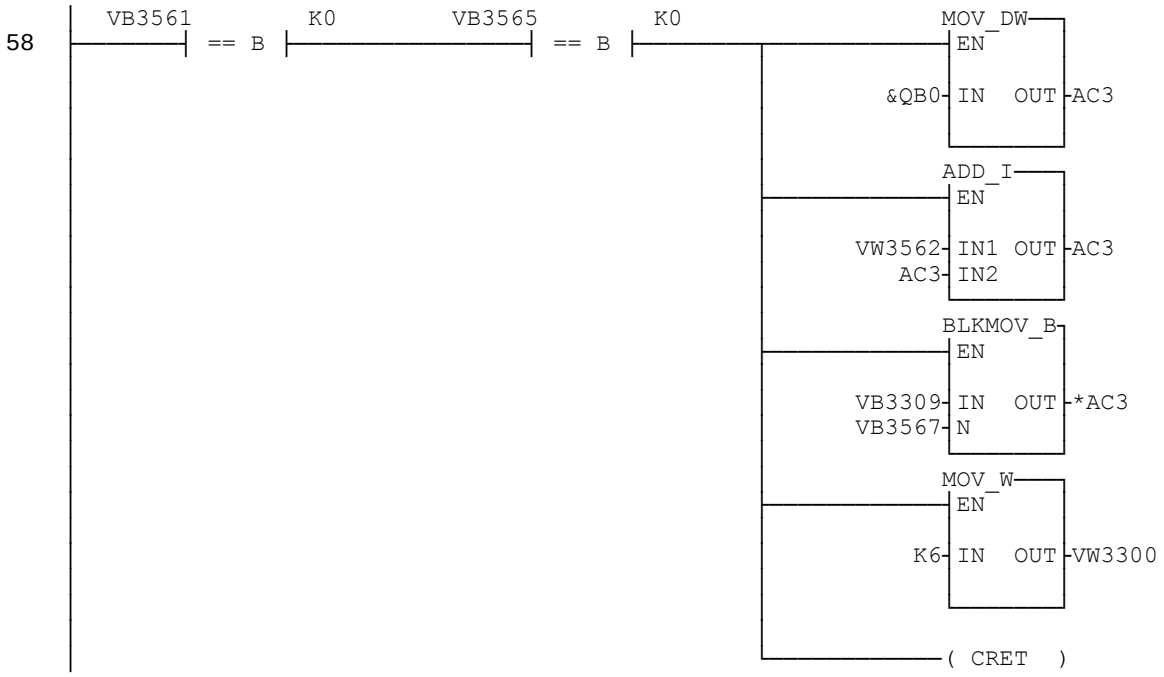

```
LD      SM0.0
MOVW    VW3304,VW3562        // получить значение начального_бита
DIV     VW3574,VD3560        // начальный_бит / 8

MOVW    VW3306,VW3566        // получить количество_бит
DIV     VW3574,VD3564        // количество_бит / 8
MOVW    VW3566,AC0           // получить (количество_бит / 8)
+I      9,AC0                // ...плюс 9 служебных байт
```

// Будте уверены, что главная ЭВМ посылает достаточно данных для запрос.
// Размер буфера д.б. равен (количеству_бит / 8) + 9 служебных байт.



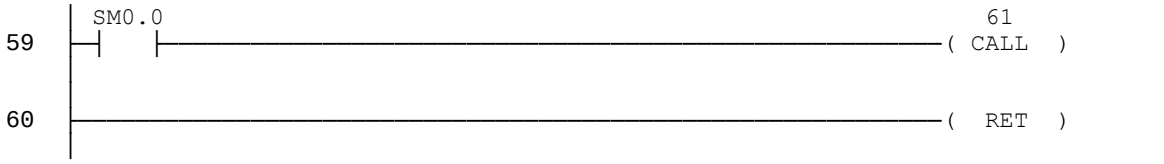

```
LDB=    VB3301,AC0           // если (расчитанная длина ( полученной длине)
NOT
CALL    61                   //  выдать ошибку
CRET                         //  возврат
```

// Определение работаем ли мы с "удобным" кадром, где начальный_бит это
// первый бит байта и количество_бит - целое число байт.
// Если это не так, то выдать ошибку главной ЭВМ.



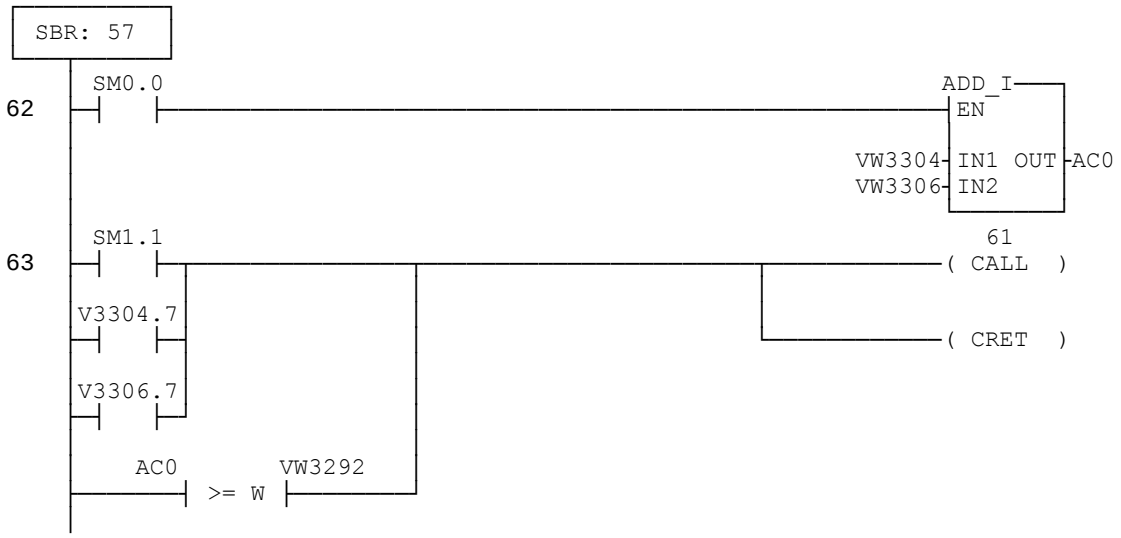
LDB=	**VB3561,0**	**// если (начальное_бит - нулевой бит байта) и**
AB=	**VB3565,0**	**// (целое число байт = TRUE)**
MOVD	**&QB0,AC3**	**// указатель на выходы**
+I	**VW3562,AC3**	**// добавить смещение к начальному выходу**
BMB	**VB3309,*AC3,VB3567**	**// скопировать запрашиваемые данные на выхода**
MOVW	**6,VW3300**	**// установить размер ответа**
CRET		**// возврат**

// Это не удобный кадр; выход с прерыванием запроса.



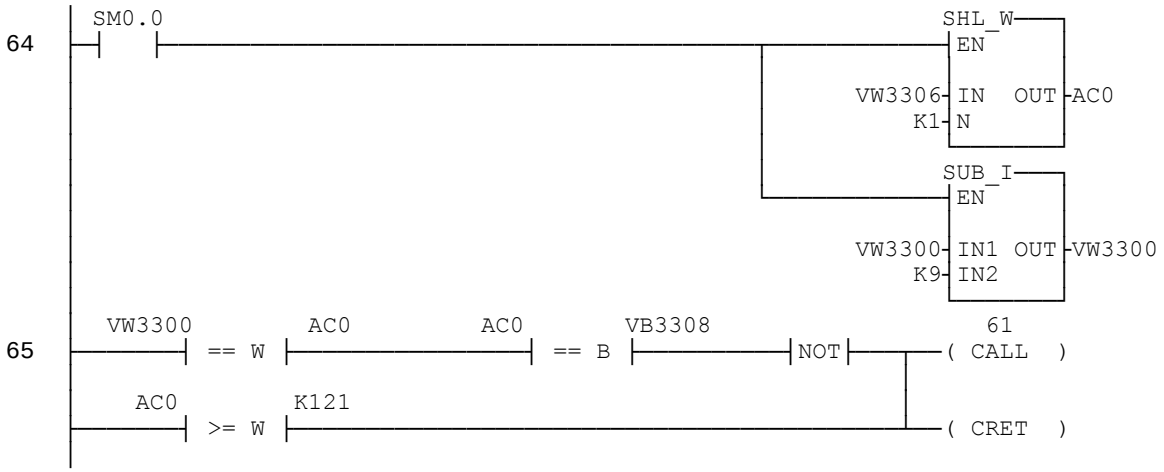
LD	**SM0.0**	**// иначе**
CALL	**61**	**// выдать ошибку**
RET		**// возврат**

```
// Подпрограмма 57
//
// Данная подпрограмма поддерживает Modbus функцию 16 (запись до 60 управляющих
// регистров в ПЛК). В данной реализации подразумевается, что управляющие регистры
// находяться в V памяти.
//
// ЗАМЕЧАНИЕ: макс. значение 60 регистров (120 байт) - ограничение записанное в
//                        спецификации Modbus.
//
//   Формат запроса:
//    адрес  10  начальное_слово(MSB,LSB) количество_слов(MSB,LSB) количество_байт  данные...
//
//   Формат ответа такой же, как запроса:
//   адрес  10  начальное_слово(MSB,LSB) количество_слов(MSB,LSB)
```



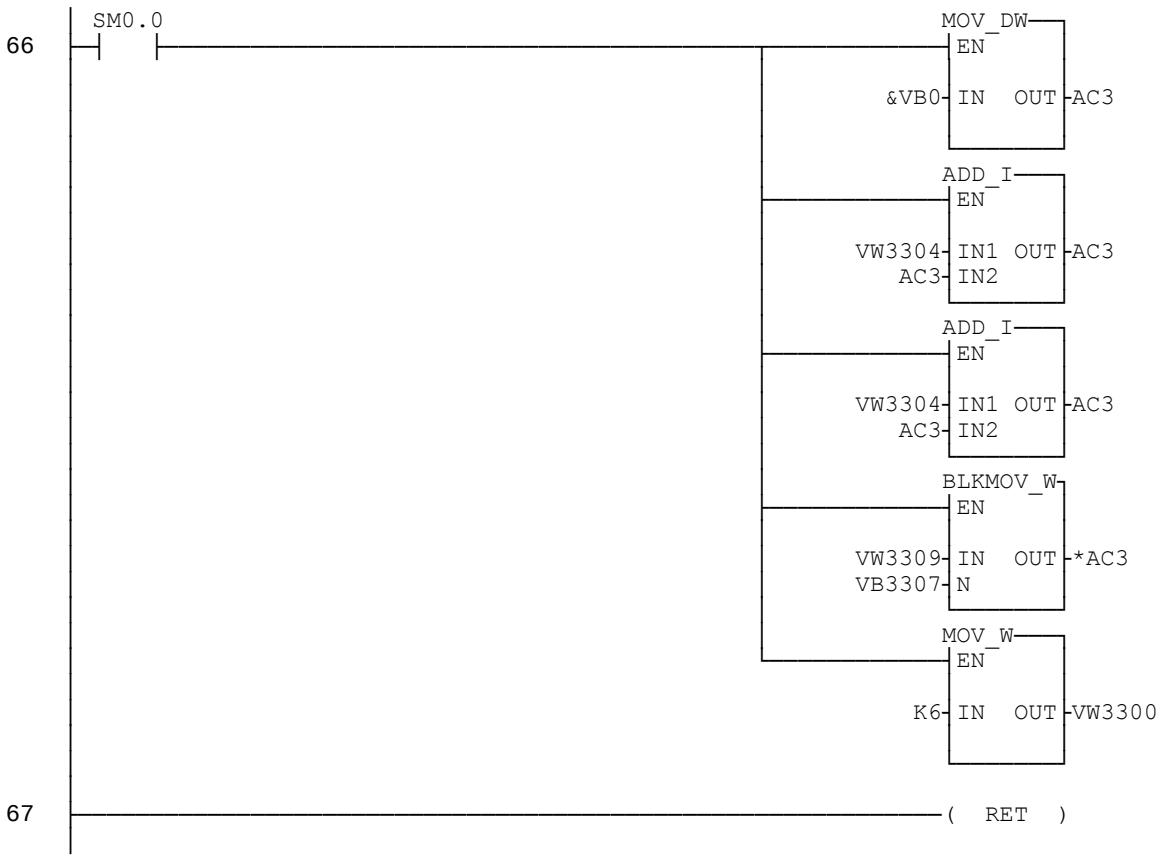

```
// Расчет адреса последнего запршенного слова путем добавления количества_слов к
// начальному_слову. Если этот последний адрес больше, чем макс. адрес доступной V памяти,
// выдать ошибку.

SBR       57
LD        SM0.0
MOVW      VW3304,AC0          // получить начальное_слово
+I        VW3306,AC0          // ...плюс количество_слов

LD        SM1.1               // если (переполнение) или
O         V3304.7             //   (начальное_слово < 0) или
O         V3306.7             //   (количество_слов < 0) или
OW>=      AC0,VW3292          //   (последний адрес >= размер памяти)
CALL      61                  //   выдать ошибку
CRET                          //   возврат
```

// Соравнение количества принятых байт с количеством_слов телеграммы, чтобы
// быть уверенным, что мы получили достаточно байт данных. Принятое количество_байт
// д.б. равно (количество_слов * 2) + 9 служебных байт. Так же проверка, что
// количество слово не больше 60 (120 байт), макс. разрешенное для функции.
//
// Так же проверка, что количество_слов * 2 равно количеству_байт. Двойная проверка всех
// этих полей - это небольшая избыточность, но мы должны выполнить все, что положено.



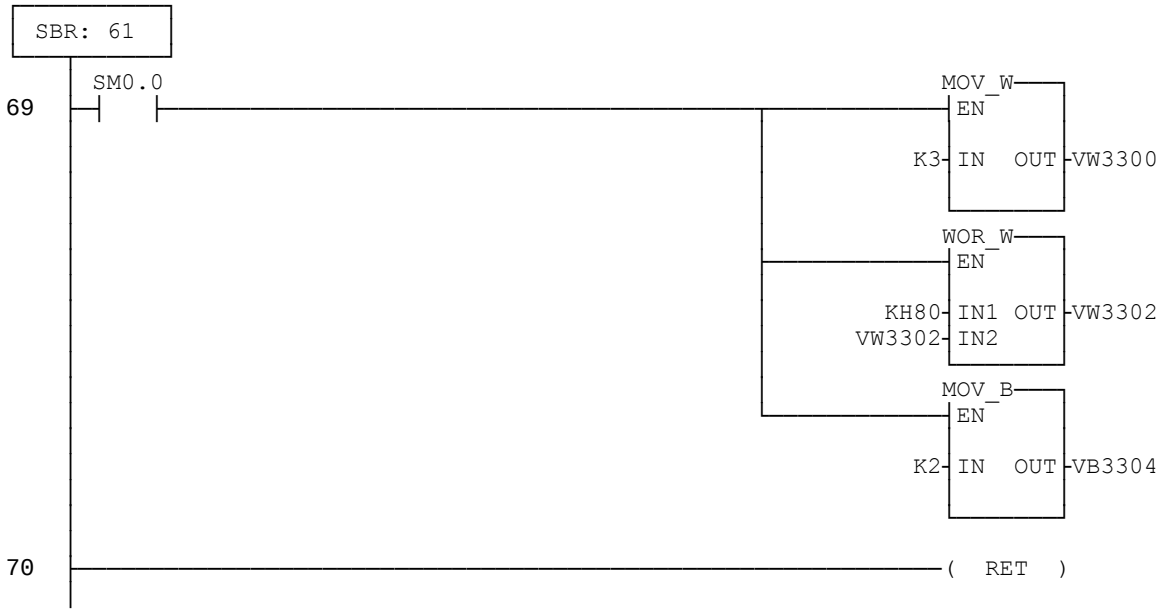

```
LD          SM0.0
MOVW        VW3306,AC0          // получить количество_слов
SLW         AC0,1               // количество_слов * 2
-I          9,VW3300            // количество_байт - 9

LDW=        VW3300,AC0          // если (количество_слов * 2 !=
                                // принятому количеству - 9) или
AB=         AC0,VB3308          //   (количество_слов * 2 != количество_байт) или
NOT
OW>=   AC0,121                  //   (количество > 60 слов + служебные)
CALL        61                  //   выдать ошибку
CRET                            //   возврат
```

// Запрос - OK; получить начальное_слово, удвоить его для смещения байта, а затем
// добавить его к исходному_указателю. Скопировать данные запрос в V память.



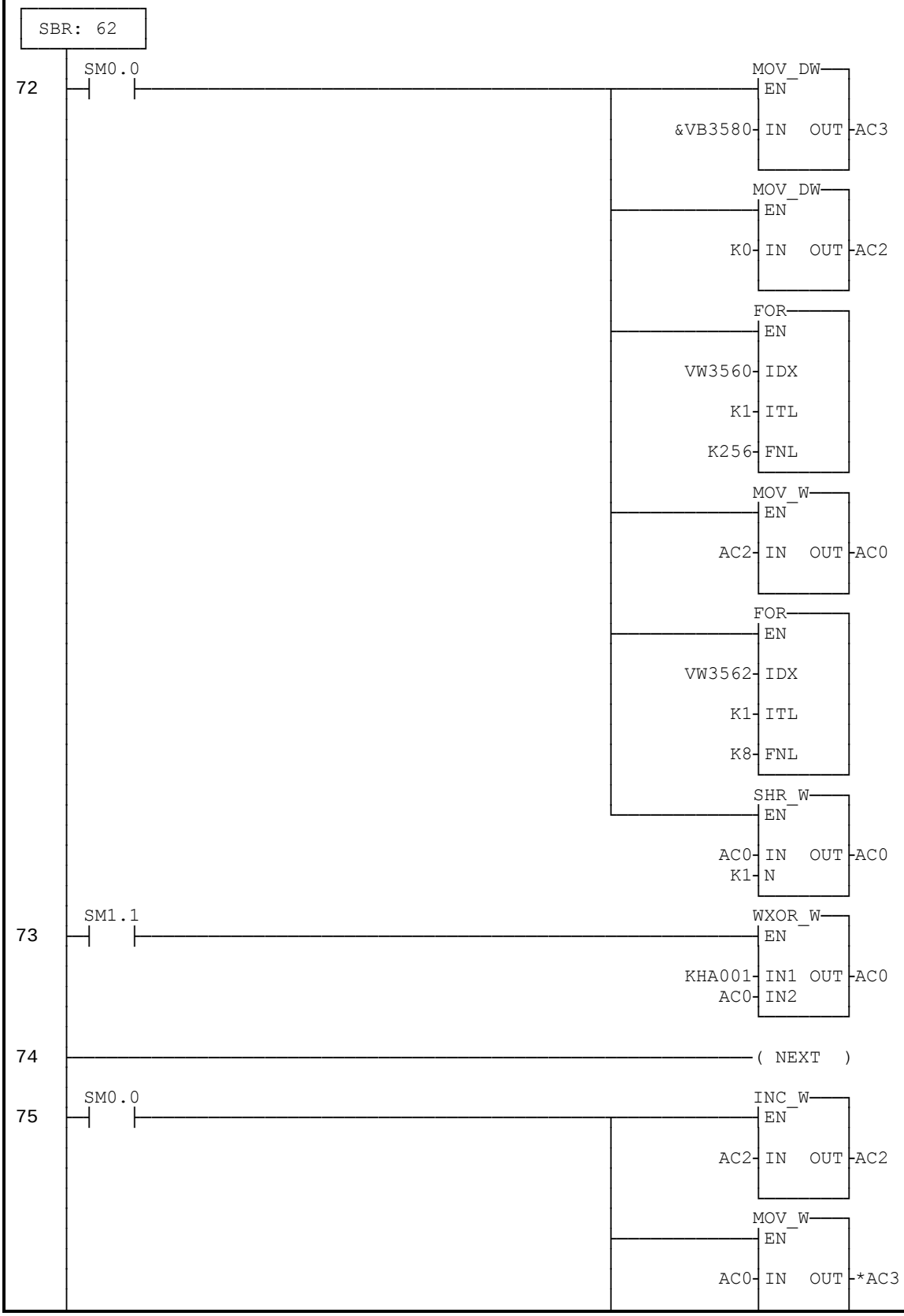

```
LD      SM0.0
MOVD    &VB0,AC3              // исходный_указатель = V память
+I      VW3304,AC3            // добавить смещение...
+I      VW3304,AC3            // ...дважды для смещения байта

BMW     VW3309,*AC3,VB3307    // запомнить данные в V память
MOVW    6,VW3300              // установить длину ответа
RET                           // возврат
```

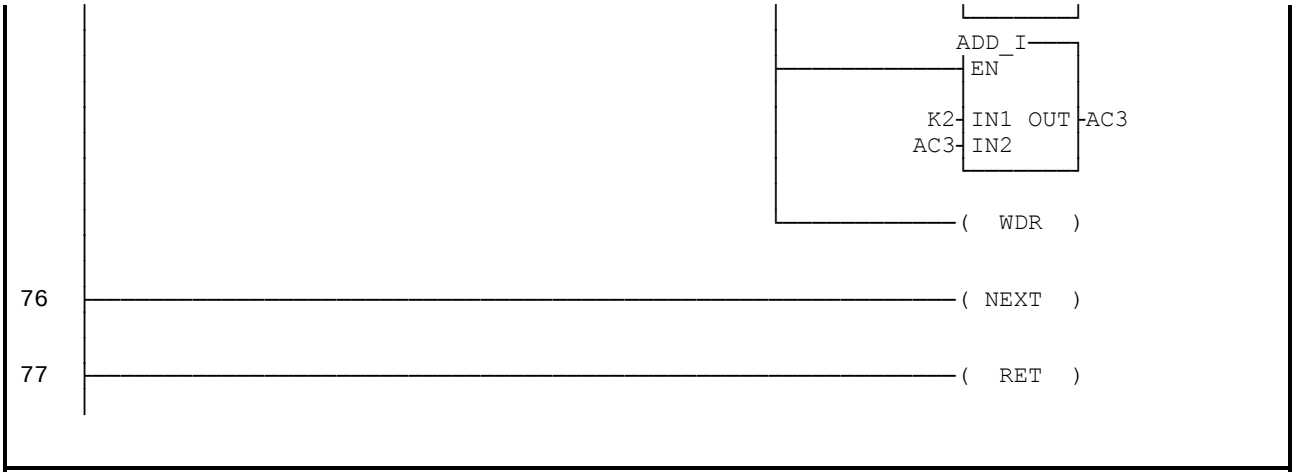
// Подпрограмма 61
//
// Данная подпрограмма устанавливает буфер ответа для особого кода Modbus два.




```
SBR     61
LD      SM0.0
MOVW    3,VW3300            // загрузить длину ответа об ошибке
ORW     16#0080,VW3302      // установить MS-Бит функции для показа ошибки
MOVB    2,VB3304            // загрузить код особой ситуации

RET
```

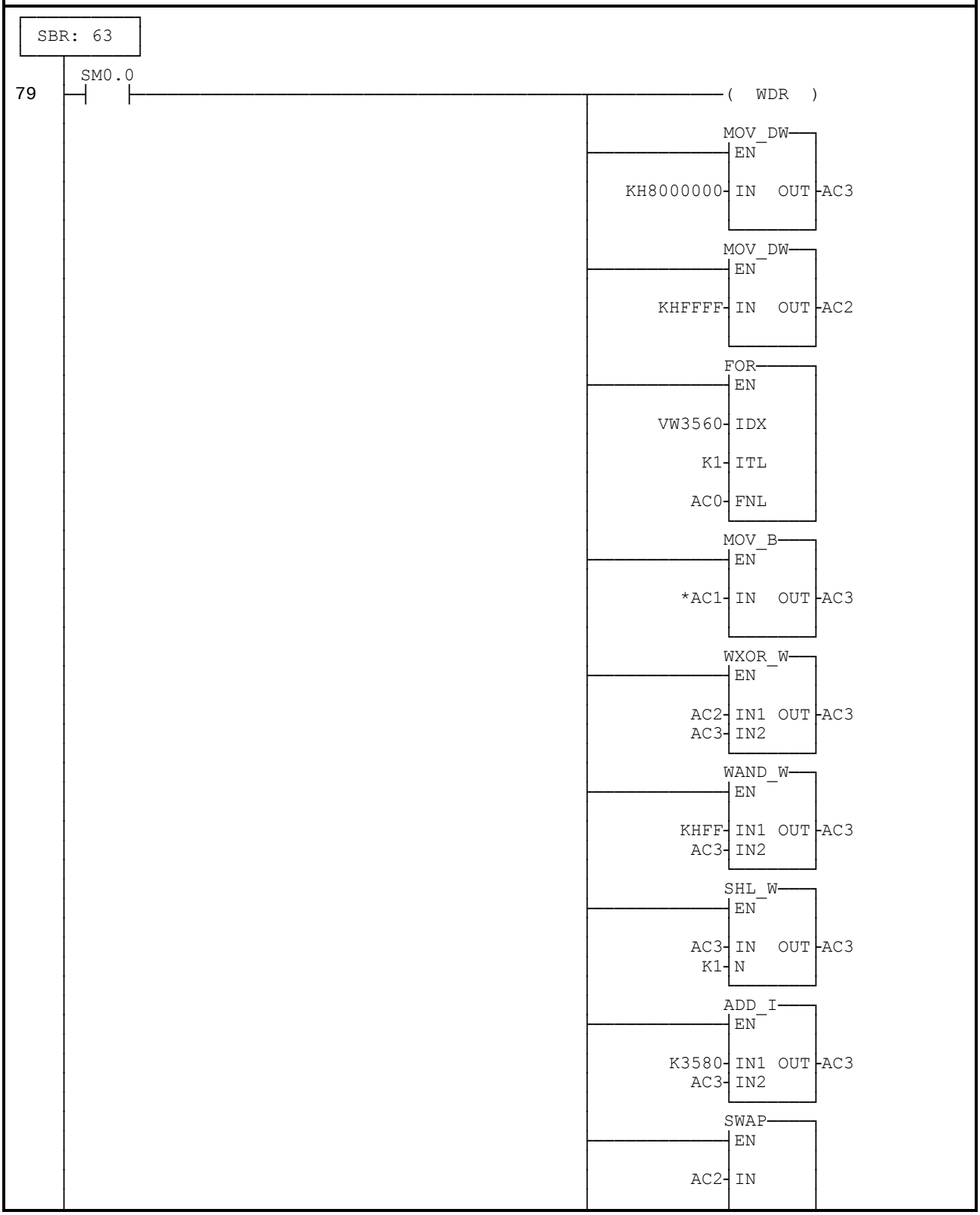
```
// Подпрограмма 62
//
// Эта подпрограмм генерирует таблицу CRC, используемую для подсчета значения CRC
// в телеграме.
//
// Используется многочлен:  x16 + x15 + x2 + 1
```

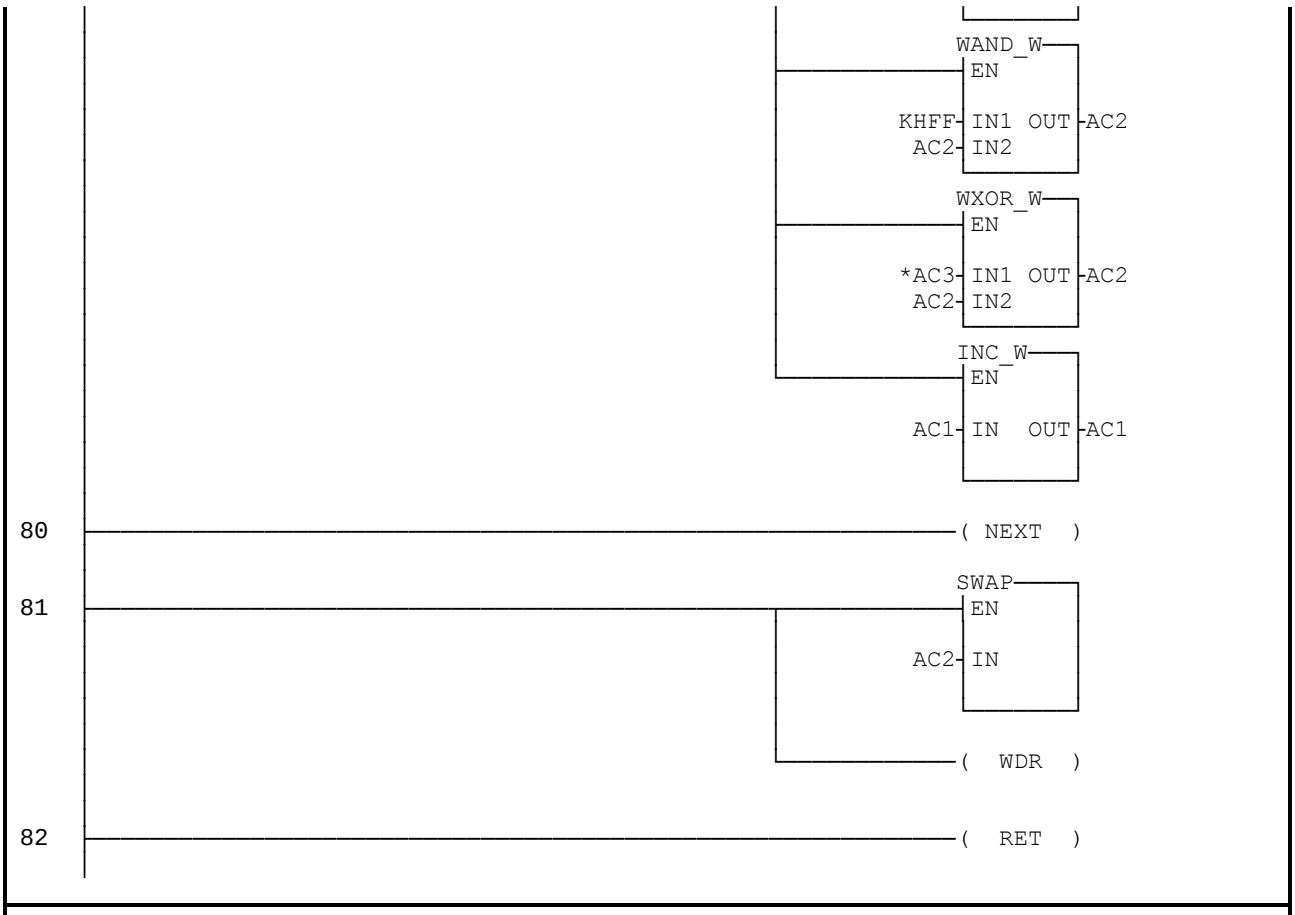





```
SBR     62
LD      SM0.0
MOVD    &VB3580,AC3          // загрузить указатель таблицы
MOVD    0,AC2                // загрузить индекс

FOR     VW3560,1,256         // для всех возможных значений байт
MOVW    AC2,AC0              //   значение = индекс
FOR     VW3562, 1, 8         //   для восьми бит из байта
SRW     AC0,1                //     сдвиг младшего бита
LD      SM1.1                //     если сдвинутый бит = 1
XORW    16#A001,AC0          //     XOR CRC со значением
NEXT                         //   конец для

LD      SM0.0
INCW    AC2                  //   увеличить индекс на единицу
MOVW    AC0,*AC3             //   запомнить слово таблицы
+I      2,AC3                //   увеличить на единицу указатель таблицы
WDR                          //   сбросить контрольный таймер
NEXT                         // конец для

RET
```

```
// Подпрограмма 63
//
// Данная подпрограмма расчитывает значение CRC для телеграммы, используя быстрый метод
// поиска в таблице.
//
// Входные парметры:    AC0    длина телеграммы
//                      AC1    указатель на телеграмму
//
// Выходные парметры:   AC2    значение CRC в младшем слове
```

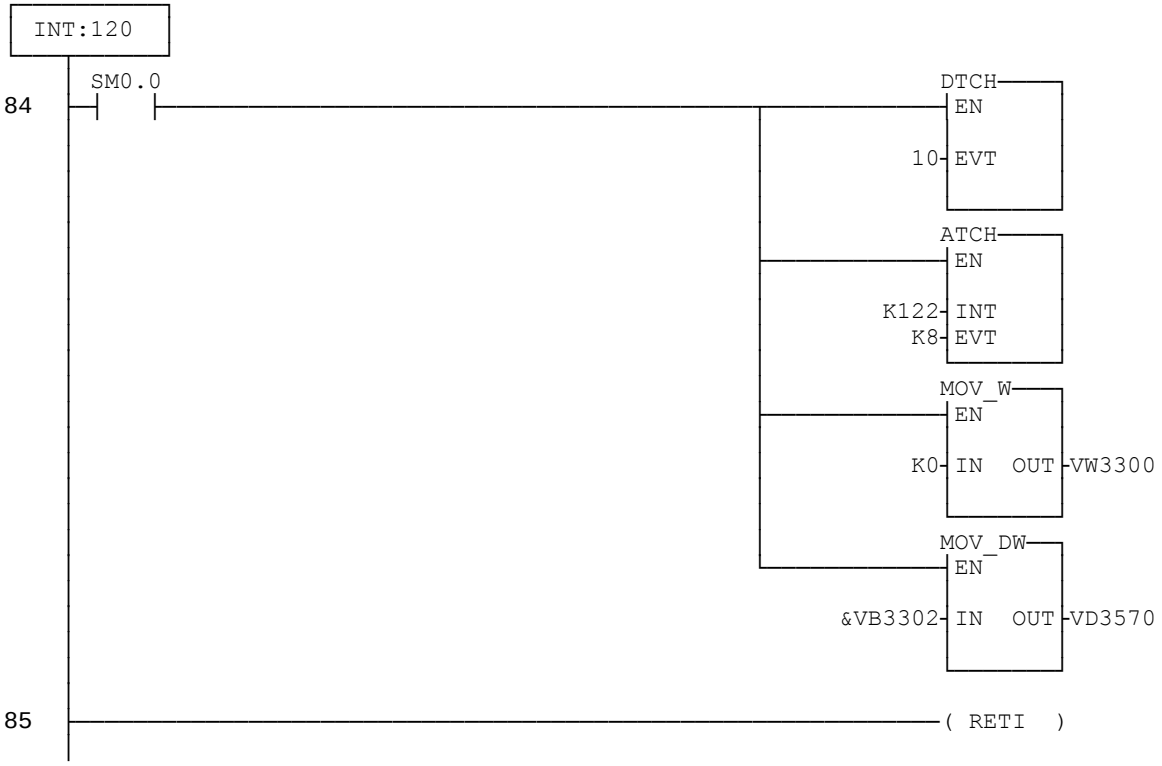





```
SBR     63
LD      SM0.0
WDR                          // сбросить контрольный таймер
MOVD    16#08000000,AC3      // очистить временный регистр
MOVD    16#0000FFFF,AC2      // инициализировать значение CRC = 0xFFFF

FOR     VW3560, 1, AC0       // для всех байт в телеграмме
MOVB    *AC1,AC3             //   получить байт телеграммы
XORW    AC2,AC3              //   XOR данные с CRC
ANDW    16#00FF,AC3          //   сохранить только младший байт
SLW     AC3,1                //   сконвертировать в индекс в слово таблицы
+I      3580,AC3             //   добавить начальный адрес таблицы (VB3580)
SWAP    AC2                  //   переставить байты в CRC
ANDW    16#00FF,AC2          //   сохранить только младший байт
XORW    *AC3,AC2             //   OR значение таблицы с CRC
INCW    AC1                  //   указатель на следующий байт телеграммы
NEXT                         // конец для

SWAP    AC2                  // переставить байты в CRC перед возвратом
WDR                          // сбросить контрольный таймер
RET
```

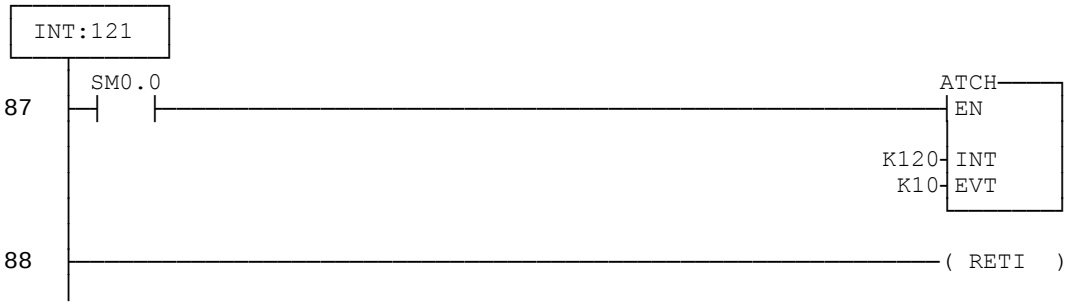
Программы обработки прерываний

// INT 120
//
// Данная программа обработки прерывания исполняется, когда превышен таймер
// свободной линии. Если это произошло, то присоединить программу прерываний для
// получения знака из порта, т.к. следующий принятый знак д.б. началом телеграммы.



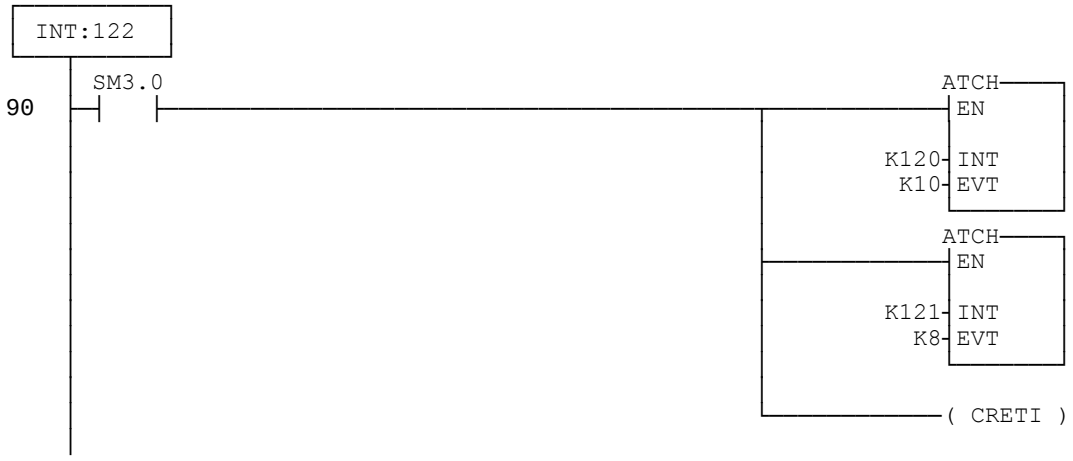
INT	**120**	
LD	**SM0.0**	
DTCH	**10**	**// блокировать таймер свободной линии**
ATCH	**122,8**	**// назначить программу приема**
		// к коммуникационному порту
MOVW	**0,VW3300**	**// установить счетчик приема в ноль**
MOVD	**&VB3302,VD3570**	**// установить указатель на буфер приема**
RETI		

// INT 121
//
// Данная программа обработки прерывания обрабатывает прерывания последовательного порта // для обнаружения свободной линии. Если во время обнаружения свободной линии принят
// символ, то перезапустить таймер свободной линии.

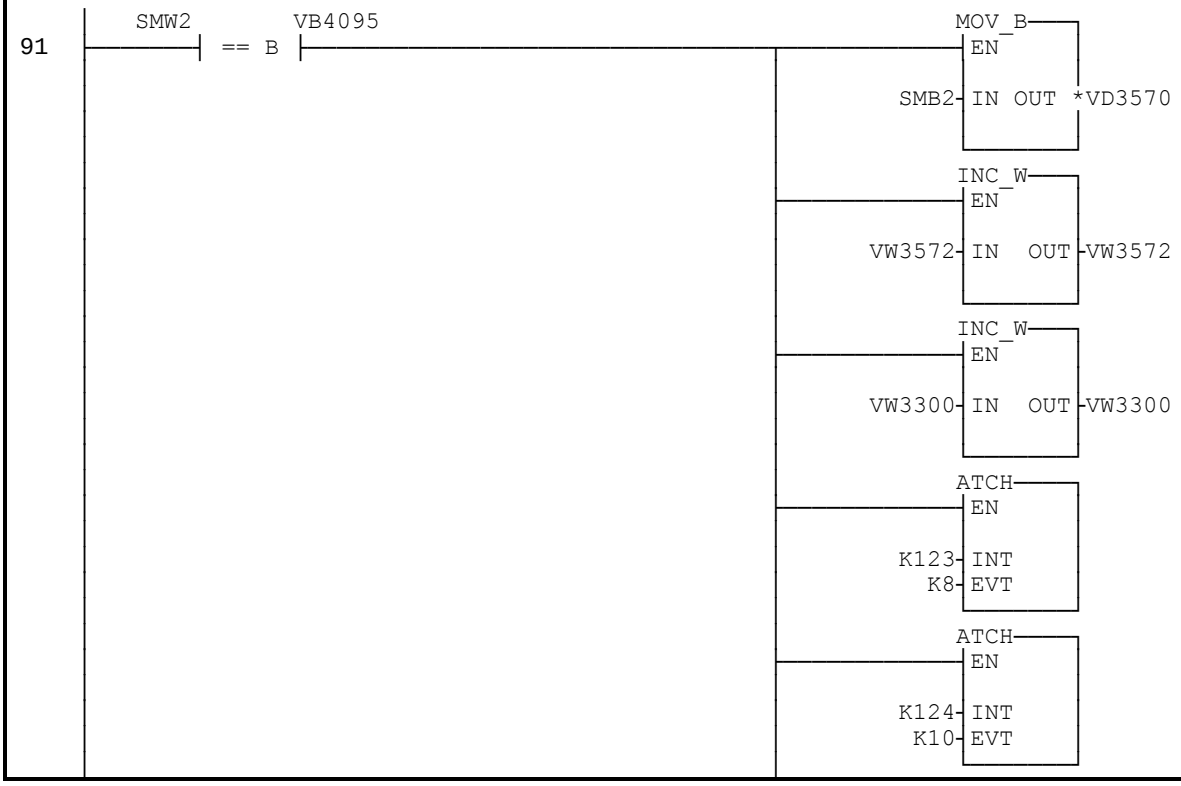


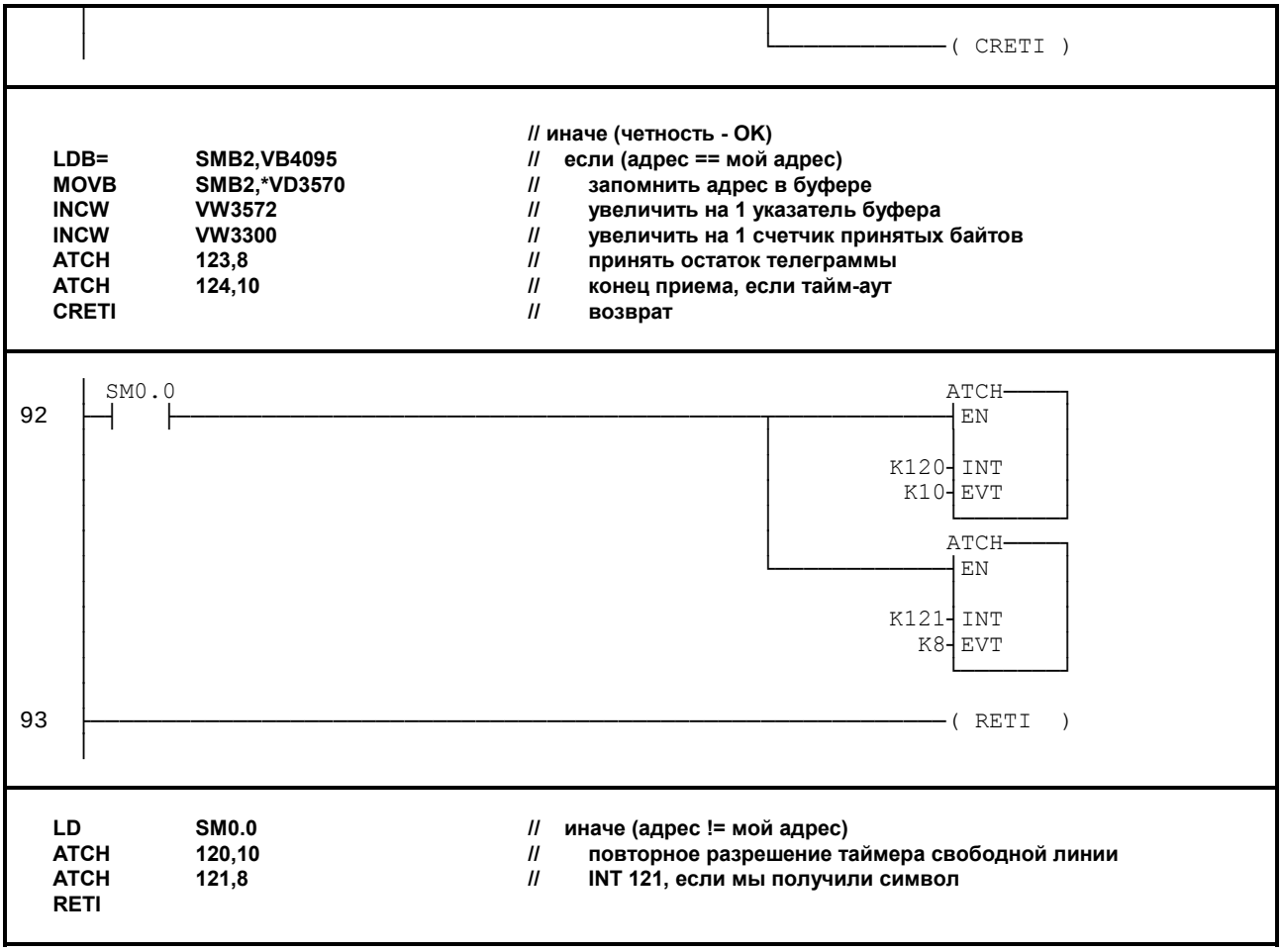

```
INT     121
LD      SM0.0
ATCH    120,10          // повторное разрешение таймера свободной линии
RETI
```

```
// INT 122
//
// Данная программа обработки прерывания принимает первый байт телеграммы после
// обнаружения времени свободной линии. Первый байт телеграммы - это адрес. Проверка,
// этот запрос для данного адреса. Если да, то присоединить INT 123 для приема всей
// телеграммы. Если телеграмма не для данного адреса, то возвратиться назад для
// поиска времени свободной линии.
//
// ЗАМЕЧАНИЕ:   широковещательный адрес в данной реализации не поддерживается,
//               т.к. он требует программирования дополнительного кода. Если данная
//               функция требуется, то д.б. модифицирована данная INT вместе со всеми
//               остальными функциями обработки, т.к. широковещательный адрес
//               не поддерживается всеми этими функциями.
```

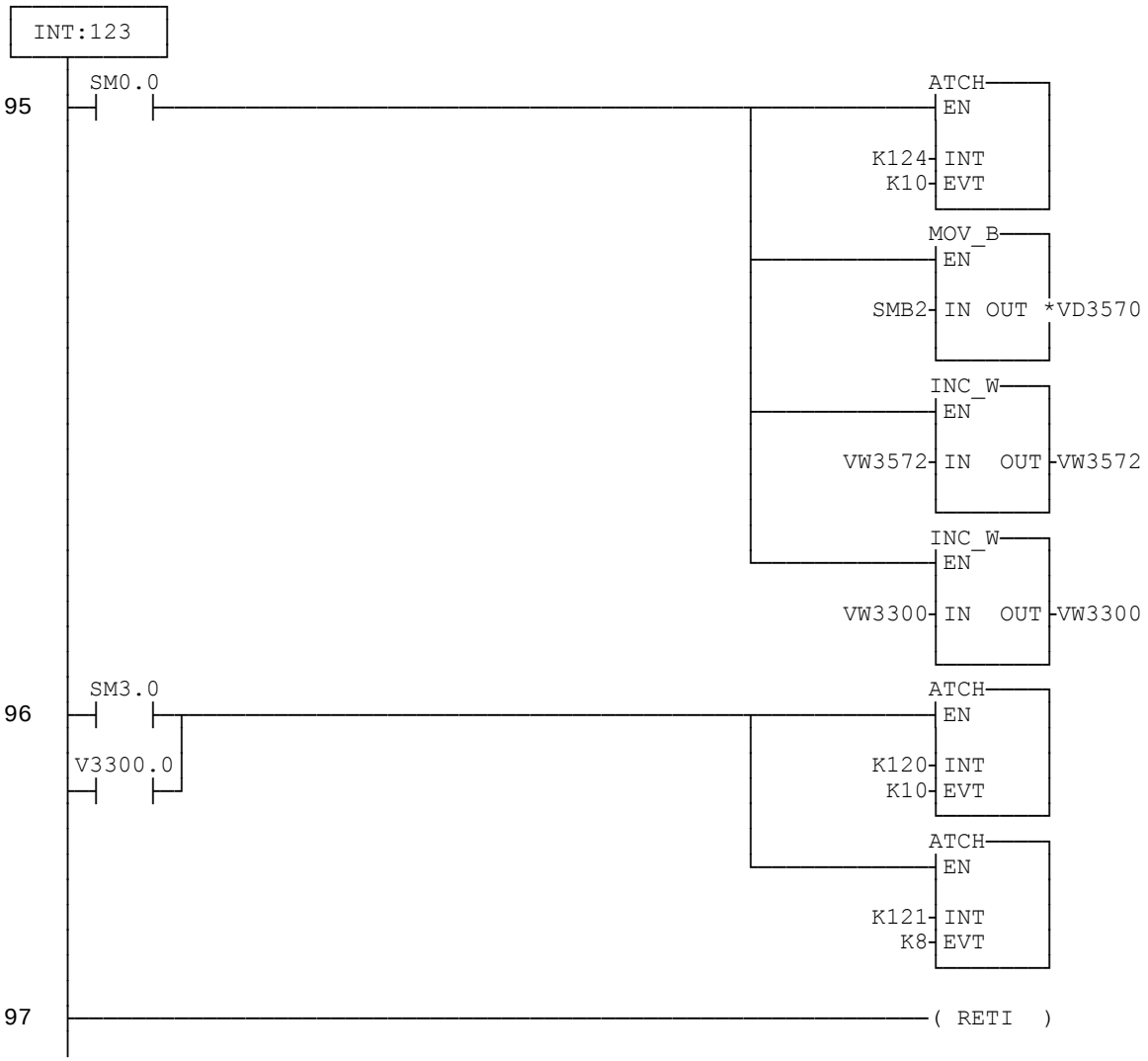



```
INT     122
LD      SM3.0          // если (ошибка четности)
ATCH    120,10         //   начальный поиск свободной линии
ATCH    121,8          //   INT 1, если мы получили символ
CRETI                  //   возврат
```



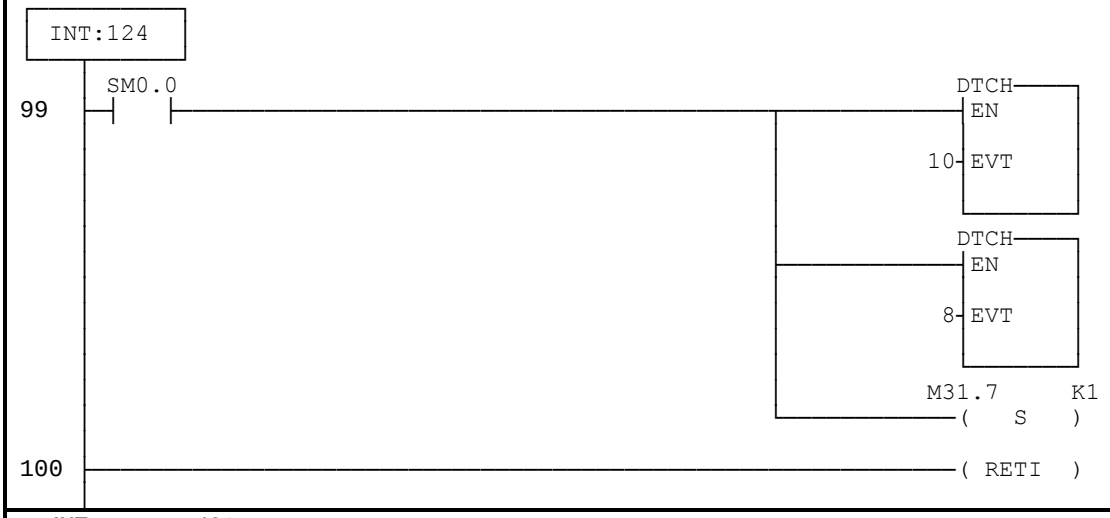

(CRETI)
// иначе (четность - OK)
LDB= SMB2,VB4095 // если (адрес == мой адрес)
MOVB SMB2,*VD3570 // запомнить адрес в буфере
INCW VW3572 // увеличить на 1 указатель буфера
INCW VW3300 // увеличить на 1 счетчик принятых байтов
ATCH 123,8 // принять остаток телеграммы
ATCH 124,10 // конец приема, если тайм-аут
CRETI // возврат
92
SM0.0
ATCH
EN
K120 INT
K10 EVT
ATCH
EN
K121 INT
K8 EVT
93
(RETI)
LD SM0.0 // иначе (адрес != мой адрес)
ATCH 120,10 // повторное разрешение таймера свободной линии
ATCH 121,8 // INT 121, если мы получили символ
RETI

```
// INT 123
//
// Данная программа обработки прерывания обрабатывает тело телеграммы.
// Символы принимаются, контролируется ошибка четности и затем они помещаются в
// коммуникационный буфер. Количество байт увеличивается на 1 для каждого символа.
// Если число принятых байт больше, чем 256, то прием прерывается.
```



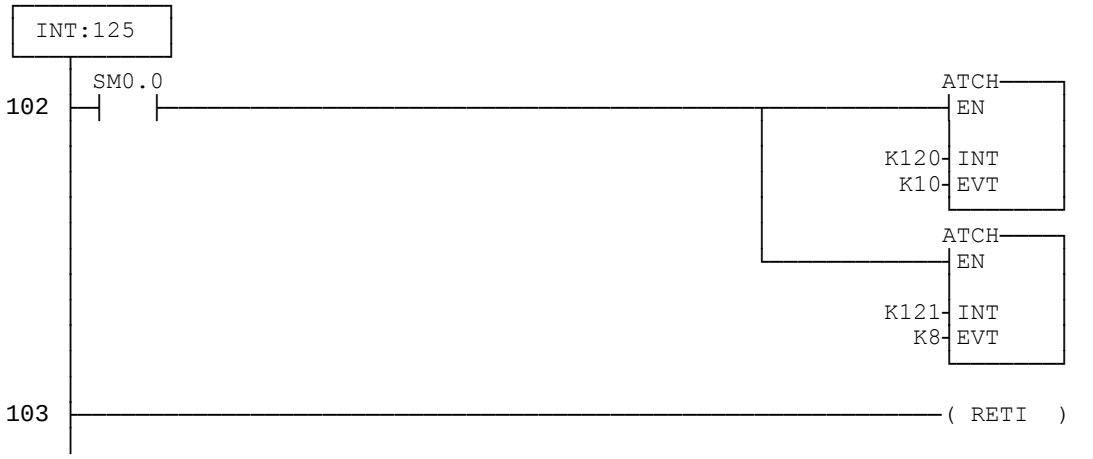

```
INT     123
LD      SM0.0
ATCH    124,10          // телеграмма закончена после следуещего тайм-аута MOVB
        SMB2,*VD3570    // запомнить байт телеграммы в буфере
INCW    VW3572          // увеличить на 1 указатель буфера
INCW    VW3300          // увеличить на 1 счетчик принятых байтов

LD      SM3.0           // если (ошибка четности) или
O       V3300.0         //    (переполнение буфера)
ATCH    120,10          //    начальный поиск свободной линии
ATCH    121,8           //    INT 1, если мы получили символ
RETI
```

```
// INT 124
//
// Данная программа обработки прерывания исполняется, когда была принята телеграмма
// и обнаружена еще раз свободная линия, это означает, что запрос завершен.
// Выполняемые действия - блокировать таймер свободной линии, блокировать прерывание
// порта и установить меркер фоновой обработки телеграммы.
```




```
INT     124
LD      SM0.0
DTCH    10              // блокировать таймер свободной линии
DTCH    8               // блокировать прерывание порта
S       M31.7,1         // установить меркер фоновой обработки
RETI
```

```
// INT 125
//
// Данная программа обработки прерывания исполняется, когда ответ был полностью
// передан обратно активной станции. Сброс системы прерывания для повторного поиска
// свободной линии и начала нового запроса.
```




```
INT     125
LD      SM0.0
ATCH    120,10          // начальный поиск свободная линия
ATCH    121,8           // INT 1 если получили символ
RETI
```

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все права защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC
S7-200 Примеры

Группа	Тема
3	**Замечания о сети S7-200 PPI**

Краткое описание

1. **Локальная сеть с маркерным кольцом**
2. **Обновления свободных адресов**
3. **Маркер, активные и пассивные станции**
4. **Добавление новой активной станции**
5. **Сигнал "Занято" и блокирование (1)**
6. **Сравнение MPI с PC/PPI**
7. **Сигнал "Занято" и блокирование (2)**
8. **Приложение A: Замечания для пользователей Micro/Win или Micro/DOS**
9. **Приложение B: Матрица сетевых компонент**
10. **Приложение C: Физические параметры**
11. **Приложение D: Информация о Panel-Tec**
12. **Общие замечания**

Объединение в информационные сети

Четыре соединенных устройства (два CPU 214 и два CPU 212) на Рисунке 42.1 образуют базовую сеть. Эта сеть будет служить основой нашей дискуссии для более сложных сетей и поможет проилюстрировать, что "можно" и что "нельзя" при объединении в сеть. Для обмена данными мы будем использовать стандартный протокол обмена для CPU 214 и 212 - PPI. В этом примере мы не будем обсуждать режим свободнопрограммируемого интерфейса (режим, в котором обмен данными происходит под контролем программы пользователя).

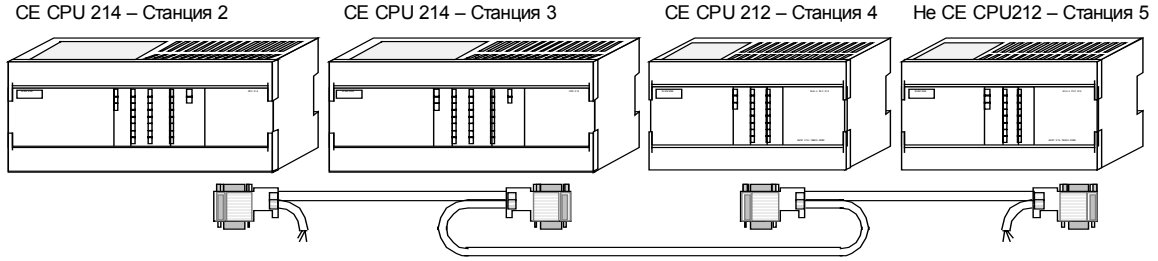
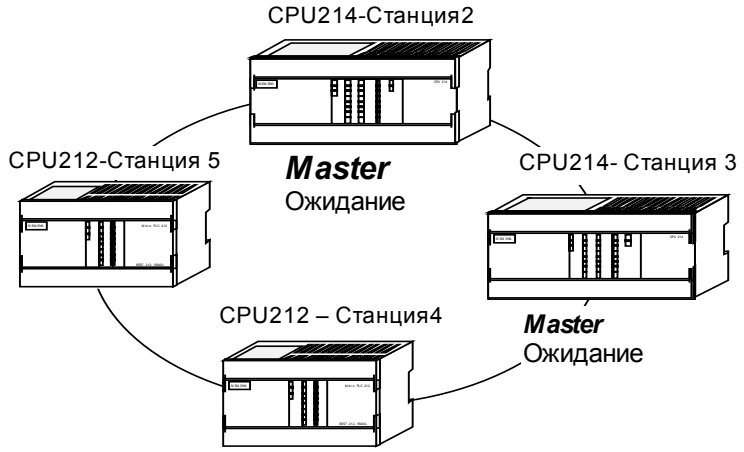


Рисунок 42.1 Базовая сеть

Сеть PPI является локальной сетью с маркерным кольцом (“Token Ring Networks”). Это означает, что активные станции ("активная станция") передают управление сетью от одной к другой посредством "маркера". Для лучшего понимания рассмотрим сеть из четырех станций, показанную на Рисунке 42.1. Станции 2 и 3 - активные, а станции 4 и 5 - пассивные ("slave").

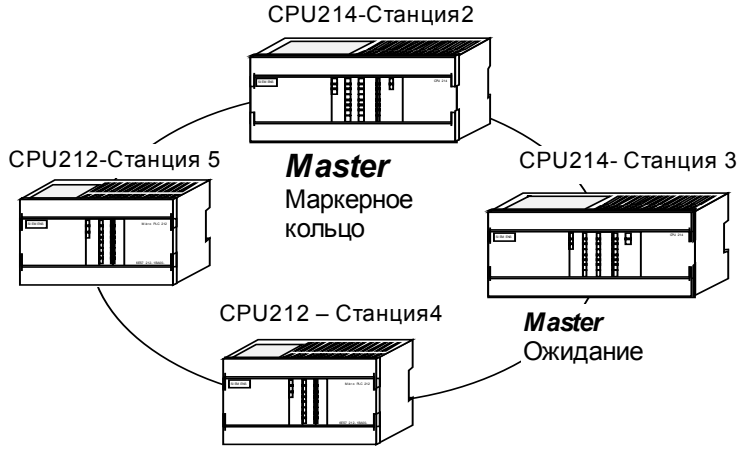
При включении питания пассивные станции просто запускаются и выпоняют свои программы, ожидая запроса из сети. Каждая активная станция перед тем, как подсоединиться к сети, анализирует трафик в сети и формирует список обнаруженных активных станций. Если активная станция закончила предустановленное время прослушивания сети (которое меньше для станций с меньшими адресами) без обнаружения какого-либо трафика, она образует сеть передав маркер самой себе. Если она успешно получила маркер, то сеть образуется, и после окончания ее собственного тайм-аута, другие активные станции включаются в сеть. Теперь посмотрим это на нашем примере.



При включении питания каждая активная станция ожидает некоторое время в зависимости от значения собственного адреса. Активная станция с наименьшим адресом заканчивает первой собственный тайм-аут и передает самой себе меркер,предотвращая,чтобы любая другая □тнация □е 4авладела . правлением.

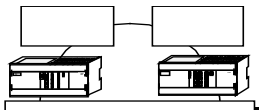
Рисунок 42.2

На риснке 42.2 показана сеть после включекния питания - обе активные станции ожидают находясь в режиме прослушивания. Первая активная станция (Станция 2) закончив анализ сети передает самой себе маркер, указывая всем другим активным станциям в сети, кто имеет контроль. Это простейшее маркерное кольцо, одна активная станция периодически передает маркер самой себе (см. Рисунок 42.3).



Станция 2 образует кольцо из одной активной станции, постоянно передавая самой себе маркер, а также перед тем,как передать маркер пытается обнаружить другие активные станции, ожидающие, чтобы войти в сеть. Если после опроса станции 3 она получает положительный ответ, то станция 3 включается в кольцо.

Рисунок 42.3



Обновление свободных адресов

Перед тем, как передать маркер, активная станция с маркером выполняет так называемое обновление списка свободных адресов (‘gap update’). В этот период опрашивается конкретный адрес, чтобы увидеть есть ли активная станция, желающая вступить в сеть (список свободных адресов активных станций состоит из всех адресов от самой станции до следующей активной станцией, получающей маркер). Таким образом, если первая подключенная станция - это Станция 2, то первый раз перед тем, как передать маркер, она проверяет есть ли активная, готовая присоединиться станция с адресом 3. Во время следующего цикла она проверяет Станцию 4, затем Станцию 5 и так далее, до тех пор пока не дойдет до самой себя (см. Рисунок 42.4).

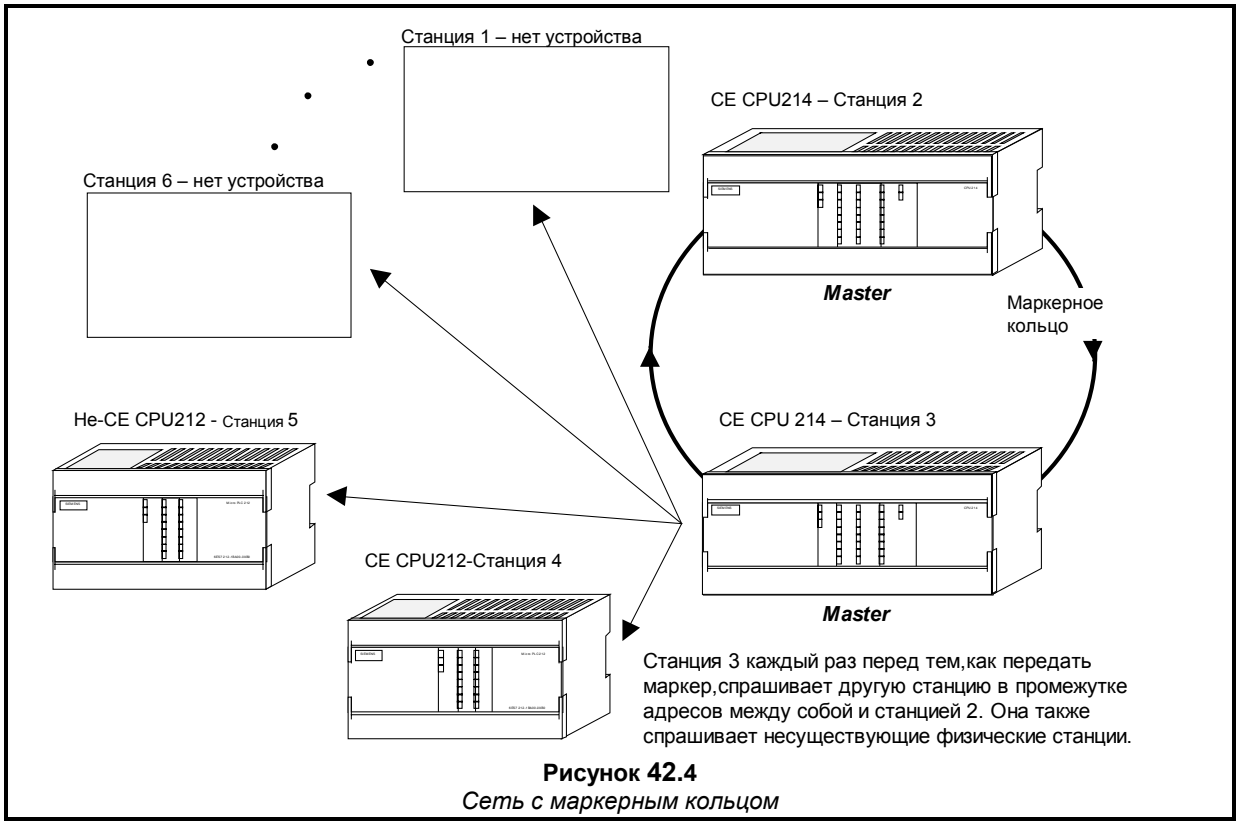


Рисунок 42.4
Сеть с маркерным кольцом

После того, как во время обновления свободных адресов все активные станции дали положительный ответ и вступили в сеть, между ними по логическому кольцу передается маркер. Активные станции, адреса которых не соприкасаются с адресом следующей активной станции, продолжают обновление свободных адресов (см. Рисунок 42.5).

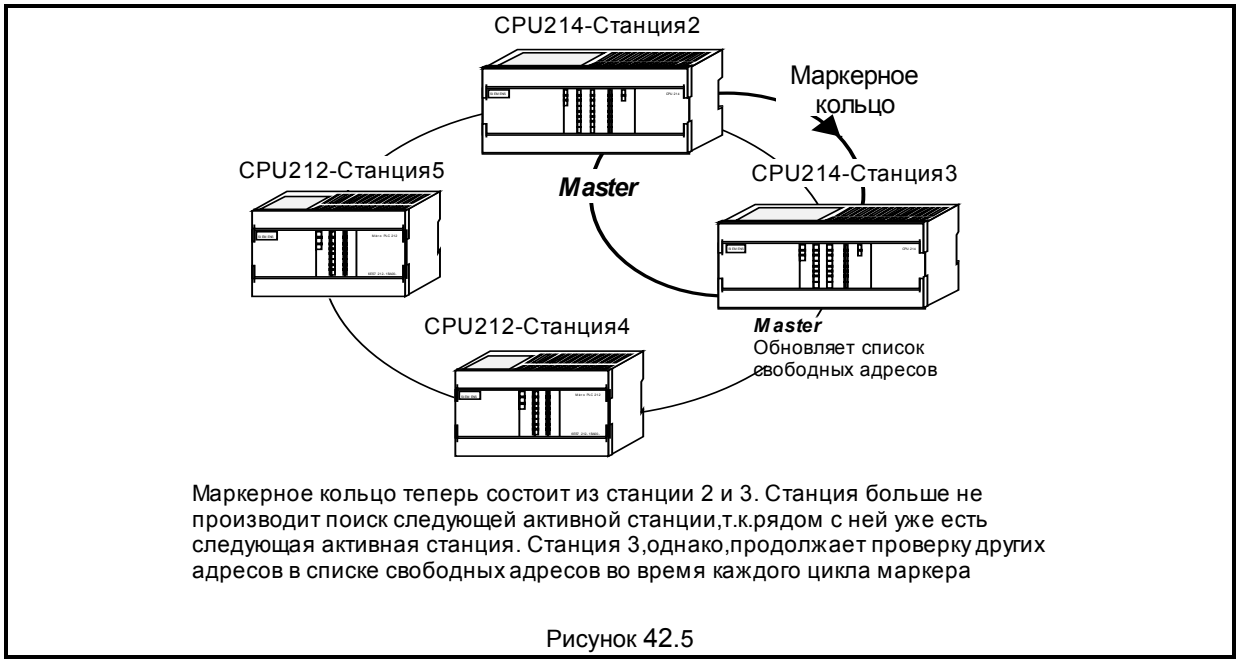
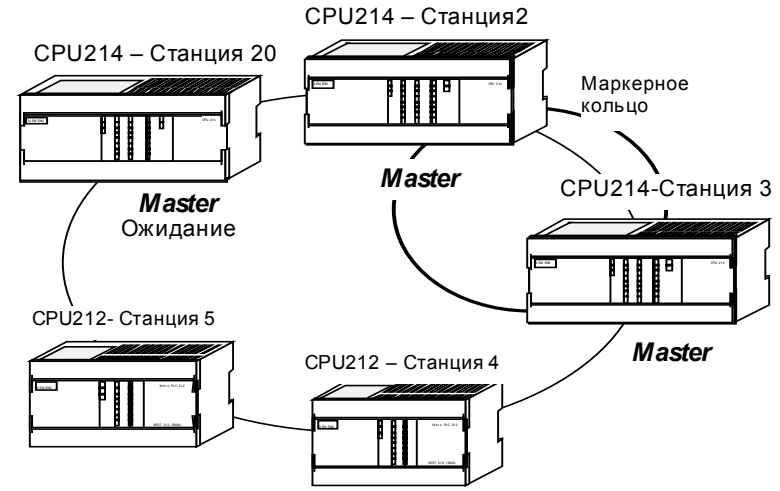


Рисунок 42.5

В нашем прмере все активные станции уже в сети, и только Станция 3 продолжает обновление свободных адресов.

Теперь наступило время обсудить разницу между активными и пассивными станциями. В локальной сети активная станция - это устройство, которое может владеть маркером, а пассивная - не может. Согласно этому определению CPU 212 всегда является пассивной станцией; оно может только получать запрос из сети и отвечать на него. Но оно не может владеть маркером или самостоятельно выдавать сетевые команды. С другой стороны CPU 214 может быть или активной или пассивной станцией в зависимости от конфигурирования специальных байтов памяти. Если оно является пассивной станцией, оно ведет себя как CPU 212 - может принимать сетевые команды, но не выдавать их. Когда оно является активной станцией, оно может владеть маркером, выдавать сетевые команды, *а так же может отвечать на запросы других активных станций.* Этот момент часто упускается из виду: активная станция так же может являться объектом для чтения или записи другой активной станции.

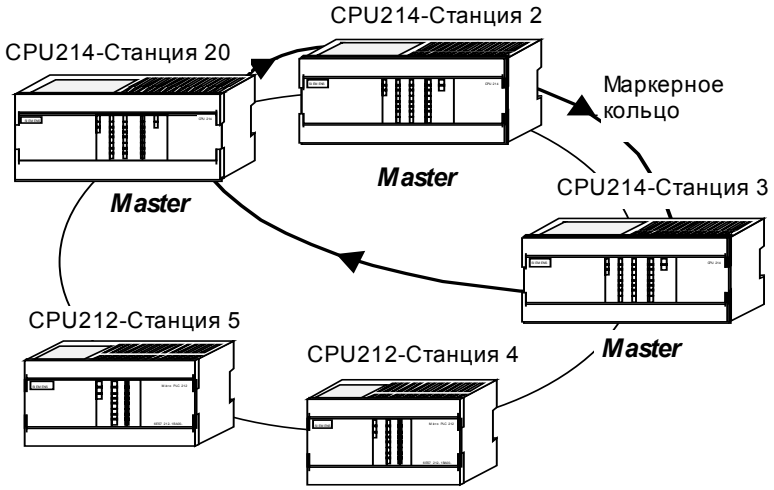
Давайте теперь предположим, что к сети подключили другую активную станцию с адресом 20 и подали питание. При первом включениив сеть Станция 20 запускает собственный цикл тайм-аута, но обнаружив, что уже есть маркерное кольцо, она перед тем, как перейти в состояние "готова присоединиться", переходит в режим ожидания на время двух полных оборотов маркера (два цикла сети) для построения карты текущей сети (см. Рисунок 42.6).



Новая активная станция при первом подключении начинает с обычного тайм-аута,но, увидев, что уже есть прохождение маркера в сети, ожидает два полных цикла для построения карты текущей сети, а затем ожидает опроса, чтобы присоединиться.

Рисенок 42.6

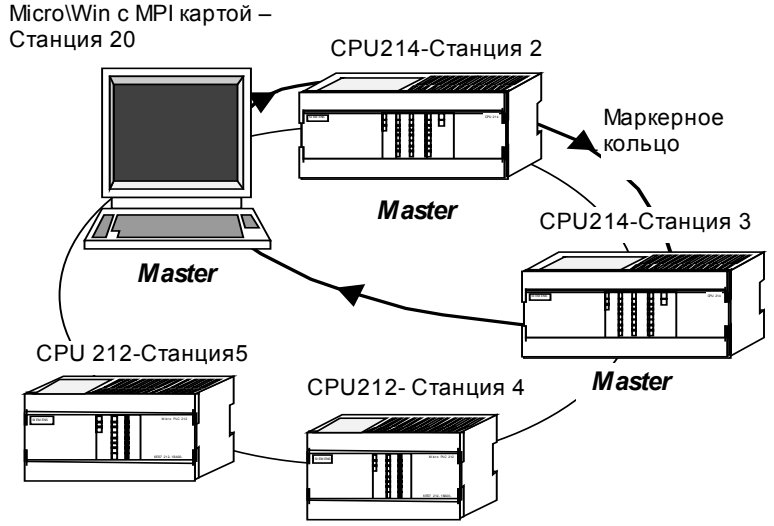
Когда она готова присоединиться, Станция 20 ожидает опроса от Станции 3. Когда обновление списка свободных адресов Станции 3 опять доходит до Станции 20, Станция 20 отвечает положительно и вступает в маркерное кольцо (см. Рисунок 42.7).



Станция 20 присоединилась к маркерному кольцу.Начиная с этого момента станции 3 и 20 обе выполняют обновление свободных адресов во время каждого цикла(обнаружение новых активных станций),т.к. адреса активных станций не являются соседними.

Рисунок 42.7

Давайте для наших дальнейших примеров предположим, что для Станции 20 вместо CPU, мы имеем 486 компьютер работающий под управлением Micro/Win или Micro/DOS. Давайте также предположим, что Станции 20 и 2 имеет несколько сообщений для Станции 3, и создадим конфликтную ситуацию.

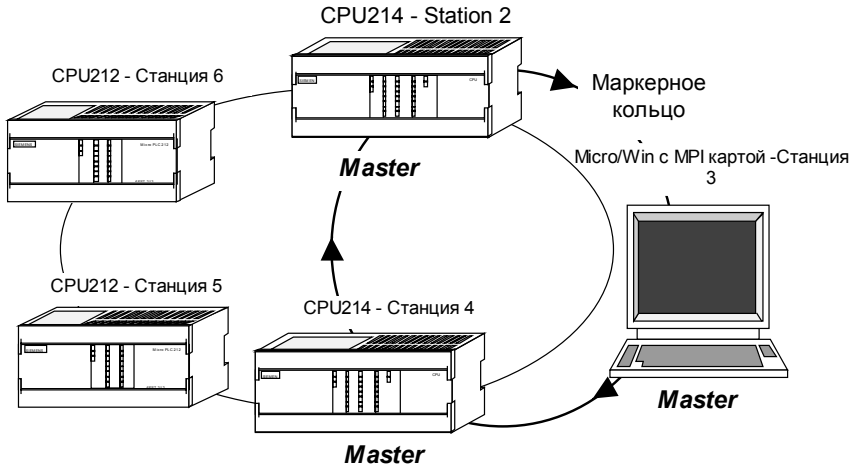


Станция 20 заменена на компъютер с MPIтой, работающий под управлением Micro\Dos или Micro\Win. В данной ситуации компъютер может блокировать одно или несколько других устройств в сети.

Рисунок 42.8

Основная проблема заключается в следующем: программатор с аппаратно реализованным PPI и быстрым процессором может связать CPU или другую станцию, так что никакая другая активная станция не сможет установить с ней связь (блокирование других активных станций). В этом случае CPU отвечает на запрос другой активной станции, что оно занято.

Существует, однако, два простых решения данной проблемы. Для предотвращения того, чтобы программатор имел дополнительное время на запросы для обновления списка свободных адресов перед тем, как передать маркер, надо поместить сразу после него другую активную станцию со следующим адресом станции или установив HSA (Highest Станция Address - Старший Адрес Станции) равным 20. Первый метод применен на Рисунке 42.9; Вы можете видеть, что для предотвращения обновления программатором списка свободных адресов, программатор теперь имеет адрес 3 и сразу после него стоит другая активная станция с адресом 4.



Станция 20 была перемещена на место станции 3 для предотовращения обновления своего списка свободных адресов и блокирования других активных станций.

Рисунок 42.9

В вышеприведенном примере нарисован программатор с MPI картой. Это очень важный пункт: программатор с PC/PPI кабелем очень отличается от программатора с MPI картой. Разница втом, что PC/PPI кабель не м.б. подсоединен к сети с несколькими активными станциями. Поэтому, если Вы программируете в сети используя PC/PPI кабель, то только программатор м.б. активной станцией в сети; в противном случае Вы получите сообщение об ошибке. А если Вы используете MPI карту, то к сети м.б. одновременно подключено и работать любое возможное число активных станций (до 32). Рисунок 42.10 показывает базовую сеть включая программатор с кабелем PC/PPI.

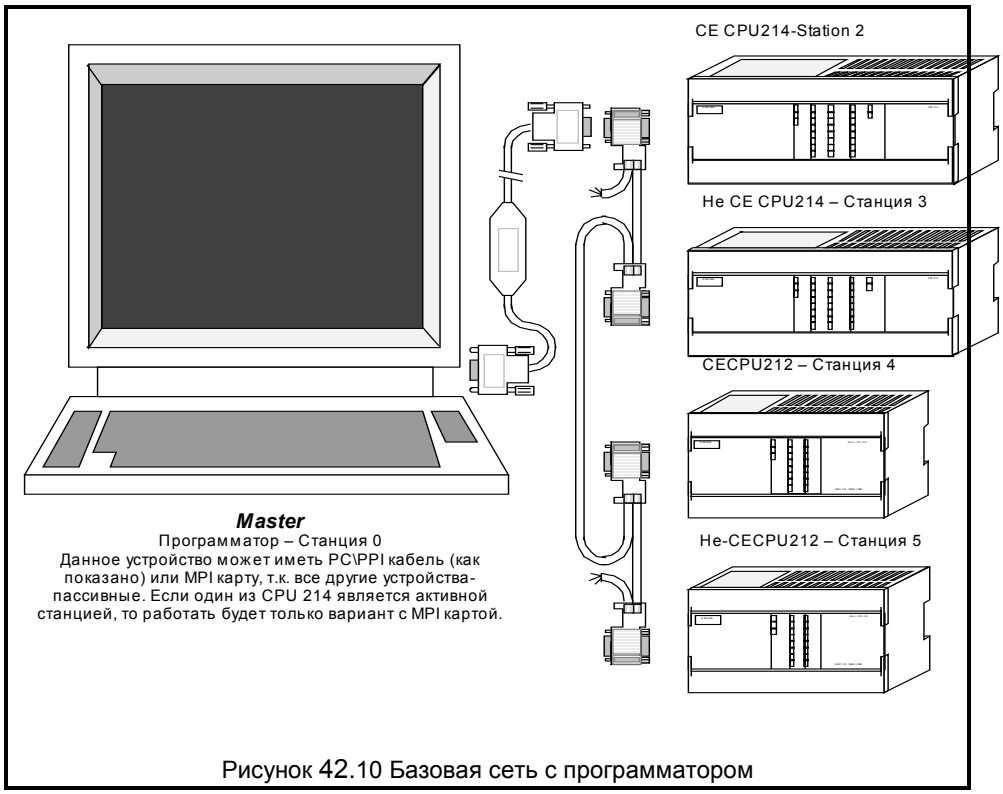


Рисунок 42.10 Базовая сеть с программатором

Вместе с программаторами и пассивными станциями, существует большое количество периферийных устройств, которые можно подключить к сети, такие как TD200 и OP3. Оба этих устройства обмениваются по сети с определенным CPU и ведут себя как активные станции. На Рисунке 42.11 показано физическое соединение двух TD200, OP3, и программатора с картой MPI с базовой сетью, представленной ранее.

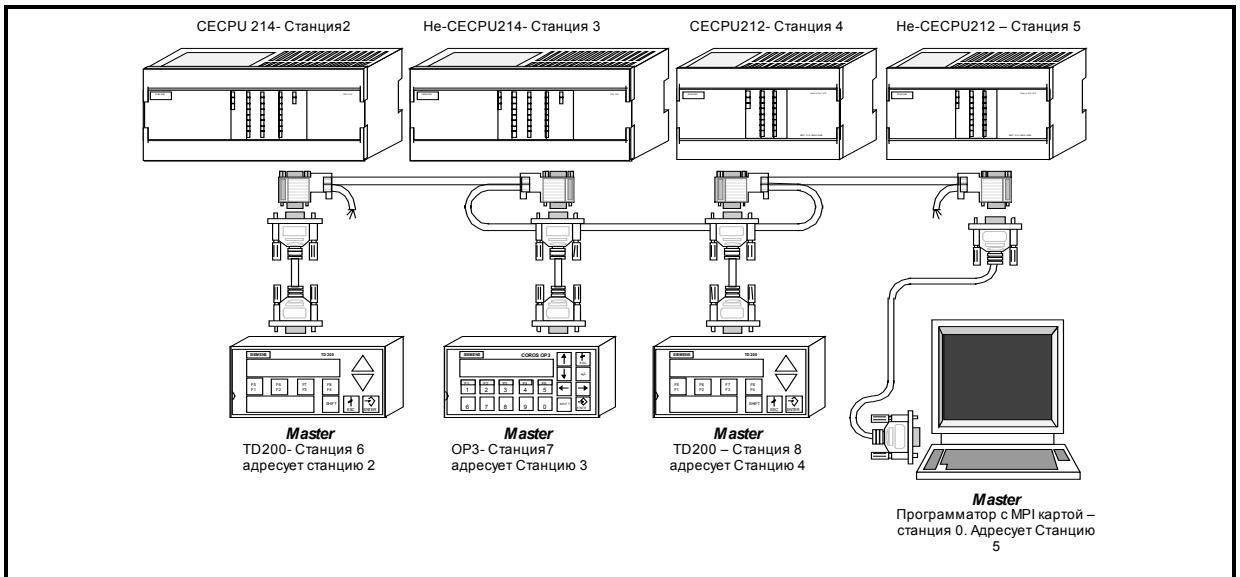


Рисунок 42.11 Базовая сеть с добавленной периферией

Важно отметить, что все периферийные устройства являются активными станциями, и составляют часть маркерного кольца (Рисунок 42.12).

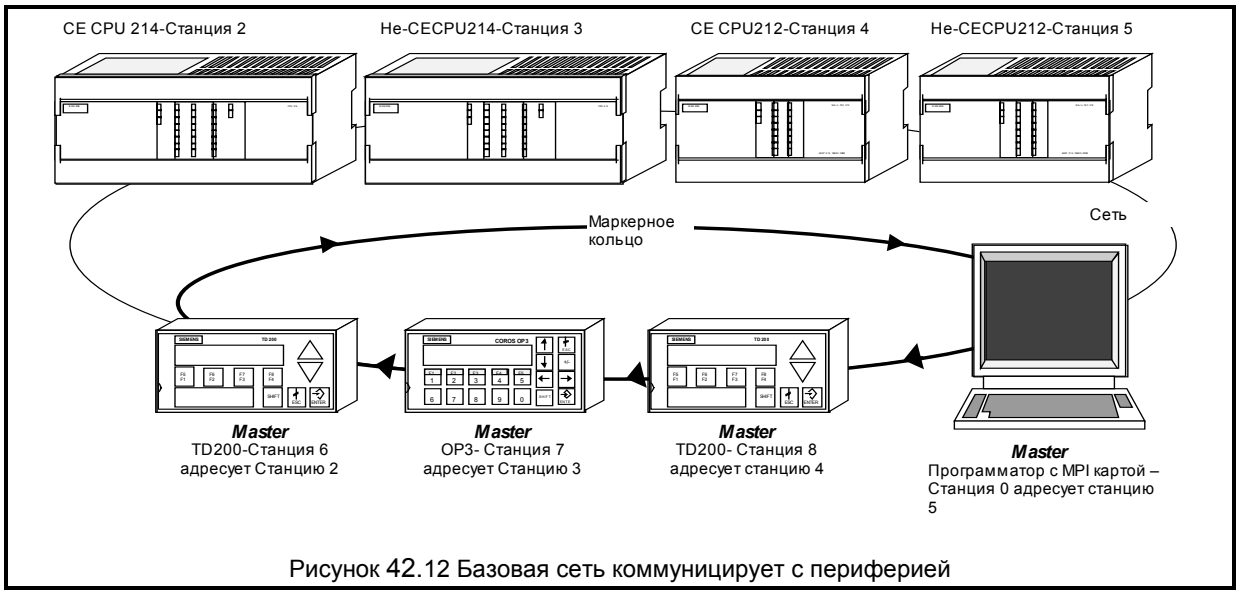


Рисунок 42.12 Базовая сеть коммуницирует с периферией

Итак вопрос, почему эти устройства являются активными станциями? Простейший ответ следующий: каждое периферийное устройство должно инициировать запрос для сбора данных из нескольких CPU. Для этого они должны иметь доступ к сети. Обычно Каждое устройство обменивается с отдельным CPU (см. Рисунок 42.13).

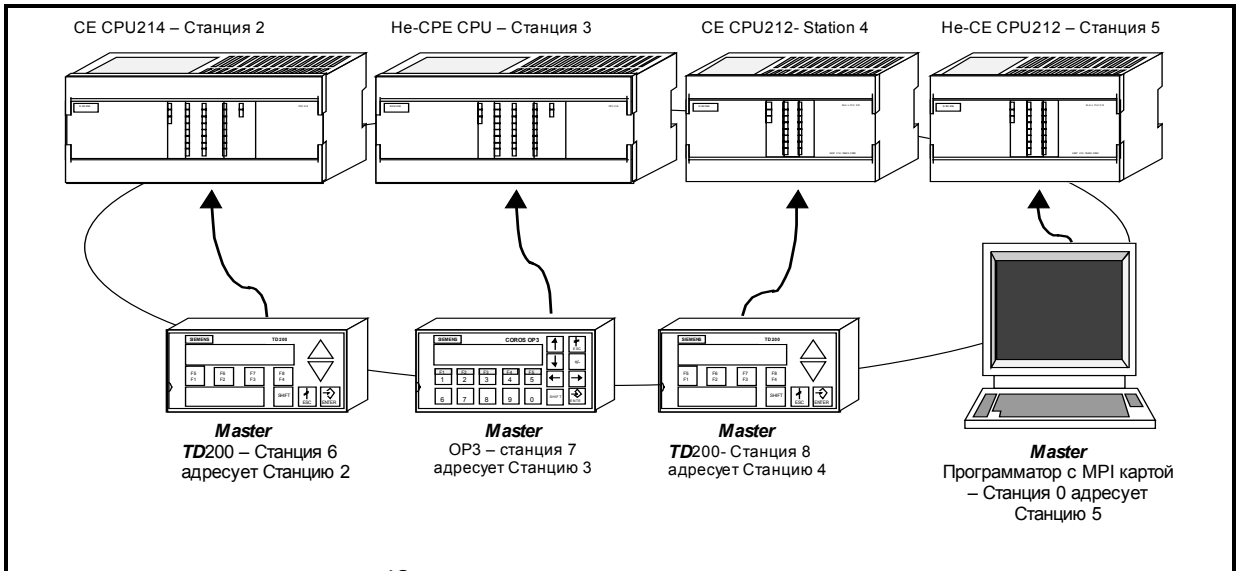
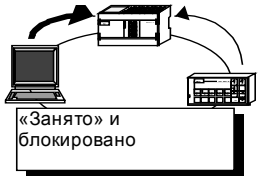


Рисунок 42.13 Периферия коммуницирует с базовой сетью



Однако, если каждое из этих периферийных устройств попытается адресовать одну и ту же станцию, может возникнуть та же проблема, которую мы обсудили ранее (Рисунок 42.14). Если TD200 может справиться с данной проблемой, подобно программатору с картой MPI, то OP5 (и выше) может это с ограничениями, а OP3 не может вообще.

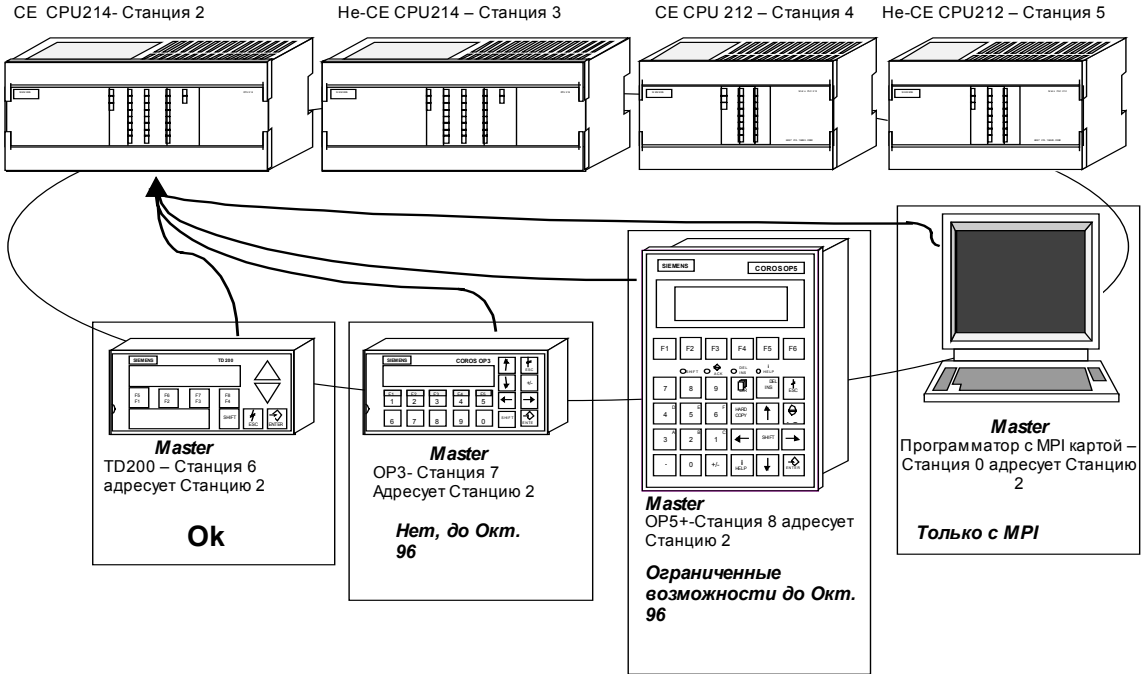


Рисунок 42.14 Коллизия в базовой сети

Приложение A:
Замечания для пользователей Micro/Win или Micro/DOS

Т.к. каждая активная станция перед тем, как войти в сеть, должна ожидать опроса, и т.к. список свободных адресов может содержать 100 и более различных адресов, то когда PC с MPI картой под управлением Micro/Win или Micro/DOS подсоединяется к сети, Вы можете получить ошибку тайм-аута, говорящую о том, что программатор не может соединиться с сетью. Лучшая рекомендация для данной ситуации - проигнорировать сообщение, подождать несколько секунд и работать дальше. Обычно за то время, пока Вы делаете что-то относящееся к сети, программатор успеет соединиться с сетью.

Приложение B: Матрица сетевых компонент

Устройство	Математика с плавающей запятой	Поддержка нескольких активных станций	Обработка сигнала "Занято"	Программно / Аппаратно реализованная связь
Micro/DOS порт RS232	Версия 1.2+			Программно
Micro/DOS порт MPI	Версия 1.2+	(	(	Аппаратно
Micro/Win порт RS232	(			Программно
Micro/Win порт MPI	(	(	(	Аппаратно
TD200	Версия 1.1+	(	(	Программно
OP3, OP5	*Нет до Октября '96*	(Если другая активная станция управляется программно. Нет, если аппаратно *до Октября '96*	*Нет до Октября '96*	Аппаратно
OP15, OP20, OP25, OP35	(*Начиная с Марта '96*	(*Начиная с Марта '96*	*Нет до Октября '96*	Аппаратно
CPU 212 Версия .92		(	(	Программно
CPU 212 CE Версия 1.0+		(	(	Программно
CPU 214 Версия .91		(	(	Программно
CPU 214 CE Версия 1.0+	(	(	(	Программно
Драйвер Win CC PPI		В разработке		
Драйвер PRODAVE PPI		В разработке		
PG702				Аппаратно
PanelTec DTU3000	(	(	(	Программно

Данная матрица поможет Вам в разрешении возможных сетевых проблем. Некоторые из наиболее часто встречающихся ошибок при работе с сетью возникают из-за того, что в сети есть устройства, которые не могут нормально коммуницировать. Лучший способ избежать их следующий: первое - проверте, что Вы не используете несколько активных станций в сети с устройствами, которые не поддерживают соединение с несколькими активными станциями; второе - что Вы не используете устройства, которые не могут обрабатывать сигнал "занято" вместе с устройствами аппаратно реализующими протокол обмена.

Приложение C: Физические параметры

- До 32 устройств без использования повторителя RS485
- До 128 устройств с использованием повторителя RS485
- Максимальное число активных станций на сети - 32.
- Длина кабеля - не более, чем 10 км
  (Конкретный кабель может отличаться,
  проконсультируйтесь с поставщиком)
- Рекомендованные кабели:
  Кабель Beldin: 3079A
  Кабель Siemens: 6XV1830-0AH10
- Рекомендованные штеккеры:
  Без интерфейса к программатору: 6ES7 972-0BA20-0XA0
  С интерфейсом к программатору: 6ES7 972-0BB20-0XA0
- На концах сети должны стоять терминаторы.

Приложение D: Информация о Panel_Tec

Устройство Panel-Tec, указанное в матрице сетевых компонент (Модель номер: DTU-3000), представляет собой последовательное устройство, которое передает информацию между разными CPU. Оно поддерживает более 25 протоколов и включает 2 порта CPU и 1 конфигурационный порт / порт расширения (каждый порт м.б. RS232 или RS422).

Тел.: 1-800-869-2026 / (205) 534-8132 Факс: (205) 534-4769
Адрес: P.O. Box 2394
Huntsville, Alabama USA
35804-2394

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все прав защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC
S7-200 Tips

Group	Topic
1	**Emulating an External Potentiometer**

Overview

This example shows how, with a connected timer, one may emulate an external potentiometer (POT) similar to the integrated analog potentiometers of the S7-200 PLCs. When using the onboard POTs, the settings have to be made directly on the PLC. However, when emulating an external POT, settings can be made from other locations like from the outside of a control cabinet.

The principle is based on the following: the CPU transmits a signal to the timer; after a time that can be set (here between 0.5 and 10 seconds), the timer transmits a return signal. The CPU measures this signal which corresponds to a time entry. Depending on requirements, this time entry can be used to respond correspondingly faster or slower. In this example, the value of the POT is displayed using a TD200 for illustrative purposes.

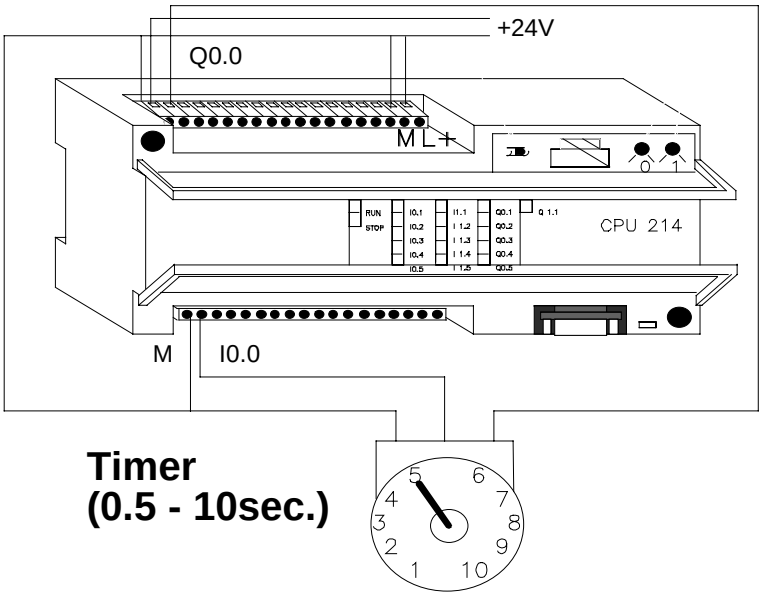
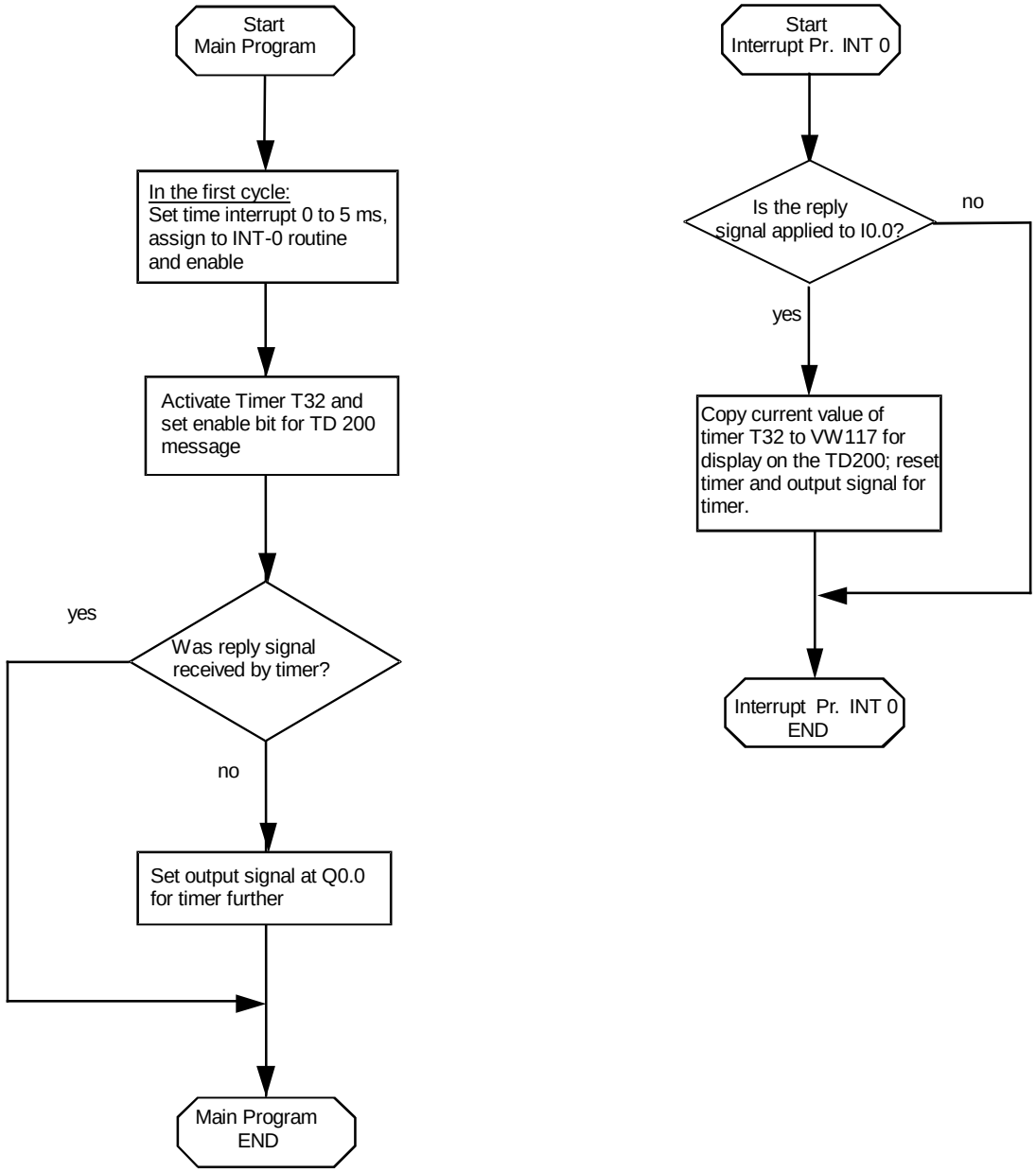


Figure 43.1 Connection of the Timer to the S7-200 PLC

Hardware Components

1	SIMATIC CPU S7-212 or S7-214
1	TD200 (optional)
1	Siemens Timer Part No.: 7PU8040-1AB30 Type: 0.5 to 10sec. / 24V DC Price: $102.00

Program Structure



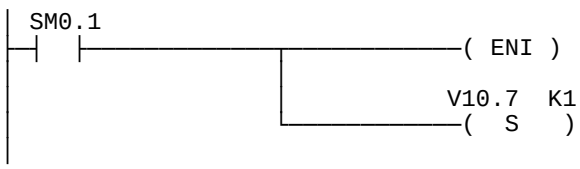
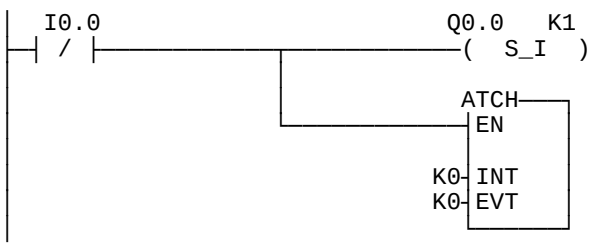
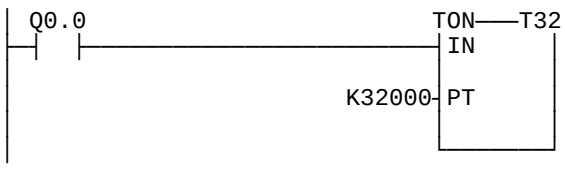
Program Description

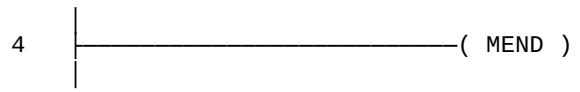
This program example consists of the main program and an interrupt routine. The main program initializes the time interrupt 0 in the first cycle by setting it to 5ms, then assigning it to the interrupt routine INT 0 and enabling it. Also, the main program activates Timer T32 (a 1ms timer), and assigns it the task of measuring the time interval between the signal to the timer and the reply signal. Moreover, the enable bit, V10.7, is set for the first message of the text display TD200 which displays the timer value. As long as there is no reply signal by the timer at input I0.0, the output signal is set for the timer at Q0.0 and thus maintained.

Every 5ms (the smallest possible time interval), time interrupt 0 checks to see if the timer has responded. With 0.05%, the maximum inaccuracy of 5ms which occurs (as against the maximum settable value of 10s at the timer) is negligible. If this is the case, the measured time value is transferred to the variable word VW117 so that it will be displayed on the TD200. To restart time measurement, the output signal for the timer is reset at Q0.0. The timer will reset its reply signal, and the whole process will start again.

This structure requires that the updating duration of the POT value corresponds to the set time value. This program uses a TD200 for displaying the POT value. Data block DB1 is needed for initializing the TD200.

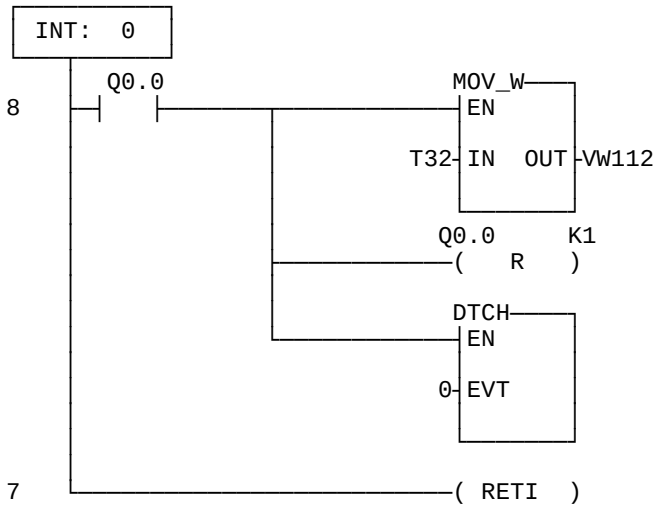
Main Program

LAD (S7-MicroDOS)	STL (IEC)
1 	**LD SM0.1 // in the first cycle** **ENI // enable interrupt(s)** **= V10.7 // and enable Message 1 // forTD200**
2 	**LDN I0.0 // if no signal // by timer** **SI Q0.0, 1 // set output signal** **ATCH 0, 0 //attach the interrupt**
3 	**LD Q0.0 // if output signal** **TON T32, 32000 // turn on timer 32**



MEND **// End Main Program**

Interrupt Programs



INT	**0**	**// Time Interrupt 0**
LD	**Q0.0**	**// If timer signal is // pending**
MOVW	**T32, VW112**	**// Copy time value to // TD200 text variable // VW112**
R	**Q0.0, 1**	**// reset output signal // for timer // for complete restart**
DTCH	**0**	**// detach interrupt 0**
RETI		**// End Time Interrupt**

Data Block DB1:

```
// Definition of the parameter block for the TD200

V0    'TD'        // ID for TD200
VB2   16#10       // Set the language to English, update as quickly as possible.
VB3   16#00       // Set the display for 20 characters.
VB4   1           // Set the number of messages to "1".
VB5   0           // Set the flag address to M0.0 to M0.7
VW6   100         // Set the address of the start message to VB100.
VW8   12          // Set the address of the bit for message enable to VB12.

// Define the message for the TD200
// Message 1

V100  ' POT Value:      '   // Text of message - 20 characters
VB115 16#10                 // MSB of the format word (making processing possible)
VB116 16#33                 // LSB of the format word (unsigned word value)
```

Conversion Notes

To Convert from IEC STL to S7-Micro/DOS STL:

- Add a 'K' before all non-Hex numerical constants (i.e. 4 K4)
- Replace '16#' with 'KH' for all Hex constants (i.e. 16#FF KHFF)
- Commas denote field divisions. Use arrow or TAB keys to toggle between fields.
- To convert an S7-Micro/DOS STL program to LAD form, every network must begin with the word 'NETWORK' and a number. Each network in this Application Tip program is designated by a number on the ladder diagram. Use the INSNW command under the EDIT menu to enter a new network. The MEND, RET, RETI, LBL, SBR, and INT commands each receive their own networks.
- Line-Comments denoted by '//' are not possible with S7-Micro/DOS, but Network-Comments are possible.

The data block shown was created in Micro/Win; in Micro/DOS use the V-memory editor to enter a data block (Refer to the Micro/DOS users' guide). Note that in Micro/Win:

- 16#.. denotes a hexadecimal value
- '*text*' represents a string value

General Notes

The SIMATIC S7-200 Application Tips are provided to give users of the S7-200 some indication as to how, from the view of programming technique, certain tasks can be solved with this controller. These instructions do not purport to cover all details or variations in equipment, nor do they provide for every possible contingency. Use of the S7-200 Application Tips is free.

Siemens reserves the right to make changes in specifications shown herein or make improvements at any time without notice or obligation. It does not relieve the user of responsibility to use sound practices in application, installation, operation, and maintenance of the equipment purchased. Should a conflict arise between the general information contained in this publication, the contents of drawings or supplementary material, or both, the latter shall take precedence.

Siemens is not liable, for whatever legal reason, for damages or personal injury resulting from the use of the application tips.

All rights reserved. Any form of duplication or distribution, including excerpts, is only permitted with express authorization by SIEMENS.

SIMATIC
S7-200 Примеры

Группа	Пример к теме
1	Einsatz von Micro/DOS auf dem HP-Palmtop

Краткое описание

Этот пример показывает, как подключить HP-Palmtop к PC и загрузить ПО Micro/DOS из PC вPalmtop. Затем следует описание подключения Palmtop к S7-200, а также загрузки программ Micro/DOS из Palmtop в ПЛК.

Схема включения

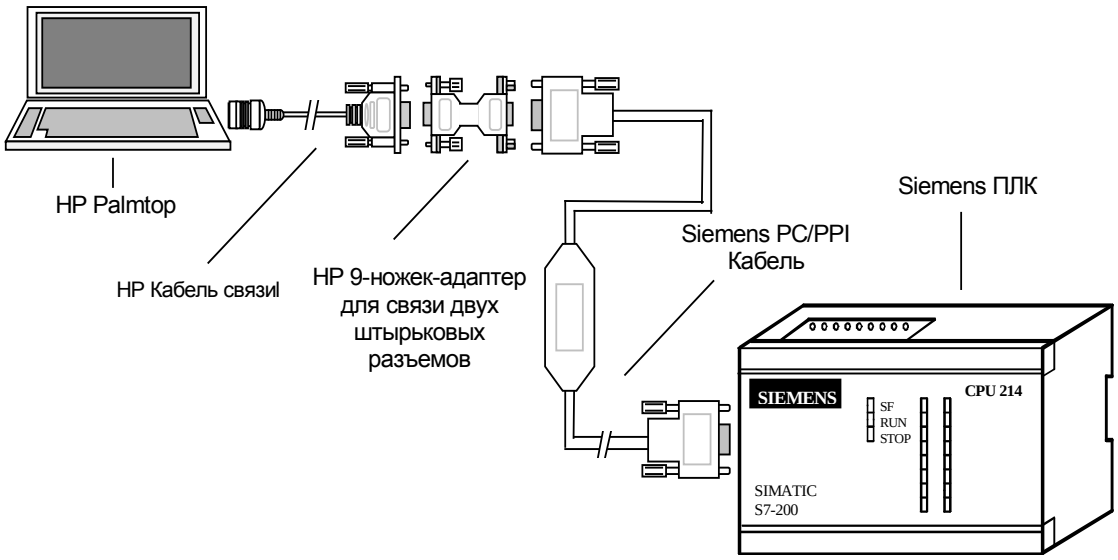


Рисунок 45.1

Требования к аппаратным средствам

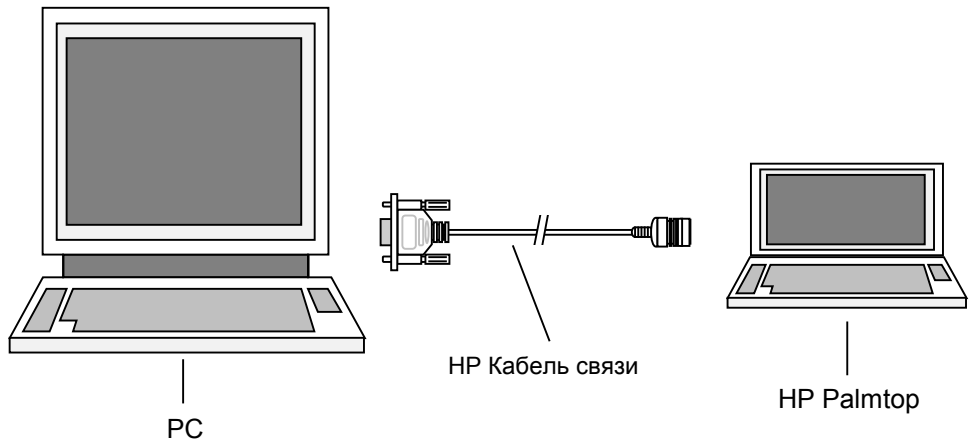
- Palmtop серии HP 100LX/200LX
- HP пакет подключения (HP F1021B):
    - 9-ножечный адаптер для связи двух штырьковых разъемов (HP 5181-6639)
    - 9 - 25 штырьковый соединитель модема папа/папа (HP 5181-6640)
    - PC-соединитель папа/папа (HP 5181-6641)
    - 9 - 25 штырьковый соединитель для принтера папа/папа (HP 5181-6642)
    - Кабель связи (HP F1015-80002)
    - Пакет подключения ПО (Версия 1.02)
- HP 5МБ Flash-диск (HP F1012 B)
- Siemens Micro/DOS
- Siemens PC/PPI Кабель
- (Опция) PCMCIA адаптер

Описание

Шаг 1: Установка Micro/DOS программы в HP-Palmtop

Есть две возможности, загрузить ПО Micro/DOS в HP-Palmtop. Простейший из них, это подключение к PC PCMCIA-адаптера. С помощью этого адаптера возможна передача файлов Micro/DOS из PC непосредственно на HP-Flash-диск в PCMCIA-адаптере, без дополнительных кабелей или адаптеров. После того как файл передан, HP-Flash-диск вынимается из PCMCIA адаптера и устанавливается в Palmtop. При этом Palmtop имеет полный доступ к файлам Micro/DOS.

Пакет подключения HP предлагает вторую возможность для передачи ПО Micro/DOS на HP-Flash-диск. PC имеет возможность обращаться к памяти Palmtop с помощью ПО, Кабеля и разъема, содержащихся в пакете подключения HP. Сначала инсталлируется ПО пакета подключения на PC (подробнее читайте в *HP Connectivity Pack Руководство пользователя*). Во время инсталляции программа предлагает Вам подключить компьютер к Palmtop с помощью кабеля связи.



После того как Вы проинсталлируете ПО, компьютер может связаться с помощью приложения Filer. Для того чтобы обратиться к Filer, введите следующее (из DOS или DOS-приложения с корректным дисководом/путем):

cd c:\cpack200
app200

Если ПО **app200** загружено, то выберите опцию Filer из графического окна. С помощью Filer (подробнее в: *HP Connectivity Pack Руководство пользователя*) директорий Micro/DOS будет передан на Flash-диск Palmtop.

После того как передан директорий, удостоверьтесь, что файл S7200.cfg (находится в директории Micro/DOS) имеет последнюю строку, приведенную в файле примера конфигурации:

ENABLE_KEYBOARD_INTERRUPT=TRUE

Чтобы ввести данную строку с помощью DOS-Editor сделайте следующее:

- удостоверьтесь, что Вы находитесь в директории Micro/DOS
- введите “edit S7200.cfg”
- после того как появится DOS-Editor, введите строку
- выберите “Сохранить” из “Файлы”-меню; затем выйдите из редактора.

Если данная строка не будет введена, то клавиатура в режиме статуса не будет реагировать или будет работать неправильно.

Файл S7200.cfg должен выглядеть примерно так:

S7200.cfg
mnemonic= Simatic
path=A:\PROGRAMS\SAMPLE language= DEUTSCH langset=DEUTSCH,ENGLISH,FRAN3AIS,ESPACOL,ITALIANO extension=A, B, C, D, E ENABLE_KEYBOARD_INTERRUPT=TRUE

Шаг 2: Подключение HP-Palmtop к Вашему ПЛК

После того как ПО инсталлировано на Flash-диск, можно подключать Palmtop к ПЛК. Для этого Вам нужно следующее: кабель связи, с помощью которого Palmtop подключается к PC; имеющийся в пакете подключения HP 9-ножечный адаптер для стыковки двух штырьковых разъемов и Siemens PC/PPI-кабель. Это необходимо, использовать предлагаемый HP 9-ножечный адаптер; он соединяет ножки так, чтобы обеспечить корректный протокол Мастер-Клиент между двумя устройствами. После того как PC/PPI кабель подсоединен к ПЛК, а кабель связи к Palmtop, оба кабеля подключаются к 9-ножечному адаптеру.

Если Ваш Palmtop подключен к ПЛК, то поддерживаются все функции Micro/DOS; Программы Micro/DOS могут теперь загружаться в ПЛК тем же способом, что и из PC.

Общие указания

Примеры SIMATIC S7-200 предоставляются заказчику бесплатно. Данные примеры не привязаны к конкретной задаче и являются общей информацией о возможностях применения S7-200. Решение заказчика может отличаться от приведённого здесь.

За правильную работу системы заказчик несёт ответственность сам. Мы обращаем Ваше внимание на действующие нормы Вашей страны и предписания по установке соответствующей системы. Ошибки и изменения возможны.

SIMATIC
S7-200 Примеры

Группа	Тема
6	**Измерение температуры при помощи цифровых входов**

Краткое описание

В данном примере описано как можно измерять температуру используя два цифровых входа. Модуль времени температуры каждую секунду в течении 18 ms выдает последовательно значение температуры в двоично-десятичном коде (BCD). Используя дополнительную схему усиления импульсы (13 в течении 18 ms) подаются на входы S7-200.

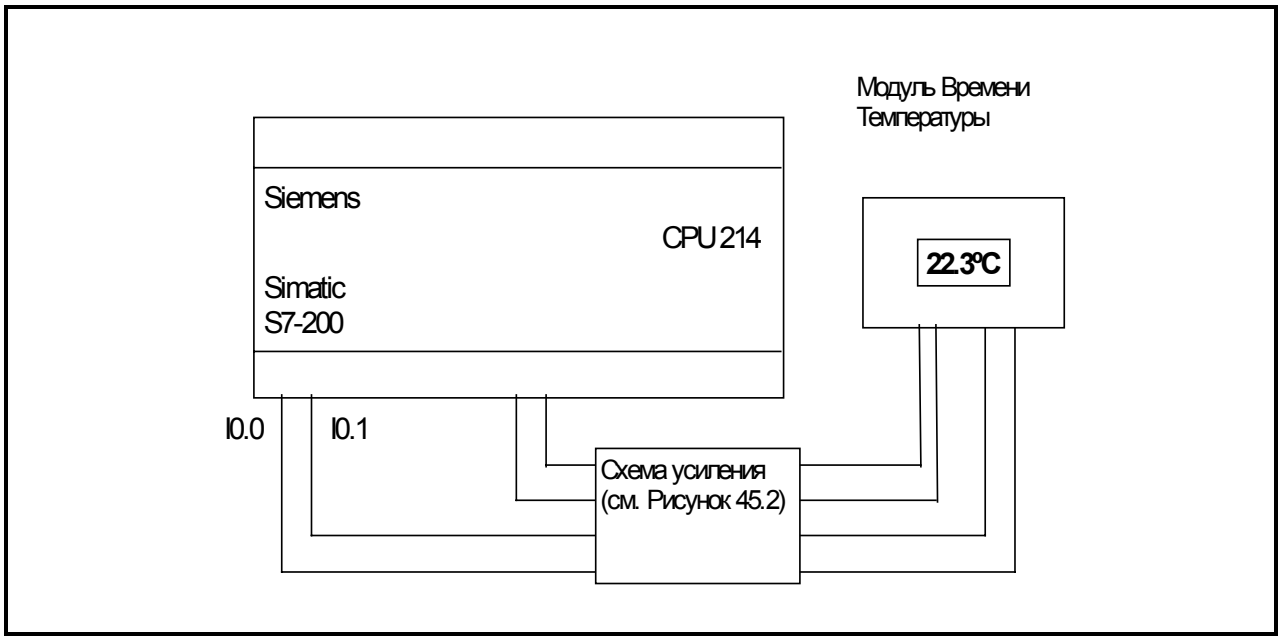


Рисунок 45.1 Подключение Модуля Времени Температуры к S7-200 PLC

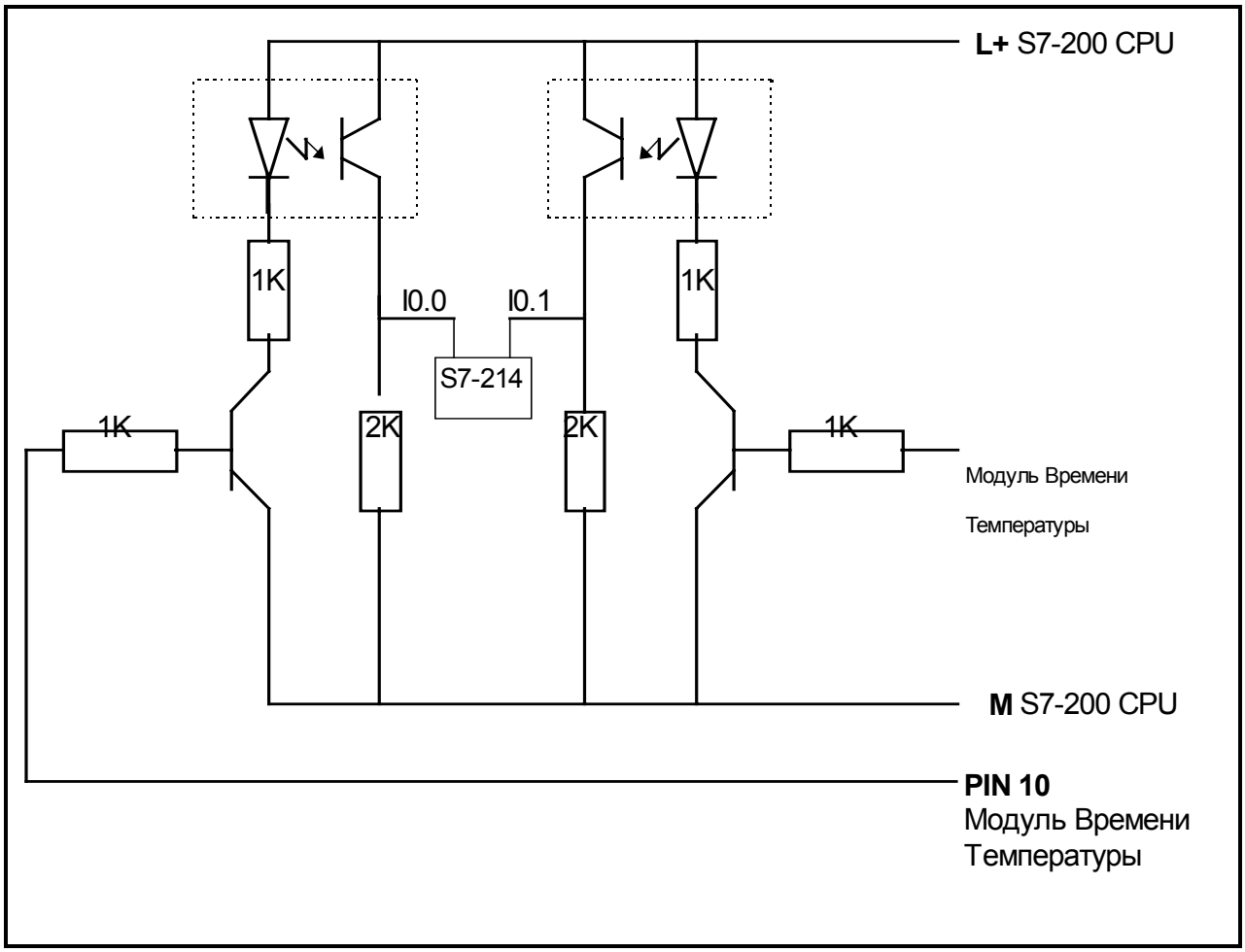


Рисунок 45.2 Структура Схемы Усиления

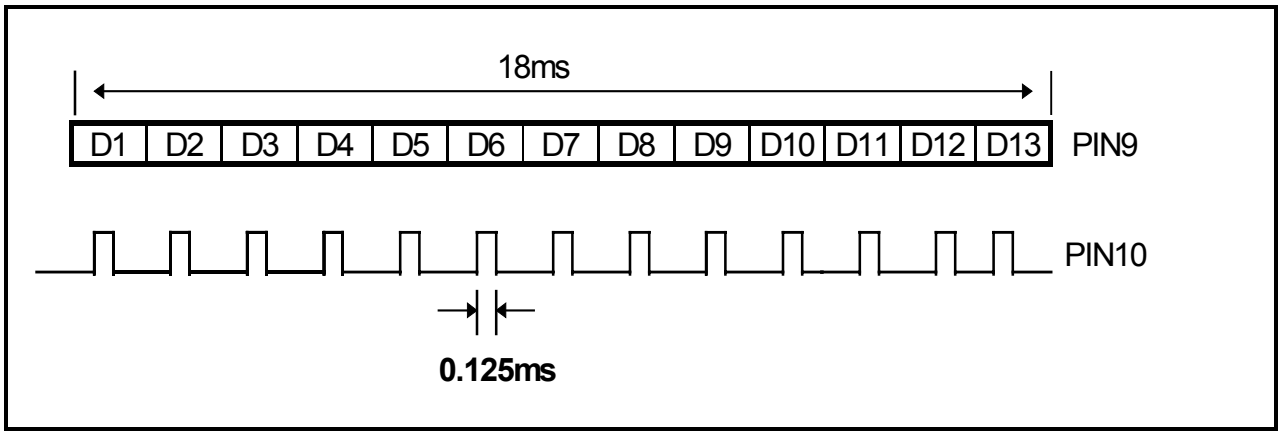


Рисунок 45.3 Временная диаграмма импульсов на выходе Модуля Времени Температуры

Замечание: Входы S7-200 гарантируют регистрацию импульсов продолжительностью не менее 0.2ms (см. Рисунок 45.4). Как видно на Рисунке 45.2, продолжительность импульсов на выходе модуля времени температуры равна 0.125ms. Поэтому необходимо использовать оптопару в схеме усиления, которая выдает более медленные импульсы, чем поступают на нее. Необходима такая оптопара, которая читает импульсы длиной 0.125ms и выдает импульсы длиной по крайней мере 0.2ms. Так же импульсы, выдаваемые с оптопары, д.б. меньше 0.7ms, т.к. они д.б. переданы до того, как модуль времени температуры выдаст следующий импульс. Таким образом, диапазон между временем срабатывания оптопары и временем выключения д.б. между 0.075ms и 0.675ms.

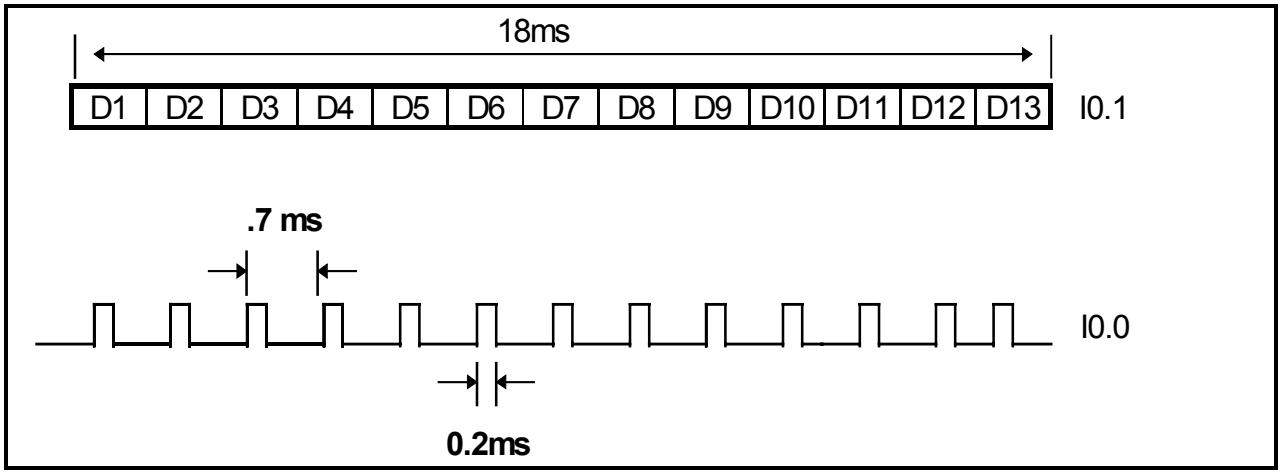


Рисунок 45.4 Временная диаграмма импульсов на входах ПЛК

Аппаратные компоненты

1	Simatic CPU 214
1	Цифровой Модуль Времени Температуры Диапазон измерения от -20 до +70 °C, Точность 0.1 °C, Погрешность ± 1 °C Производитель: Conrad Electronic GmbH.
2	Оптопары 4N 28/SU25
2	Транзистора BC 109 C
4	Резистора 1 кОм
2	Резистора 2 кОм

Замечание:
- Схема может работать только с CPU 214. Скорость обработки S7-212 - недостаточна.
- Небольшая программа и простая схема усиления делают возможным быструю и несложную реализацию.
- Благодаря своим небольшим размерам (также монтируется на DIN профиле) LCD модуль времени температуры, непосредственно отображающий температуру, м.б. использован где угодно.
- Температура регистрируется при помощи внешнего датчика температуры.
- С использованием мультиплексора (управляемого через DOs), м.б. легко выбраны несколько точек измерения для записи температуры с любых мест.
- Сопротивление линии (контура) датчика д.б. не более 30 Ом.

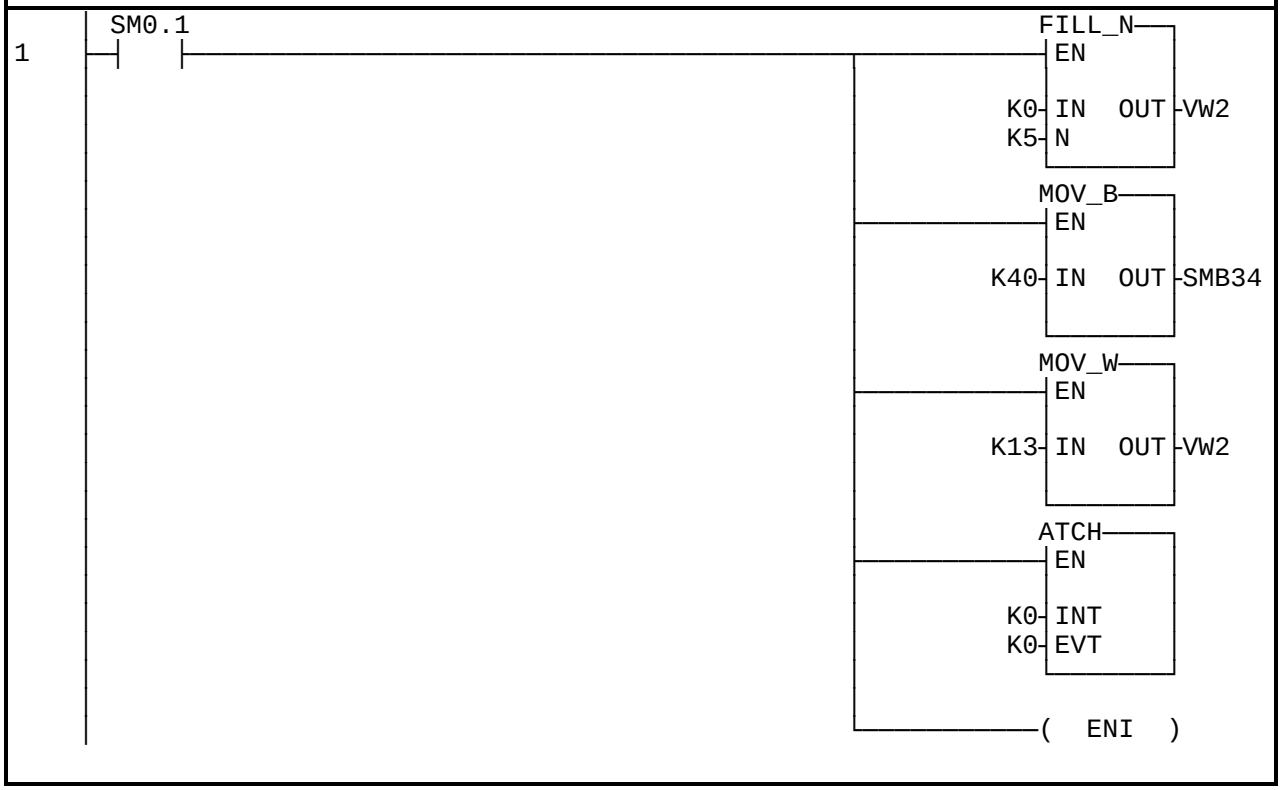
Структура программы и описание

Главная программа -
Главная программа инициализирует все области памяти в 0, а счетчик (VW2) в 13. Так же событию 0 назначается прерывание INT 0. Для задания времени для INT 1 равным 40ms в SMB34 заносится значение 40.

Прерывание 0 -
Программа обработки прерывания INT 0 вызывается при каждом положительном фронте на входе I0.0. Если счетчик равен 13, программа назначает INT 1 Событию 10 и уменьшает на 1 счетчик импульсов - слово VW2. По поступлению импульса, позиции записываются в двоично-десятичном коде. Затем записанное BCD значение преобразуется в целое для выдачи.

Прерывание 1 -
Если в течении 40ms не были прочтены 13 импульсов, вызывается INT 1. Она сбрасывает счетчик в 13 и очищает области памяти для ипульсов.

Главная программа



```
LD      SM0.1                          // Бит первого цикла
FILL    0       VW2     5              // Инициализация областей памяти
MOVB    40      SMB34                  // Задание времени вызова INT 1 = 40ms
MOVW    13      VW2                    // Инициализация счетчика в 13 импульсов
ATCH    0       0                      // Назначить Прерывание 0 Событию 0
ENI                                    // Разрешить Прерывание
```

```
2  |-------------------------------------------------------------( MEND  )
```

```
MEND                                   // Конец главной программы
```

Программы обработки прерываний

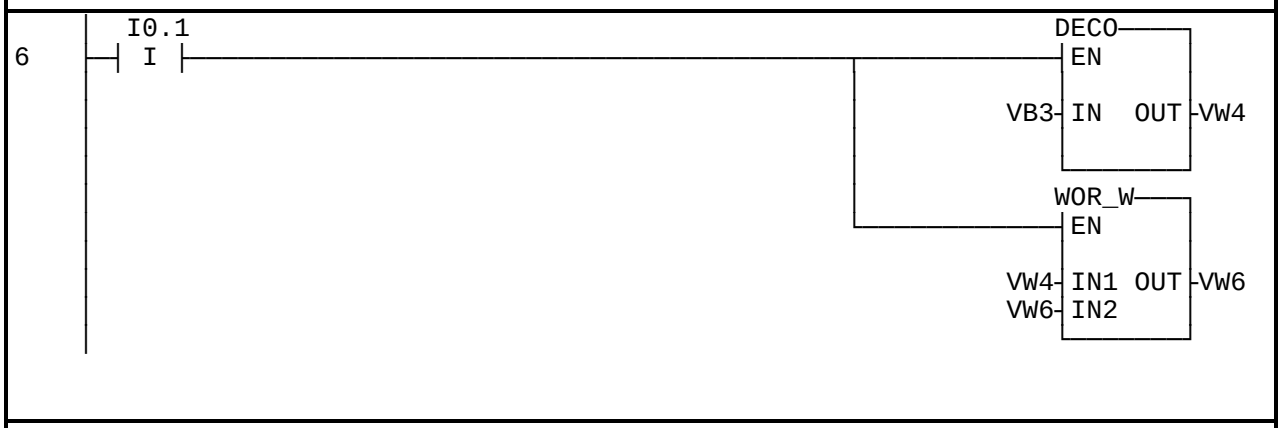
```
INT:  0
4  |   K13          VW2                                          ATCH
   |-------| == W |-------------------------------------------------| EN
   |                                                              |
   |                                                          K1- INT
   |                                                         K10- EVT
   |                                                              |
```

```
INT     0                              // Прерывание 0
LDW=    13      VW2                    // Если счетчик равен 13 импульсам,
ATCH    1       10                     // Назначить Прерывание 1 Событию 10
```

```
5  | SM0.0                                                        DEC_W
   |-|   |-------------------------------------------------------| EN
   |                                                              |
   |                                                         VW2- IN   OUT -VW2
   |                                                              |
```

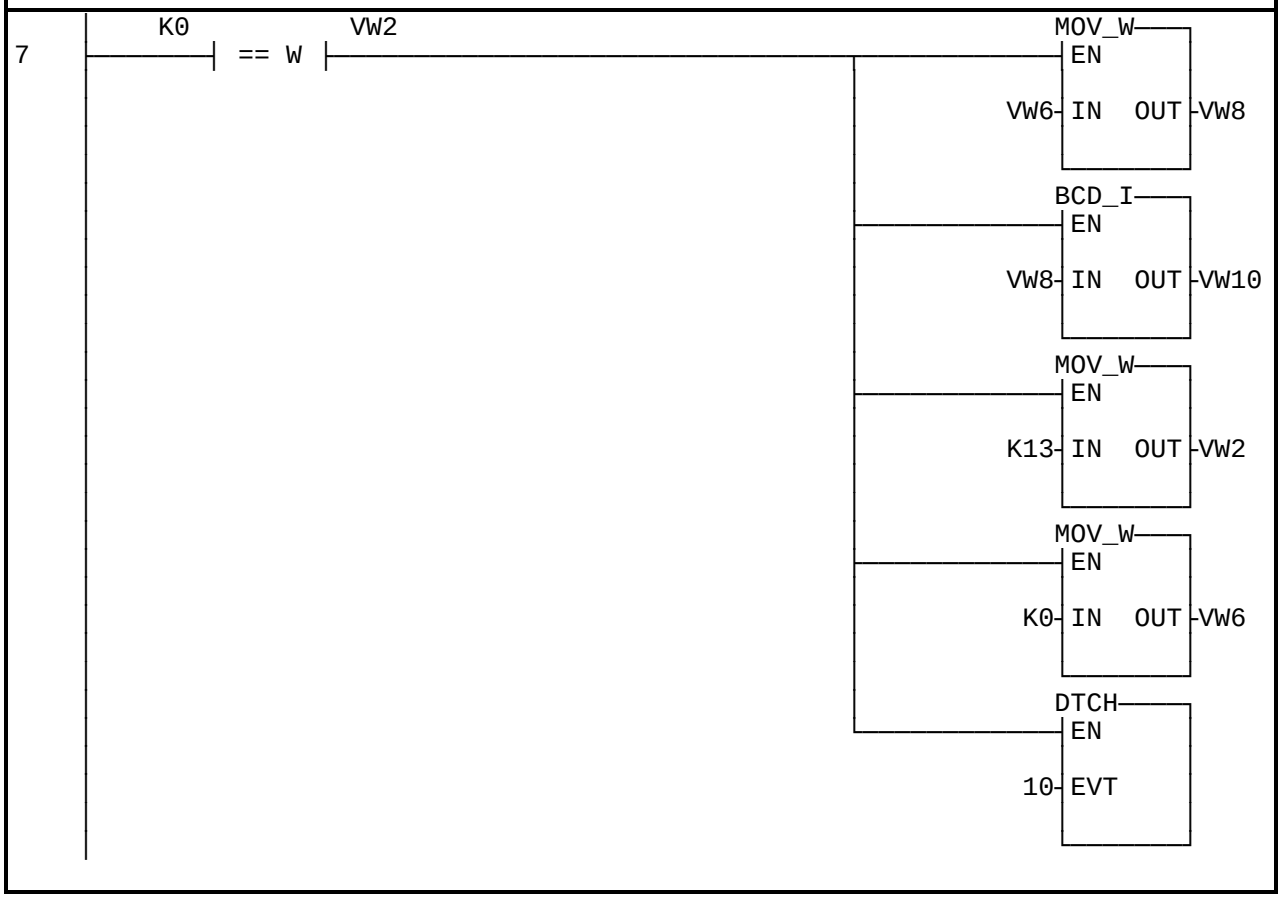
```
LD      SM0.0                          // Бит всегда "1"
DECW    VW2                            // Уменьшить счетчик на 1
```

Copyright ©1996 by SIEMENS

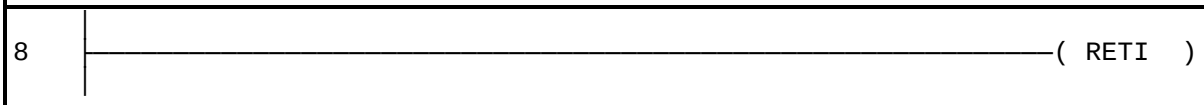


// Инструкция декодирования устанавливает бит в слове выходов (VW4), соответственно биту,
// выбранному в байте входов (VB3). Все остальные биты в VW4 устанавливаются в ноль.
// При помощи операции OR над VW4 и VW6 на каждом шаге новый бит назначается в VW6, и
// BCD значение записывается побитно в VW6. После приема всех 13 импульсов (VW2=0), это BCD
// значение записывается в VW8. Затем результат юудет преобразован в целое и помещен в VW10.
//

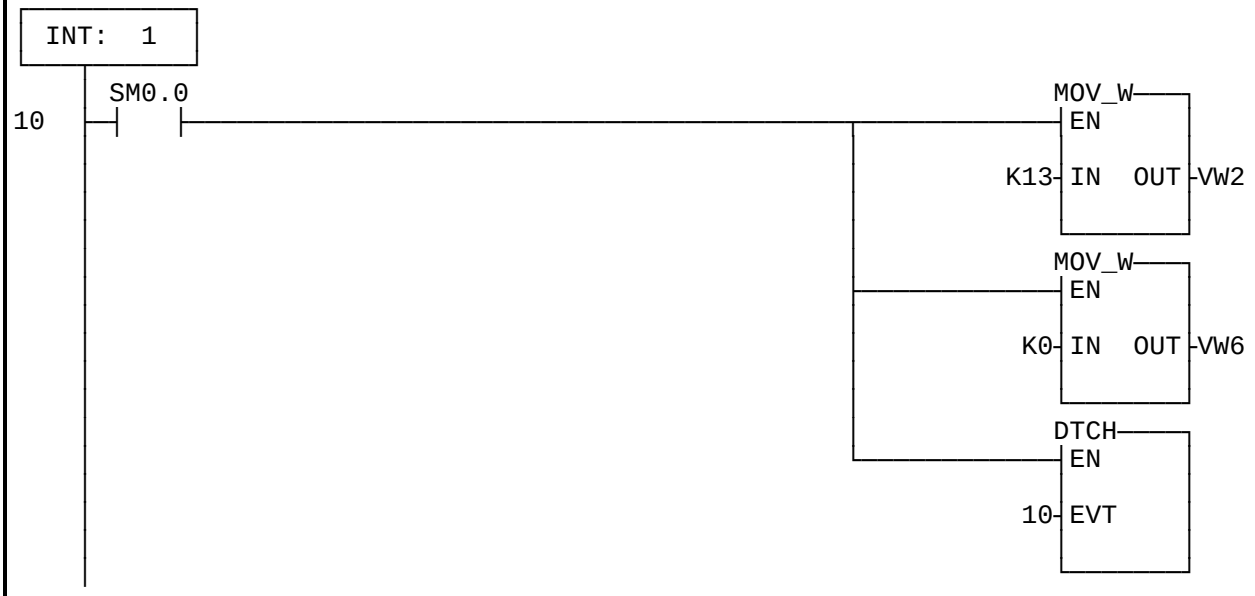
LDI	**I0.1**		**// Загрузить непосредственно I0.1**
DECO	**VB3**	**VW4**	**// Определить местоположение импульсов**
ORW	**VW4**	**VW6**	**// Сохранить местоположение в VW6**



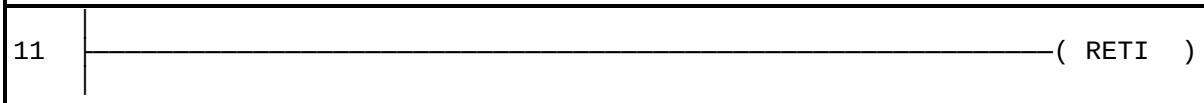
```
LDW=    0     VW2     // Если счетчик равен 0,
MOVW    VW6   VW8     // поместить BCD выход в VW8
MOVW    VW8   VW10    // Преобразовать BCD выход в целоеr, и
BCDI    VW10          // поместить значение в VW10
MOVW    13    VW2     // Сбросить счетчик после 13 импульсов
MOVW    0     VW6     // Очистить области памяти для BCD выхода
DTCH    10            // Снять прерывание с События 10
```




```
RETI                  // Конец программы прерывания 0
```




```
INT     1             // Прерывание 1
LD      SM0.0         // Бит всегда "1"
MOVW    13    VW2     // Сбросить счетчик после 13 импульсов
MOVW    0     VW6     // Очистить области памяти для BCD выхода
DTCH    10            // Снять прерывание с События 10
```




```
RETI                  // Конец программы прерывания 1
```

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажіый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Комментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены комментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все права защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC

Группа	Тема
1	**Использование нескольких термопар с ПЛК S7-200**

Краткое описание

Термопары м.б. подключены к ПЛК S7-200 используя аналоговый модуль EM235. Однако, использование термопар с ПЛК S7-200 не рекомендуется для любых применений, требующих значительной точности или линеаризации. Термопары при использовании с аналоговым модулем EM235 не обеспечивают большой точности и не д.б. использованы для приложений, которые требуют больше, чем просто значение темпертуры. Для приложений, требующих большей точности, следует использовать вместе с термопарой преобразователь в 4-20mA для термопары. Также, RTD обеспечивает большую точность по сравнению с термопарой. Использование RTD с ПЛК S7-200 объяснено в примере 47.

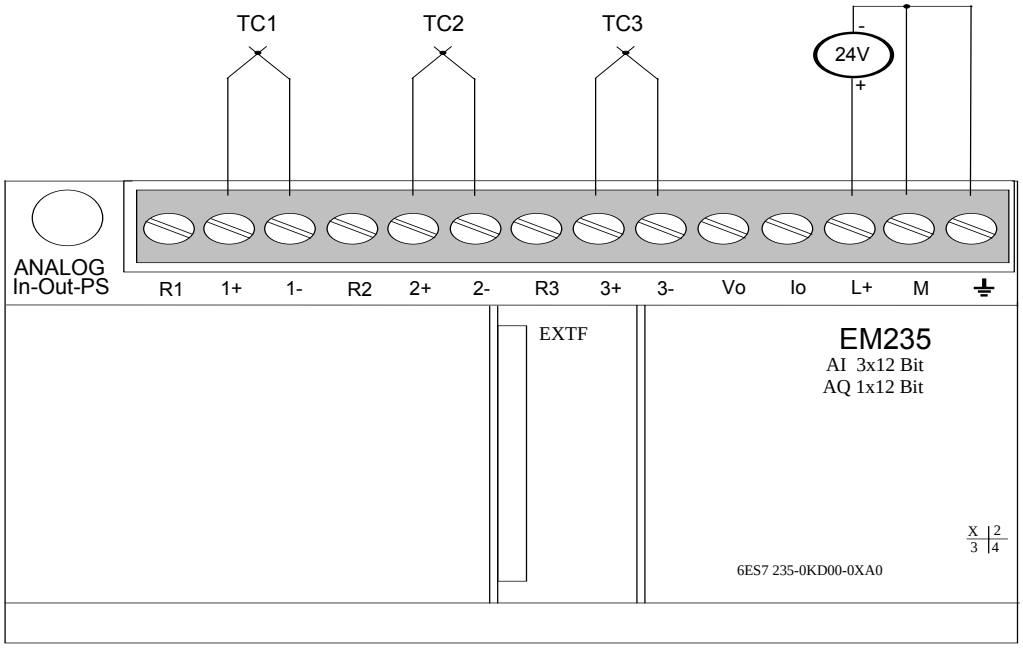


Рисунок 46.1 Подключение термопар к аналоговому модулю EM235

Описание

Входной диапазон аналогого модуля д.б. выбран для диапазона 50mV. Для этого необходимо установить dip переключатели на модуле в следующие положения:

1	3	5	7	9	11
OFF	ON	OFF	OFF	ON	OFF

Разрешение: 25(V
Повторяемость: (0.30%
Абсолютная точность: (1.0%

Для измерения термоэлектрического напряжения термопары требуется двухполярный входной диапазон. Считываемое значение соответствует напряжению вырабатываемому элементом термопары. Для преобразования его в температуру, пользователь должен использовать термоэлектрическую таблицу зависимости напряжения от температуры.

Настоятельно рекомендуется, чтобы пользователь калибровал ноль и максимальное значение входного напряжения к нулю и максимальному значению температуры термопары. Для этого пользователь должен поместить термопару в ванну с температурой 0(C (или установить симулятор термопары в ноль градусов) и откалибровать считывание ноля. Затем, термопару (или симулятор) следует поместить в максимальное значение и откалибровать считывание максимального значения. Таким образом откалиброваны входы для этих двух граничных точек. Предполагается, что термоэлектрическое напряжение будет линейное между двумя этими граничными точками. Без подобной калибровки, пользователю следует ожидать до 20(C ошибки.

Однако даже с подобной калибрацией пользователь должен ожидать значительное количество ошибок. Например, при использовании термопары типа K:

Пределы нижних и верхних ошибок 0(C = 2.2(C для самой термопары.

// рисунок общая ошибка

Схема ниже показывает как подсоединяются термопары к входным клемам модуля. Для подключения следует использовать витую пару, в экране, со специальными средствами против импульсных помех.

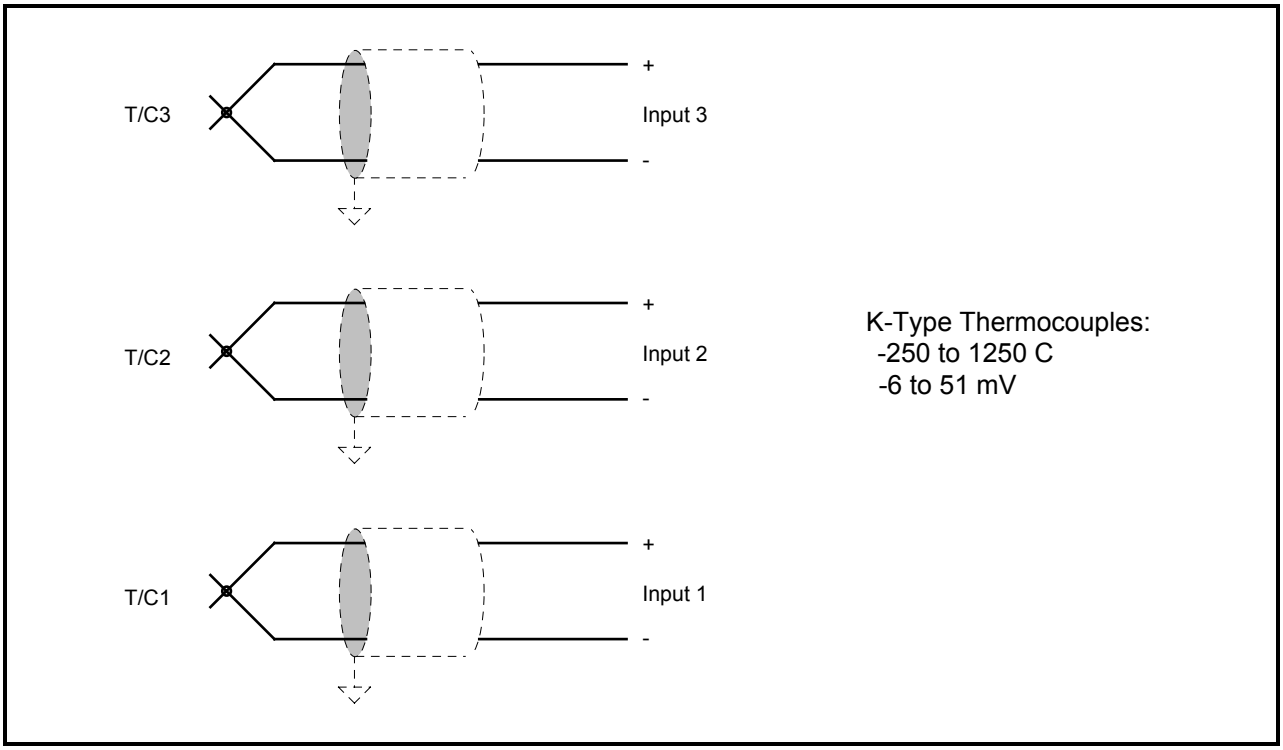


Рисунок 46.2 Подключение термопар к входным клемам модуля

General Notes

The SIMATIC S7-200 Application Tips are provided to give users of the S7-200 some indication as to how, from the view of programming technique, certain tasks can be solved with this controller. These instructions do not purport to cover all details or variations in equipment, nor do they provide for every possible contingency. Use of the S7-200 Application Tips is free.

Siemens reserves the right to make changes in specifications shown herein or make improvements at any time without notice or obligation. It does not relieve the user of responsibility to use sound practices in application, installation, operation, and maintenance of the equipment purchased. Should a conflict arise between the general information contained in this publication, the contents of drawings or supplementary material, or both, the latter shall take precedence.

Siemens is not liable, for whatever legal reason, for damages or personal injury resulting from the use of the application tips.

All rights reserved. Any form of duplication or distribution, including excerpts, is only permitted with express authorization by SIEMENS.

SIMATIC

S7-200 Примеры

Группа	Тема
1	**Использование нескольких термосопротивлений (RTD) с ПЛК S7-200**

Краткое описание

Термосопротивление (Resistance Temperature Detectors - RTD) м.б. использован с ПЛК S7-200 при подключении его аналоговому модулю EM235. Использование RTD вместе с ПЛК S7-200 дает более точный и линейный результат по сравнению с использованием термопар с ПЛК. Более подробную информацию об использованиии RTD с ПЛК S7-200 см. пример № 36 и 39.

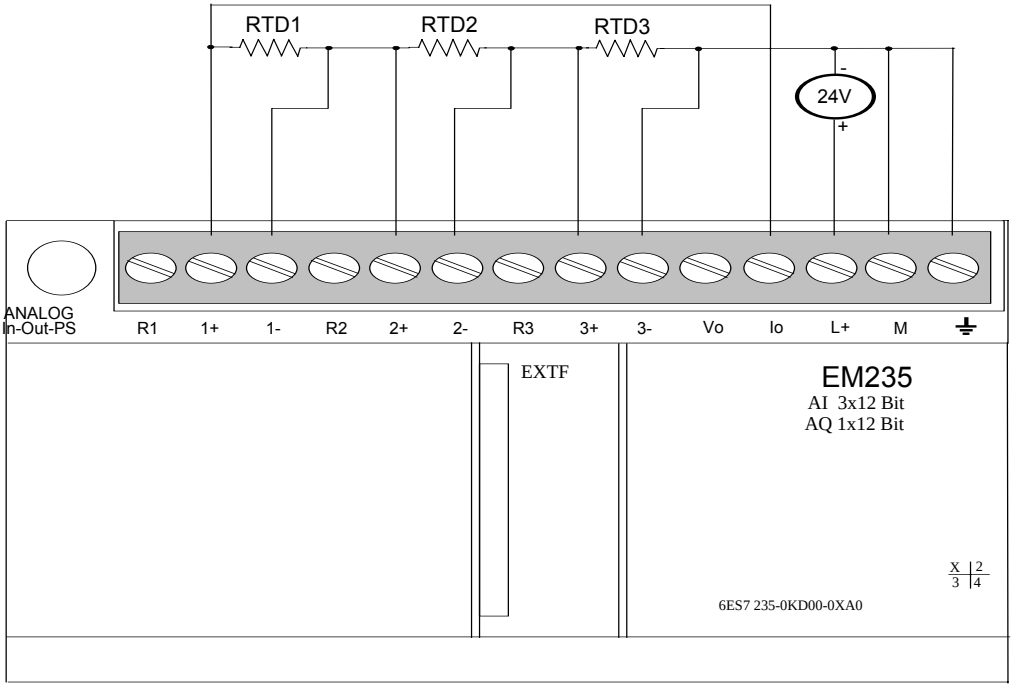


Рисунок 47.1 Подсоединение RTD к аналоговому модулю EM235

Описание

Входной диапазон аналогого модуля д.б. выбран для диапазона 50mV. Для этого необходимо установить dip переключатели на модуле в следующие положения:

1	3	5	7	9	11
ON	ON	OFF	ON	OFF	OFF

Разрешение: 12.5(V
Повторяемость: (0.25%
Абсолютная точность: (1.0%

Для измерения положительной или отрицательной температур достаточен однополярный диапазон. RTD никогда не генерирует отрицательное напряжение. Считываемое значение соответствует падению напряжения на RTD при восприятии температуры с возбуждением тока. Для преобразования его в температуру, пользователь должен использовать таблицу зависимости напряжения от сопротивления.

Максимальное разрешенное выходное (согласованное) напряжение = 12V, а максимальное разрешенное напряжение в любой точке = 10V. При превышении данных пределов, число ошибок существенно увеличивается.

Настоятельно рекомендуется, чтобы пользователь калибровал ноль и максимальное значение входного напряжения к нулю и максимальному значению температуры RTD. Для этого пользователь должен поместить RTD в ванну с температурой 0(C (или установить симулятор RTD в ноль градусов) и откалибровать считывание ноля. Затем, RTD (или симулятор) следует поместить в максимальное значение и откалибровать считывание максимального значения. Таким образом откалиброваны входы для этих двух граничных точек. Предполагается, что напряжение RTD будет линейным между двумя этими граничными точками. Оно более линейно, чем термоэлектрическое напряжение термопары.

// Ошибка самого аналогого модуля _____. Добавьте ошибку масштабирования RTD
// для получения общей ошибки.

Схема ниже показывает как подключаются RTD к входным контактам модуля и как источник тока подключается выходным контактам модуля. Данная конфигурация соответствует 4-проводному подключению. Для подключения следует использовать витую пару, в экране, со специальными средствами против импульсных помех.

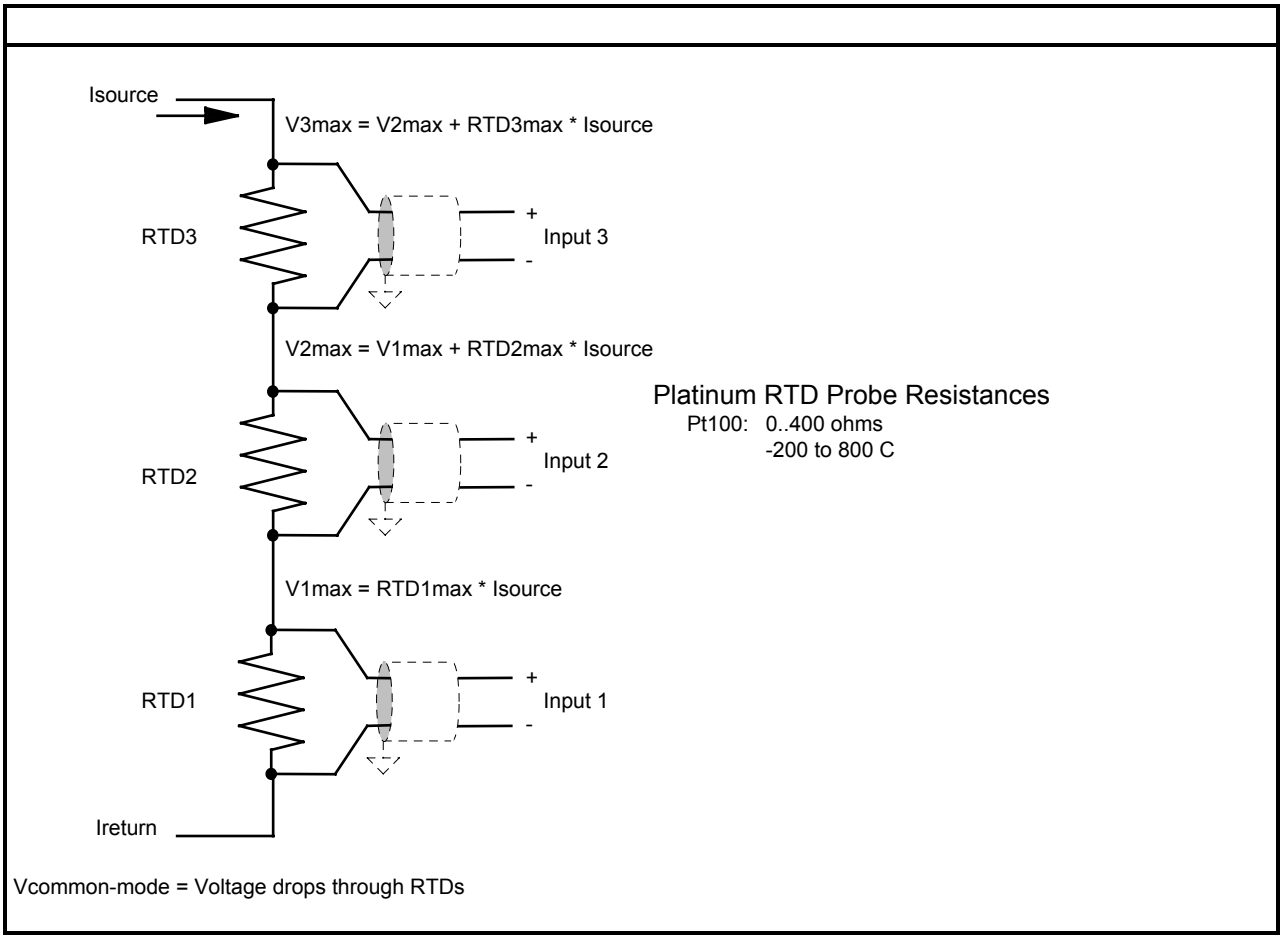


Рисунок 47.2 Подсоединение RTD к входным клемам модуля и подсоединение источника тока к выходным клемам

Пример

Если все из RTD в данном примере - это PT100, максимальное сопротивление любого из RTD = 400(. Если источник тока токового выхода = 10mA, то д.б. использованы следующие значения для $V1_{Max}$, $V2_{Max}$ и $V3_{Max}$ для всех RTD для максимального сопротивления, например, при температуре 800°C: $V1_{Max} = 4V$

$V2_{Max} = V1_{Max} + 4V = 8V$
$V3_{Max} = V2_{Max} + 4V = 12V$

Тогда V3 выходит за максимально разрешенное и выходной ток д.б. уменьшен с 10mA. Рекомендуется, чтобы токовый выход был установлен так, чтобы $I_{SOURCE} * (RTD_{1MAX} + RTD_{2MAX} + RTD_{3MAX})$ (10V.

Если температурный диапазон ограничен, например, 0°C - 200°C, то сопротивление, предполагая линейность, должно изменяться от ~ 80(до 160(. Измеряемый диапазон напряжения при токовом выходе 10mA будет:

0.8V (V1 (1.6V
1.6V (V2 (3.2V
2.4V (V3 (4.8V

Входной диапазон д.б. установлен = 0 - 5V.

При выходном токе = 1 mA, принятый диапазон напряжения д.б.:

80mv (V1 (160mv
160mv (V2 (320mv
240mv (V3 (480mv

Входной диапазон в данном случае д.б. установлен = 0 - 500mV.

General Notes

The SIMATIC S7-200 Application Tips are provided to give users of the S7-200 some indication as to how, from the view of programming technique, certain tasks can be solved with this controller. These instructions do not purport to cover all details or variations in equipment, nor to provide for every possible contingency. Use of the S7-200 Application Tips is free.

Siemens reserves the right to make changes in specifications shown herein or make improvements at any time without notice or obligation. It does not relieve the user of responsibility to use sound practices in application, installation, operation and maintenance of the equipment purchased. Should a conflict arise between the general information contained in this publication and the contents of drawings or supplementary material or both, the latter shall take precedence.

Siemens is not liable, for whatever legal reason, for damages or personal injury resulting from the use of the application tips.

All rights reserved. Any form of duplication or distribution, including excerpts, is only permitted with express authorization by SIEMENS.

SIMATIC
S7-200 Примеры

Группа	Тема
3	**Использование NETR и NETW**

Краткое описание

Этот пример объясняет использование команд чтения (NETR) и записи (NETWs) в сети для CPU 214, CPU 212 и TD 200. Простая программа использует сетевые команды для зажигания LEDs на одном CPU, отображающие входы другого, и для отображения значений каждого байта входов (IB0) и байта выходов (QB0) на TD200.

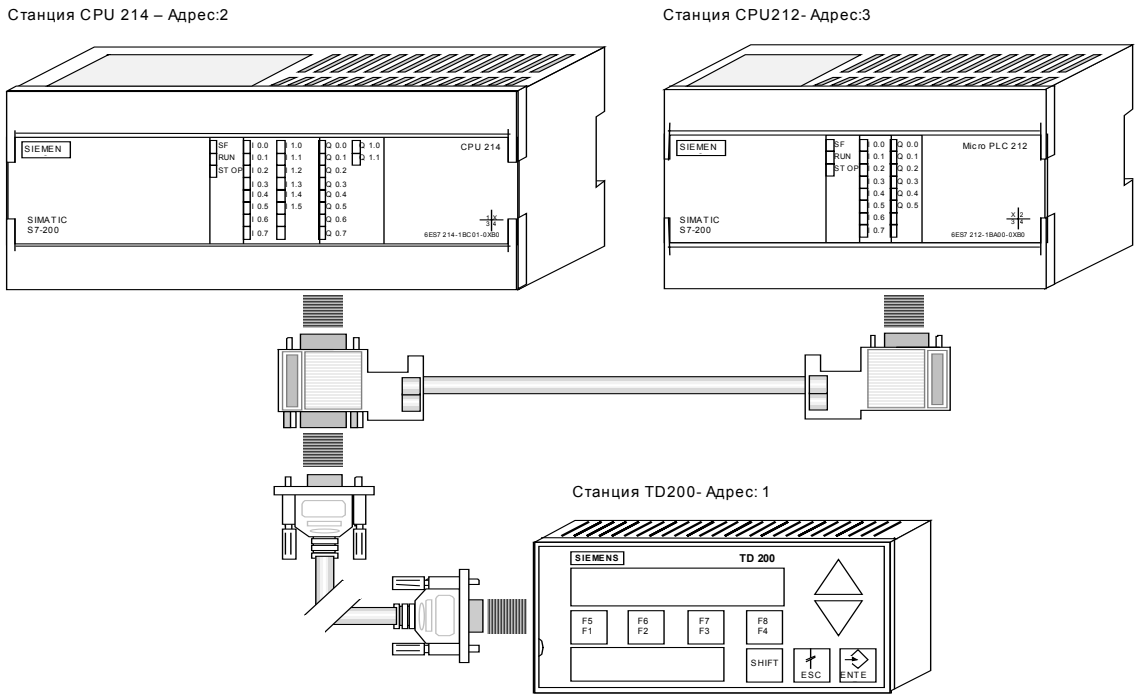


Рисунок 48.1 Конфигурация сети

Описание программы

Часть 1: Аппаратура

Этот пример показывает правильное использование команд NETR и NETW вместе с CPU 214, CPU 212 и TD 200. На Рисунке 48.1 показано структура сети для нашего примера.

Каждое устройство имеет изменяемый сетевой адрес для предотвращения коллизий при обмене данными. ПЛК имеет по умолчанию адрес 2, программатор - 0, а большинство других устройств (таких как TD200) - 1. Т.к. при использовании значений по умолчанию в локальной сети находятся две станции с адресом 2 (CPU 212 и CPU 214), Вы должны установить для одного ПЛК (например для 212) другой адрес. В нашем примере 214 имеет адрес 2, 212 - 3, а TD200 - 1. Вы можете изменить адрес станции используя или Micro/DOS или Micro/Win (Более подробную информацию Вы найдете в руководстве пользователя).

Во время выполнения нашей программы 214 ведет себя как активная станция; Micro/DOS и Micro/Win не поддерживают при программировании сеть с несколькими активными станциями (только, если Вы не используете карту MPI; в этом случае этот абзац не имеет значения). Без карты MPI Вы должны убрать все активные станции кроме одной (программатора) из сети для правильной работы программатора. Когда Вы подсоединяете программатор к 214, 212 илиTD 200, по крайней мере два устройства в сети думают, что они являются активными станциями: TD 200 и программатор. Когда 214 находится в состоянии RUN и установлен в режим PPI+, он ведет себя как еще одна активная станция и продолжает мешать программатору. Простейший способ избежать эту проблему - это отсоединить ПЛК, который Вы программируете от сети, перевести его в состояние STOP и подключить его напрямую к программатору (Рисунок 48.2).

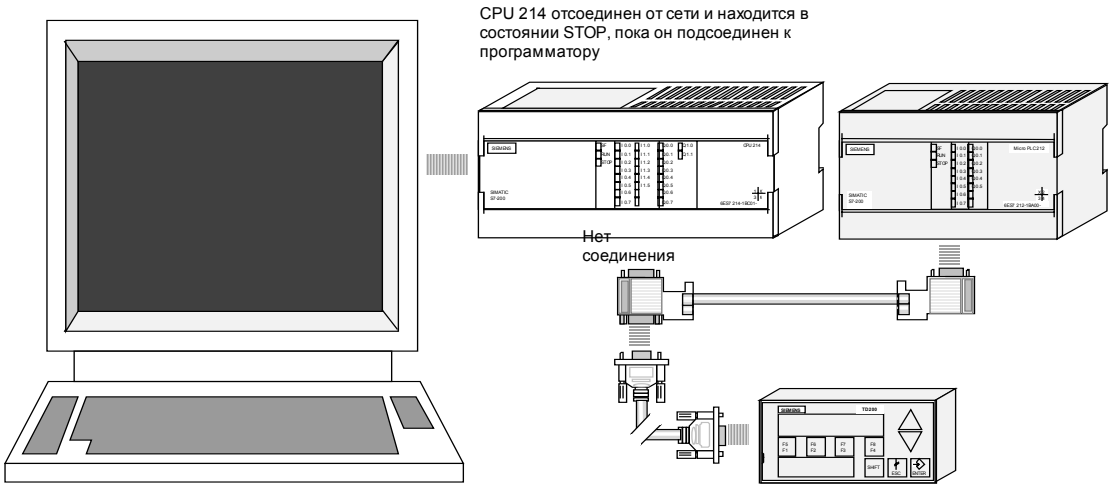


Рисунок 48.2 Возможное соединение с программатором

Если Вы не хотите отсоединять ПЛК от сети, Вы просто можете перевести ПЛК в состояние STOP и отсоединить TD 200 (Рисунок 48.3).

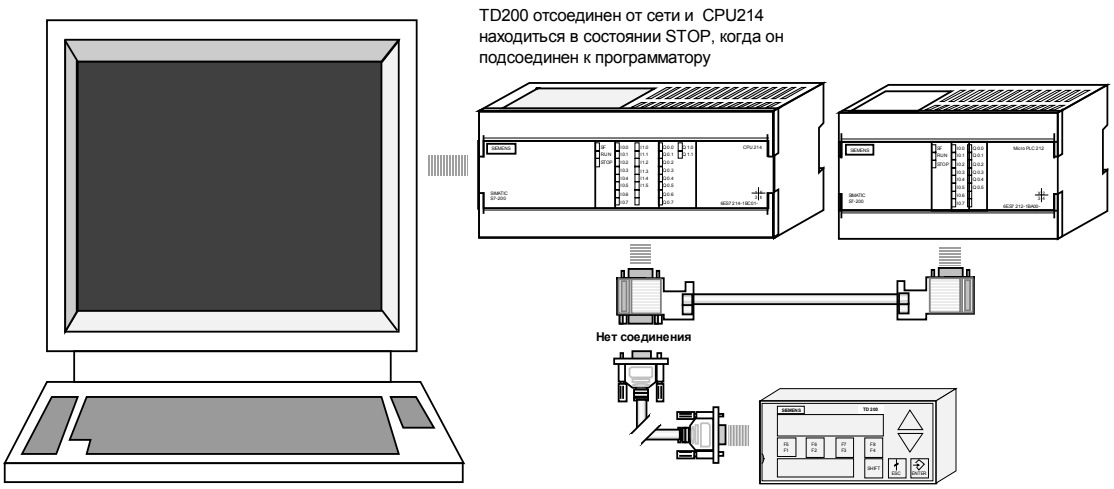


Figure 48.3 2 Другое возможное соединение с программатором

Как было сказано выше, в нашем примере мы будем использовать две сетевые команды CPU214: NETR и NETW (Network-Read и Network-Write). Каждая из этих команд требует два параметра: первый байт из таблицы (находящийся в V-памяти) и порт (который всегда имеет значение 0, т.к. CPU 214 имеет только один порт). Таблица (или буфер) состоит из 7 байт "заголовка" и от 1 до 16 байт "данных". На Рисунке 48.4 показана более подробная информация о различных составляющих таблицы.

Выполнено (функция завершена) : 0 = нет 1 = да
Активна (функция обрабатывается) : 0 = нет 1 = да
Ошибка (функция вернула ошибку) : 0 = нет 1 = да

Байт смещен.	7				0
0	B	A	O	0	Код ошибки
1	Адрес удаленной станции				
2	Указатель на Область данных Удаленной станции (I,Q,M or V)				
3					
4					
5					
6	Длина данных				
7	Бит данных 0				
8	Бит данных 1				
	...				
22	Бит данных 15				

Адрес удаленной станции: адрес ПЛК, к данным которого обращаются.

Указатель на область данных удаленной станции : косвенный указатель на адресуемые данные.

Длина данных : число байтов данных, адресуемых в удаленной станции (1 – 16 байт).

Область данных для приема или передачи: 1 – 16 байт, зарезервированных для данных. NETR записывает сюда полученные данные, а NETW записывает сюда данные для удаленной станции.

Рисунок 48.4 Таблица для NETR и NETW

Часть 2: Программа

Программа, включенная в данный пример (загружаемая в CPU 214), зажигает светодиоды на выходах CPU 212 соответственно входам CPU 214, и наоборот. Она также использует TD 200 для отображения значений IB0 и QB0 для обоих 214 и 212.

Во время первого цикла в первом сегменте выполняется несколько шагов. Во-первых, SMB30 устанавливается равным 2, устанавливается режим PPI+ и сообщается 214, что он является активной станцией. Во-вторых, очищается область памяти для трех таблиц (используемые NETR и NETW), которые создаются V200 (мы назовем ее Таблица 1), V250 (Таблица 2) и V300 (Таблица 3). Программа использует Таблицу 1 для адресации станции 3, чтения/записи IB0 и для данных, размером 1. Таблицы 2 и 3 тождественны Таблице 1, но обе для чтения/записи QB0.

Следующие три сегмента проверяют биты "обработано" для таблиц 1, 2 и 3 соответственно. Т.к. ни один из этих битов не устанавливается во время первого цикла, программа ничего не делает и переходит к сегменту 5, который использует SM0.0 для хранения отображаемых битов сообщений TD 200. Мы потом еще раз вернемся к сегментам 2 - 4.

Во время первого цикла в последнем сегменте устанавливается бит "обработано" для первой таблицы. На первый взгляд этот сегмент кажется не на правильном месте (Почему он не на первом месте в программе и откладывается обработка SM0.1 ?), но есть два довода, чтобы он находился здесь. Во-первых: Когда SMB30 устанавливается равным 2, чтобы работать в режиме PPI+ как активная станция, то потом до конца цикла не выполняется больше никакое действие. Таким образом, после установки SMB30 во время первого цикла, больше не происходит никаких изменений до второго цикла; также, после устанавки его во время двенадцатого цикла, не происходит никаких изменений до тринадцатого цикла. Во-вторых, команды NETR и NETW не работают, до тех пор пока SMB30 не воспримит изменения. Таким образом, если мы установили бит "обработано" в начале программы, при вызове NETW будет получена ошибка, и (конечно) не установиться бит "обработано" и программа будет находиться в режиме ожидания, наблюдая за битом "обработано" до перезапуска. Эта позиция позволяет программе опросить его, если это необходимо, до исполнения NETR и NETW.

Вы может быть удивлены всей этой игрой с битами"обработано". Идя такова: каждый раз, когда CPU 214 получает двойной запрос из сети, он помещает их в очередь. Чем больше очередь, тем менее регулярно выполняются команды NETR и NETW, и вся система работает медленнее. В нашем примере это избегается при помощи вышеупомянутого бита"обработано".

Когда выполняется команда NETR или NETW и адресуется какая-то таблица, то устанавливается бит "обрабатывается". Когда команда выполнена, бит "обрабатывается" очищается, а бит "обработано" устанавливается. Используя бит"обработано", мы играем с сегментами так, что в любое время активен только один сетевой запрос. Таким образом, когда бит "обработано" для 1 установлен, 2 - вызывается; когда для 2 - установлен, 3 - вызывается; когда для 3 - установлен, 1 - вызывается; и так далее (см. Рисунок 48.5). Конечно, в сегментах выполняется больше, чем было описано выше.

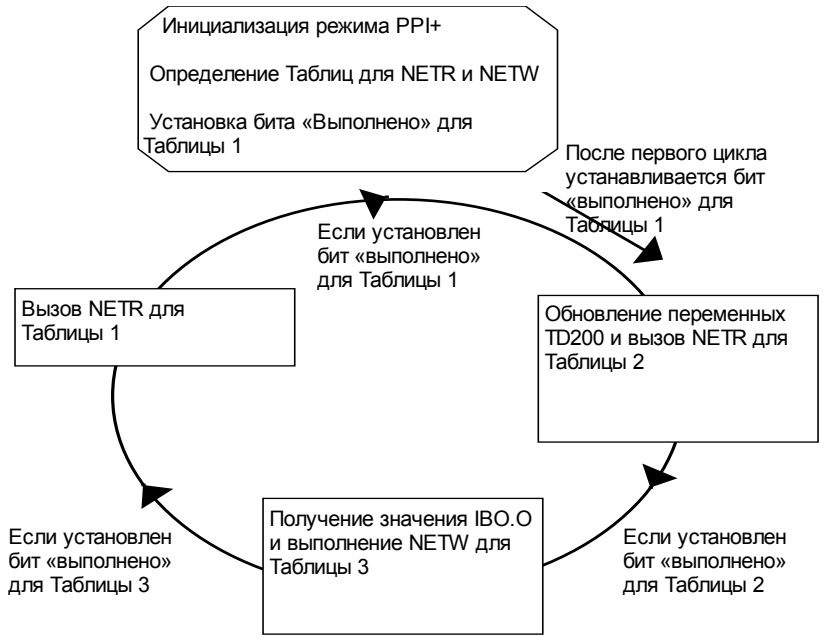


Рисунок 48.5 Диаграма состояний

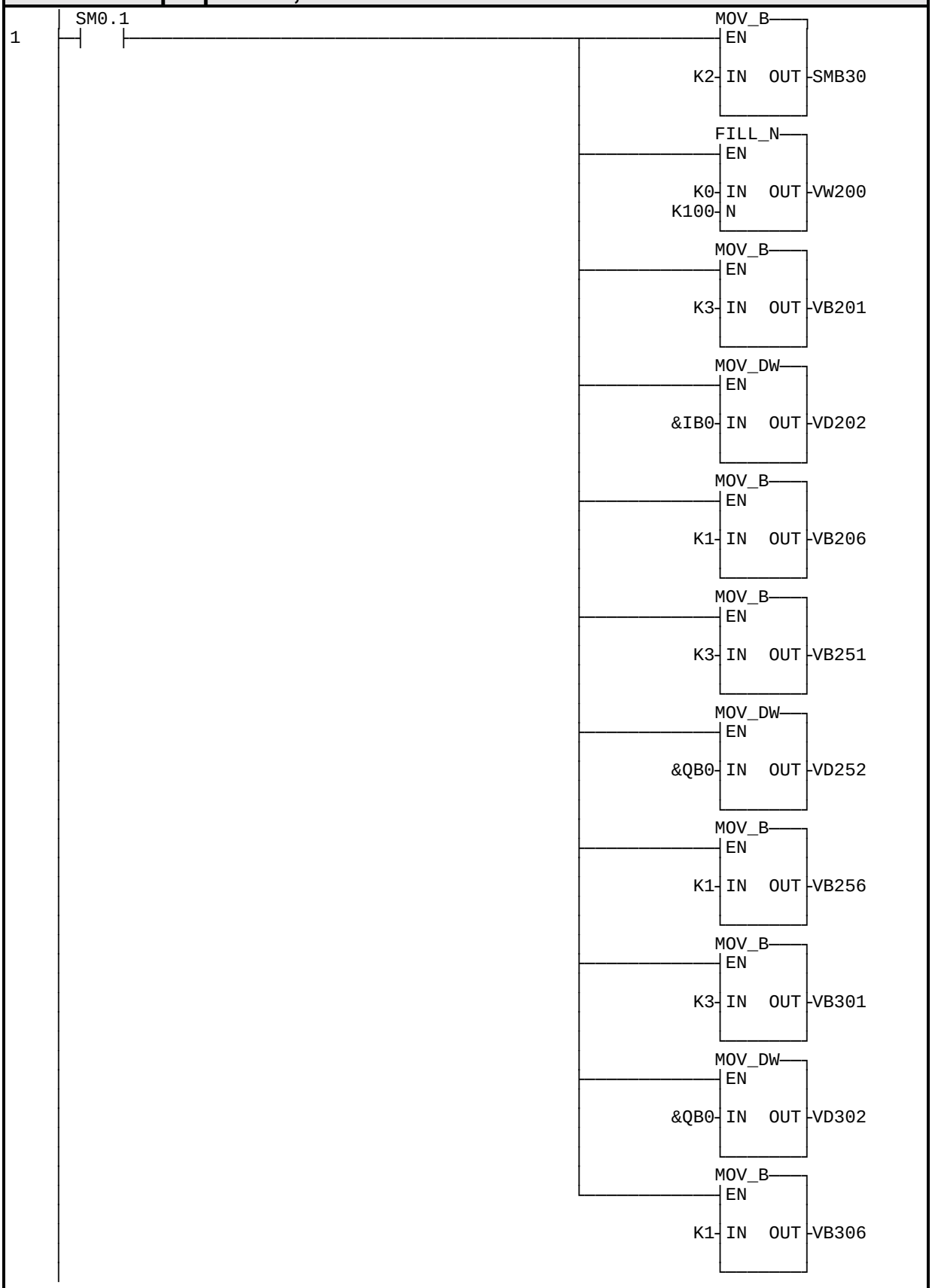
Если бит "обработано" для Таблицы 1 установлен, сегмент 2 копирует значение IB0 в область данных Таблицы 3 и вызывает NETW для Таблицы 3 записывая это значение в QB0 CPU 212.

Если установлен бит "обработано" для Таблицы 2, сегмент 3 берет значение из области данных Таблицы 1 (равное значению IB0 CPU 212) и записывает его в QB0 CPU 214 и в одну из переменных сообщений TD 200. Он также копирует IB0 и QB0 CPU 214 в переменные сообщений TD 200. И наконец он копирует данные из Таблицы 2 (значение QB0 CPU 212), завершая процесс обновления переменных сообщений TD200 перед вызовом NETR для Таблицы 2, чтобы получить новое значение QB0 CPU 212.

Если установлен бит "обработано" для Таблицы 3, вызывается NETR для Таблицы 1, чтобы получить значение IB0 CPU 212.

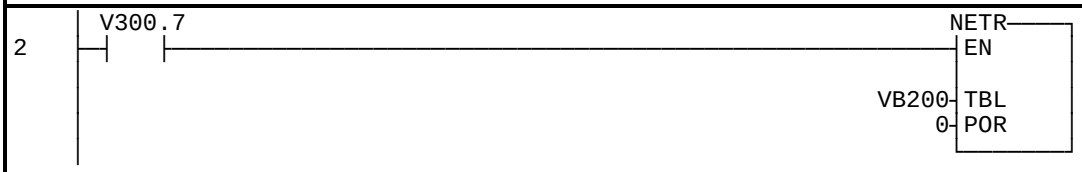
Более подробную информацию о сетевых командах Вы найдете в Главе 6, “Специальные команды” в *Руководстве по программированию S7-200*.

Описание программы, включая листинг

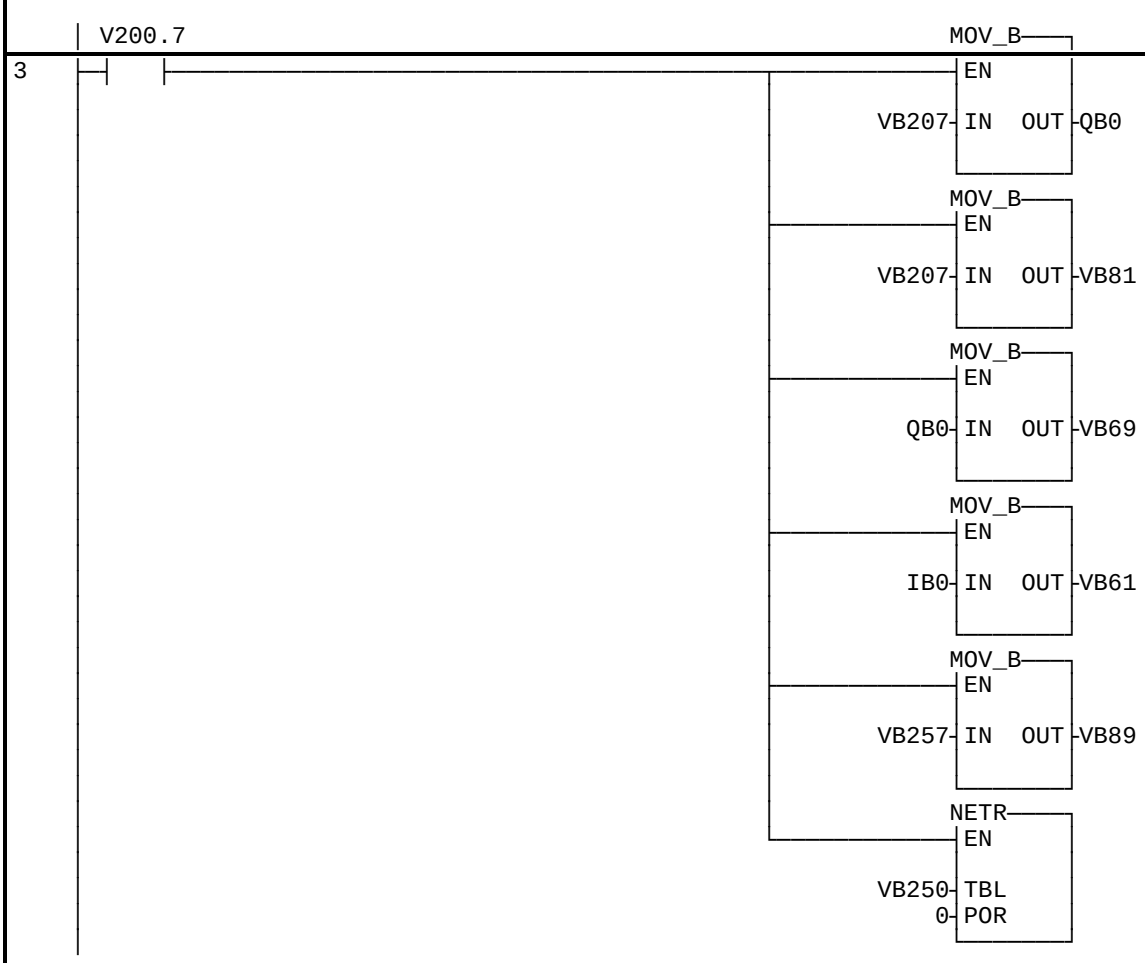


LD	**SM0.1**	**// Выполнить этот сегмент только во время первого цикла**
MOVB	**2, SMB30**	**// Установить CPU в режим PPI+**
FILL	**0, VW200, 100**	**// Очистить область памяти, используемую для таблиц**
		// * Установить Таблицу 1 @ VB200 (для NETR) *****

```
MOVB   3, VB201          // Адрес устройства = '3'
MOVD   &IB0, VD202       // Установить адрес для чтения = первому байту входов
MOVB   1, VB206          // Установить размер передачи = 1
                         //   *** Установить Таблицу 2 @ VB250 (для NETR) ***
MOVB   3, VB251          // Адрес устройства = '3'
MOVD   &QB0, VD252       // Установить адрес для чтения = первому байту выодов
MOVB   1, VB256          Установить размер передачи = 1
                         //   *** Установить Таблицу 3 @ VB300 (для NETW) ***
MOVB   3, VB301          // Адрес устройства = '3'
MOVD   &QB0, VD302       // Установить адрес для чтения = первому байту выодов
MOVB   1, VB306          // Установить размер передачи = 1
```

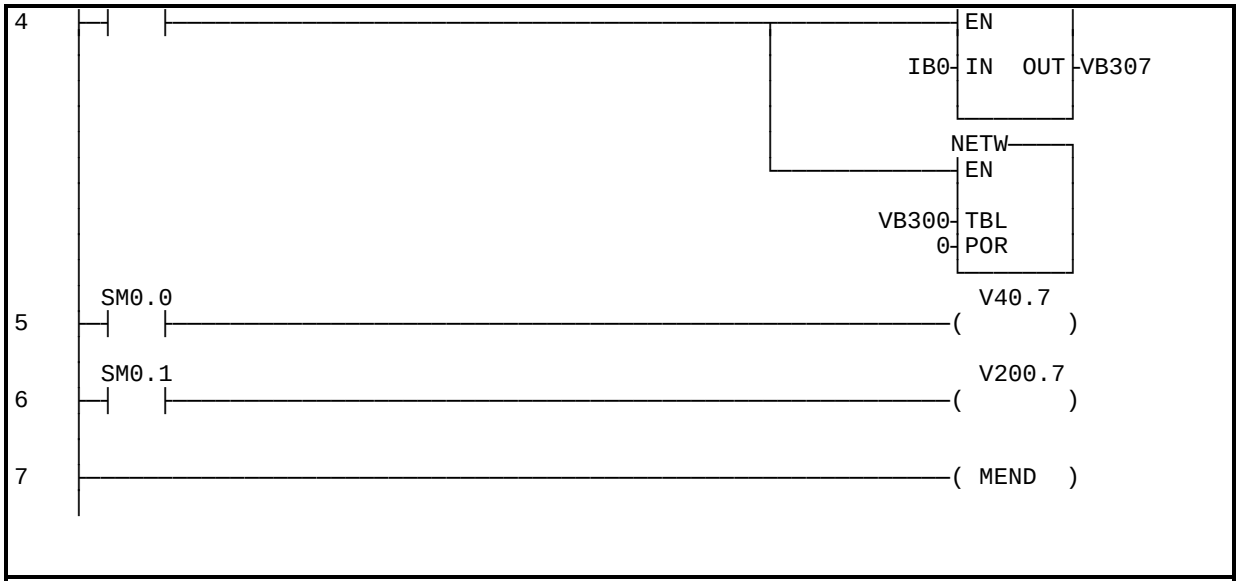



```
LD     V300.7            // Если закончена команда NETW для Таблицы 3,
NETR   VB200, 0          // Выполнить NETR для Таблицы 1
```




```
LD     V200.7            // Если закончена команда NETR для Таблицы 1
MOVB   VB207, QB0        // Передать полученное значение из Таблицы 1 в байт выходов
MOVB   VB207, VB81       // Передать полученное значение из Таблицы 1 на экран TD200
MOVB   QB0, VB69         // Передать значение байта выходов в TD200
MOVB   IB0, VB61         // Передать значение байта входов в TD200
MOVB   VB257, VB89       // Передать полученное значение из Таблицы 2 в TD200
NETR   VB250, 0          // Вызвать NETR для Таблицы 2
```

V250.7 MOV_B




```
LD      V250.7          // Если закончена команда NETR для Таблицы 2
MOVB    IB0, VB307      // Передать значение байта входов в Таблицу 3
NETW    VB300, 0        // Вызвать NETW для Таблицы 3
                        // для записи IB0 в пассивную станцию

LD      SM0.0           // Для отображения сообщений на TD200
=       V40.7           // всегда использовать один бит

LD      SM0.1           // Только во время первого цикла
=       V200.7          // установить бит "обработано" для Таблицы 1

MEND                    // Конец
```

Блок данных

```
VB0     'TD'            // Начало блока параметров TD200
VW2     16#1001         // Английский; обновлять как можно быстрее
VW4     16#0100         // 1 сообщение
VW6     50              // Сообщения начинаются с VB50
VW8     40              // Разрешения начиная с VB40
VB50    '214: I0='      // Часть сообщения
VW58    16#0030         // Конфигурация переменных
VW60    16#0000         // Значение переменных
VB62    ' Q0='          // Следующие данные сообщений
VW66    16#1030         // Конфигурация переменных
VW68    16#0000         // Значение переменных
VB70    '212: I0='      // Следующие данные сообщений
VW78    16#0030         // Конфигурация переменных
VW80    16#0000         // Значение переменных
VB82    ' Q0='          // Следующие данные сообщений
VW86    16#1030         // Конфигурация переменных
VW88    16#0000         // Значение переменных
```

Указания по преобразованию

Для того чтобы преобразовать IEC STL в S7-Micro/DOS STL

- Добавьте ‘K’ перед каждым числом, не являющимся шестнадцатеричной константой (например, 4 ⇒ K4)
- Замените ‘16#’ на ‘KH’ для всех шестнадцатеричных констант (например, 16#FF ⇒ KHFF)
- Поставьте запятые для смены полей. Используйте клавиши перемещения или клавишу TAB для перехода от поля к полю.
- Для преобразования программы S7-Micro/DOS STL в LAD-форму каждый сегмент должен начинаться со слова ‘NETWORK’ и номера. Кажiый сегмент в этом примере имеет свой номер на диаграмме LAD. Используйте команду INSNW в меню редактора для ввода нового сегмента. Команды MEND, RET, RETI, LBL, SBR и INT требуют отдельных сегментов.
- Коментарии строк, обозначенные "//" не поддерживаются в S7-Micro/DOS, но разрешены коментарии сегментов

Общие указания

Примеры применения SIMATIC S7-200 предназначены для того, чтобы дать пользователям S7-200 начальную информацию, как можно решить с помощью данной системы управления определенные задачи. Данные примеры применения S7-200 бесплатны.

В приведенных примерах программ речь идет об идеях решения без претензии на полноту или работоспособность в будущих версиях программного обеспечения S7-200 или STEP7 Micro. Для соблючения соответствующих технически безопасных предписаний при применении необходимо предпринять дополнительные меры.

Ответственность Siemens, все равно по каким правовым нормам, при возникновении ущерба из-за применеия примеров программ исключается, равно и при ущербе личным вещам, персональном ущербе или при намеренных или грубо неосторожных действиях.

Все прав защищены. Любая форма размножение и дальнейшего рапространения, в том числе и частично, допустимо только с письменного разрешение SIEMENS AG.

SIMATIC S7-200 Примеры

Группа	Тема
3	**Использование модема для удаленного программирования и мониторинга ПЛК S7-200 через PPI интерфейс**

CPU необходимые для данного примера					
CPU 210 (	CPU 212 (	CPU 214 (	CPU 215 (	CPU 216 (	ДРУГИЕ (

Краткое описание

В данном примере описывается использование модемов с ПЛК SIMATIC S7-200 через PPI интерфейс. Модемы соединяют S7-200 CPU с удаленным PC или PG. Это позволяет пользователю программировать или контролировать удаленный ПЛК используя Micro/WIN или Micro/DOS. Обратите внимание, что ПЛК S7-200 требует, чтобы модем использовал 11 бит данных для PPI обмена, а стандартный Hayes совместимый модем обычно использует 10 бит данных. Более подробную информацию об использовании Hayes модемов с S7-200 CPU в режиме свободнопрограммируемого интерфейса см. пример № 26.

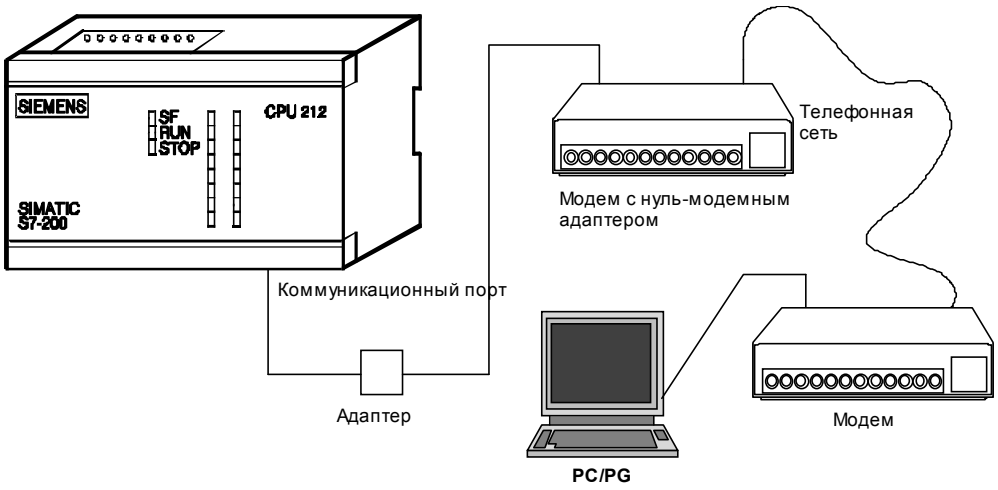


Рисунок 49.1 - Сеть из S7-200, модемов и ПК

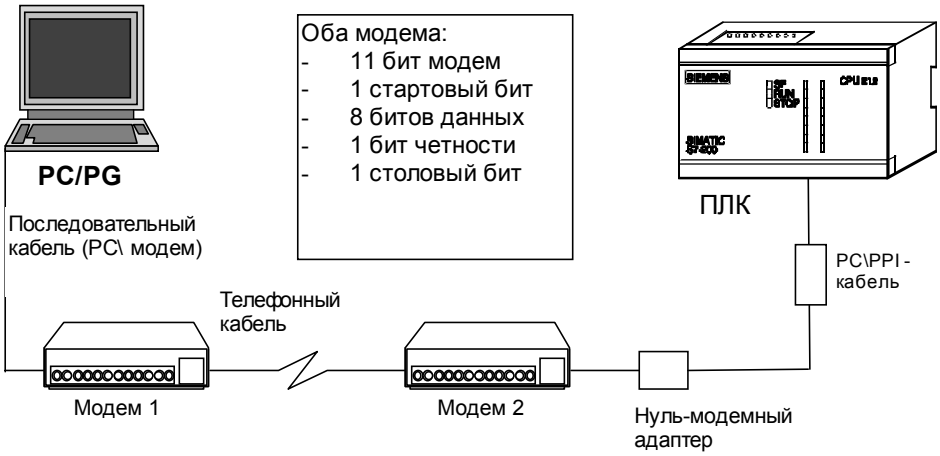


Рисунок 49.2 Структура соединений

Рисунок 49.3 - Разводка нуль-модемного адаптера

Важное замечание

10 битовые модемы не подходят. Модемы <u>должны</u> поддерживать 8 бит данных с четностью.

Аппаратные требования

1 SIMATIC S7-200 (CPU 212, 214, 215 или 216)
1 Siemens PC\PPI кабель
1 соответствующий интерфейс (см. рисунок 49.3 - в зависимости от разъема модема, обычно это нульмодемный кабель 9-контактов "мамма" на 25 контактов "мама" с перекрестием приемной и передающих линий.)
1 Модем Multi Tech (MultiModemZDX MT1932ZDX или MT2834ZDX), 9600 Бод
1 Терминальная программа для конфигурирования модема, которая не изменяет состояние модема при ее закрытии (модем остается в online, при выходе из программы)

Описание

Для обмена по PPI интерфейсу ПЛК S7-200 необходим модем, использующий 11 бит data string. ПЛК S7-200 требует для PPI: 1 Стартовый Бит, 8 Битов Данных, 1 Бит Четности, 1 Стоповый Бит, асинхронный обмен и 9600 Бод. Многие распространенные модемы не способны работать с такими параметрами.

В данном примере используются заводские установки по умолчанию модема Multi Tech за исключением некоторых, описанных в настройке модема.

Для подключения компьютера к модему используйте прямой кабель. Со стороны ПЛК д.б. использован стандартный кабель для программирования S7-200 (6ES7 901-3BF00-0XA0) с нуль-модемным адаптером (см. рисунок 49.3). Используйте терминальную программу модема для изменения настроек модема, затем вызовите удаленный модем: *ATDT 'номер, который д.б. набран'*. После приема CD (будет выведено Сообщение об обнаружении несущей), выйдите из программного пакета модема без разрыва соединения. Затем запустите Micro/DOS или Micro/WIN и работайте как обычно, но уже через теолефонную линию.

ЗАМЕЧАНИЕ:

- При попытке запустить Micro-DOS под Windows как приложение DOS могут возникнуть проблемы коммуникации.
- Большие задержки на телефонной линии могут вызвать ошибки превышения времени.
- ПК может изменить режим CPU 212 или 214 только, когда переключатель режима находиться в положении TERM. Если переключатель находится в положении TERM, то при включении-выключении питания все CPU автоматически переходят в режим STOP. В данном случае, необходимо, чтобы PG/ПК перевел CPU назад в режим RUN.
- Если переключатель режима находится в положении RUN, то при включении-выключении питания все CPU автоматически возвращаются в режим RUN. Однако, если преключатель режима на CPU находится в положении RUN, то возникают проблемы с изменением программы на удаленном расстоянии при использовании CPU 212 или 214 т.к. эти CPU д.б. в режиме STOP для загрузки новой программы. (ЗАМЕЧАНИЕ: Это не касается CPU 215 и 216 т.к. для них возможно изменение режима ПЛК в STOP для загрузки программы, даже если переключатель режима находиться в положении RUN.) См. следующую таблицу.

Удаленная загрузка разрешена

	Положение переключателя режима		
	STOP	TERM	RUN
CPU 212	Да	Да	Нет
CPU 214	Да	Да	Нет
CPU 215	Да	Да	Да
CPU 216	Да	Да	Да

Настройки модема:

При соединении модема с телефонной подсетью, наберите:

(например, внутри компании, имеющей телефонную подсистему, подключенную к общей сети):

AT &F0 X3 $EB1 $SB9600 $MB9600 &D0 &E4 &E12 &W0

При соединении модема с общей телефонной сетью, наберите:

AT &F0 X0 $EB1 $SB9600 $MB9600 &D0 &E4 &E12 &W0

Для модема, соединенного с ПЛК установите значение регистра для автоматического ответа после первого звонка

AT &S0=1

Проверьте настройки:

AT $H0 См. экран подсказки ... как показано:

B1 E1 M1 Q0 R0 V1 X0 &E1 **&E4** &E6 &E8 &E10 **&E12** &E15 $E0
%C0 #C1 &C1 **$MB9600 $SB9600** $BA0 &W0

$A0 &A0 #A0 &B0 &BS1 $D0 **&D0** %DL0 %DC0 **$EB1** %E1 %E4 #F2 $F1
&G0 #L0 &P0 &Q0 $R0 &R1 &RF1 &S1 &SF0 &T5 #T1 $T1 $V0 $V6 $VD0 Y0

для модема со стороны ПЛК:
S0=1 S2=43 S3=13 S4=10 S5=8 S6=2 S7=45 S8=2 S9=6 S10=7 S11=70
S13=37 S24=20 S25=0 S30=0

После этого, сохраните параметры в энергонезависимой памяти:

AT &W0

Описание необходимых Hayes-Команд:

&F0	активировать заводские установки
X3	тон звонка в подсистеме
X0	тон звонка в общей системе
$EB1	формат данных (8 данных, 1 стартовый, 1стоповый, 1 бит четности)
$SB9600	9600 Бод - скорость передачи модем <-> ПК
$MB9600	9600 Бод - скорость передачи на линии
&D0	игнорировать сигнал DTR
&E4 &E12	аппаратное квитирование установления связи
&W0	сохранить параметры в EEPROM модема
DT####...	тональный набор номера ####...
DP####...	импульсный набор номера ####...

Производитель модема: Multi-Tech Systems
612-631-3550
800-328-9717

General Notes

The SIMATIC S7-200 Application Tips are provided to give users of the S7-200 some indication as to how, from the view of programming technique, certain tasks can be solved with this controller. These instructions do not purport to cover all details or variations in equipment, nor do they provide for every possible contingency. Use of the S7-200 Application Tips is free.

Siemens reserves the right to make changes in specifications shown herein or make improvements at any time without notice or obligation. It does not relieve the user of responsibility to use sound practices in application, installation, operation, and maintenance of the equipment purchased. Should a conflict arise between the general information contained in this publication, the contents of drawings or supplementary material, or both, the latter shall take precedence.

Siemens is not liable, for whatever legal reason, for damages or personal injury resulting from the use of the application tips.

All rights reserved. Any form of duplication or distribution, including excerpts, is only permitted with express authorization by SIEMENS.

SIMATIC S7-200 Примеры

Группа	Тема
5	**Использование PTO Ramp с S7-200 PLC**

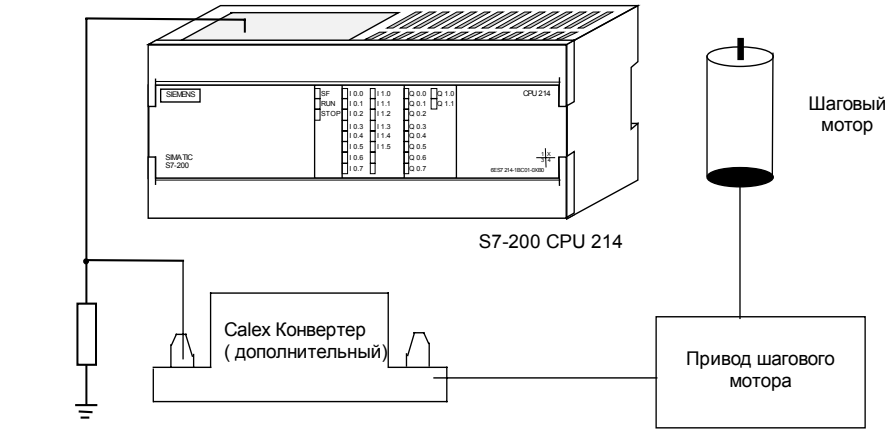
Требуемые для этого совета CPU					
CPU 210 ☐	CPU 212 ☐	CPU 214 ☑	CPU 215 ☑	CPU 216 ☑	ДРУГИЕ ☐

Обзор

Этот пример программирования иллюстрирует использование PTO Ramp с S7-200 (CPU 214, 215 или 216). S7-200 позволяет Q0. 0 и Q0., 1 для генерации быстродействующих выводов передачи импульса (PTO), которые обеспечивают квадрат волнового вывода для определенного числа импульсови циклов.

PTO Ramps используются с шаговыми двигателями, для уменьшения ошибок. Когда дан сигнал для двигателя, произвести внезапное и решительное изменение в быстродействии, двигатель может совершить несколько неповторяемых ошибок. Используя PTO ramp, это изменение быстродействия произойдет более постепенно для того, чтобы избежать или уменьшить ошибки.

Большее количество информации относительно использования выводов передачи импульса обеспечивается в S7-200 Руководстве Системы. Также, см. совет 23 для другого примера программирования, которая использует PTO.



Загрузка резистора, требуется для изменения между размерностями импульса

Рисунок 50.1 Соединение S7-200 с Конвертером Calex , приводом шагового мотора , и шаговым мотором

Аппаратные требования

SIMATIC S7-200 (CPU 214, 215 или 216)
Calex Конвертер напряжения (Модель # 8504)
Привод шаговогового мотора
Шаговый мотор

Описание программы

Хотя и Q0. 0 и Q0. 1 могут использоваться для PTO, эта программа использует только Q0. 0. Для Q0. 0 байт управления - SM67; значение времени цикла загружается в SM68 (значение слова), и значение счетчика импульсов, загружается в SM72 (двойное значение слова).
При использовании функции PTO, число импульсов может быть определено от 1 до 4,294,967,295. Основа времени цикла может быть установлена или в микросекундах или миллисекундах. Диапазон времени цикла в микросекундах - от 250 до 65,535. Диапазон в миллисекундах - от 2 до 65,535. Для этой типовой программы, ядро времени PTO установлено в микросекундах. Для сегментов, требующих более длительного времени, ядро могжет быть установлено в миллисекундах, загружая 16 * 8D вместо 16 * 85 в байт управления в SM67.
Эта программа использует PTO Трубопровод. Функция PTO позволяет двум сегментам вывода импульса быть " в трубе " один после другого. Это обеспечивает непрерывность между последовательными выводами передачи импульса. В этой программе, устанавливается первый сегмент, и выполняется команда PLS. Немедленно после выполнения первой команды PLS, устанавливается следующий сегмент PTO , и выполняется вторая команда PLS. Если делается третья спецификация PTO, перед завершением первого PTO, происходит переполнение трубопровода , который устанавливает бит переполнения PTO трубопровода, SM66. 6.
Программа разработана для " выполнения" следующей RAMP таблицы, которая загружается в DB1:

VW100	Число сегментов в конфигурации
VW102	Число сегментов, оставшихся для выполнения в конфигурации Первоначально устанавливается число сегментов на I0. 0 верхнего предела и уменьшается для каждого выполненного сегмента
VW104	Сегмент 1 время цикла
VW106	Сегмент 1 счетчик импульсов
VW108	Сегмент 2 время цикла
VW110	Сегмент 2 счетчик импульсов
.	
WORD	Сегмент n время цикла
WORD	Сегмент n счетчик импульсов
VW152	Сегмент 13 время цикла
VW154	Сегмент 13 счетчик импульсов

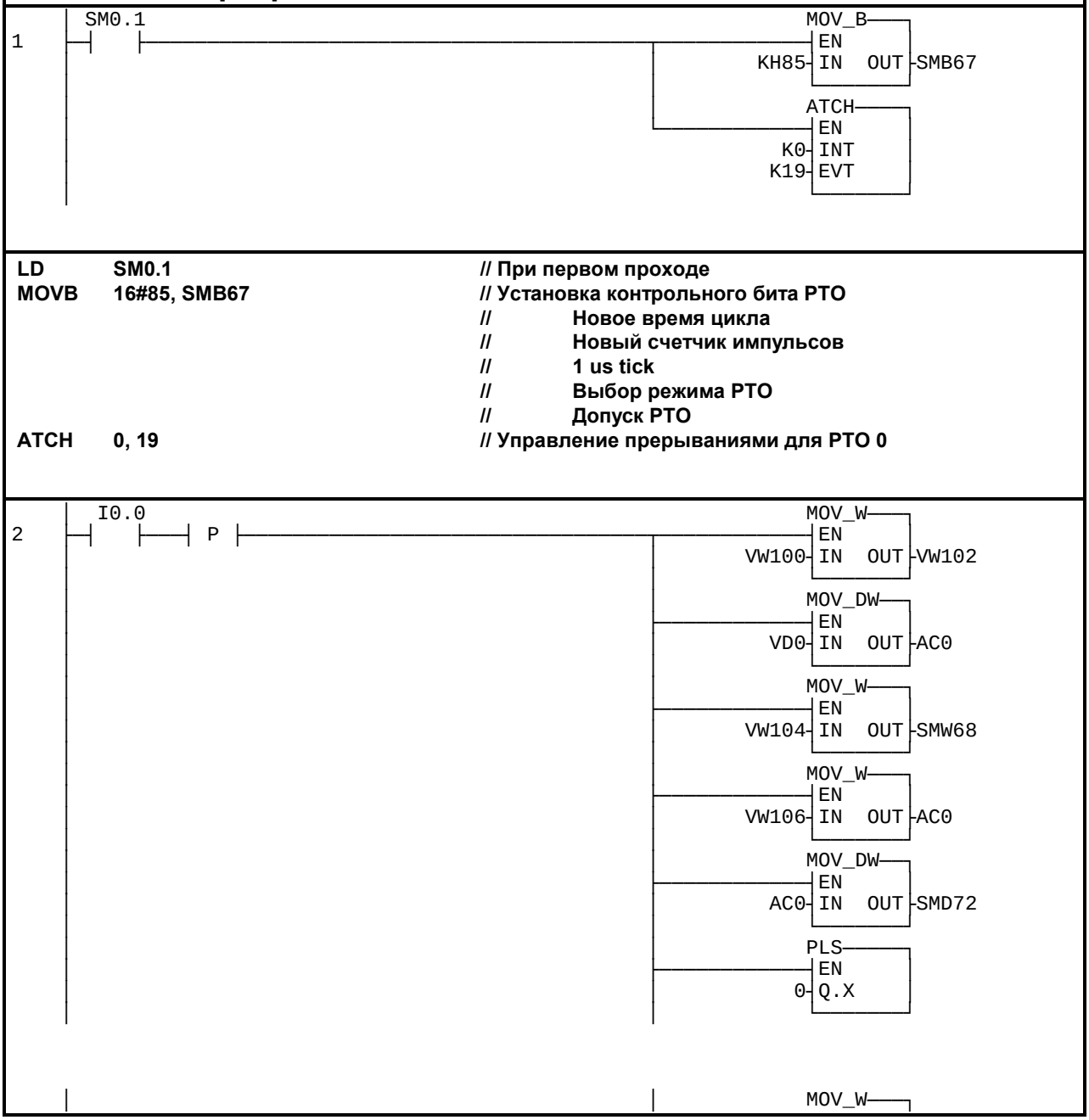
Структура программы

Блок данных 1 Установка Ramp таблицы

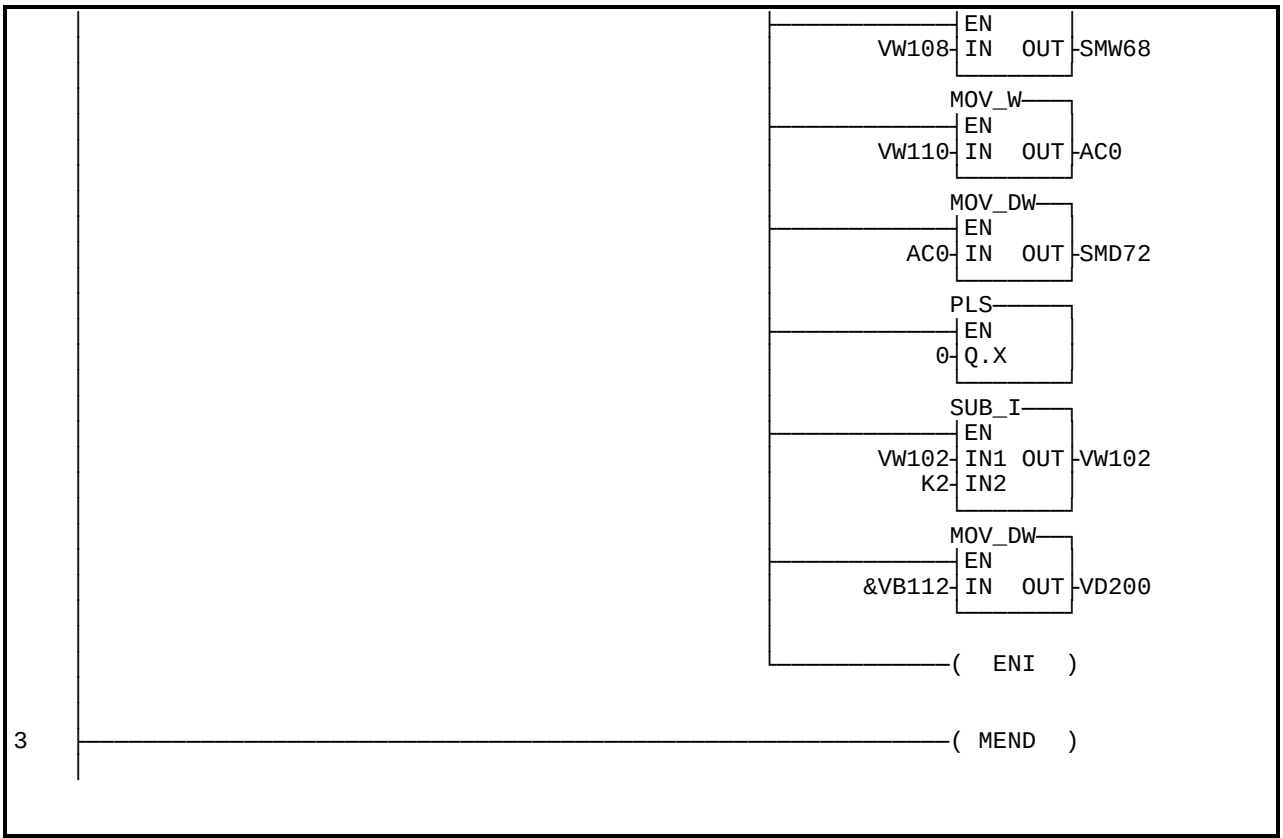
Главная программа Программирование первой передачи импульса и очереди второго

Прерывание 0 Как только завершается одна передача, другая ставится в очередь

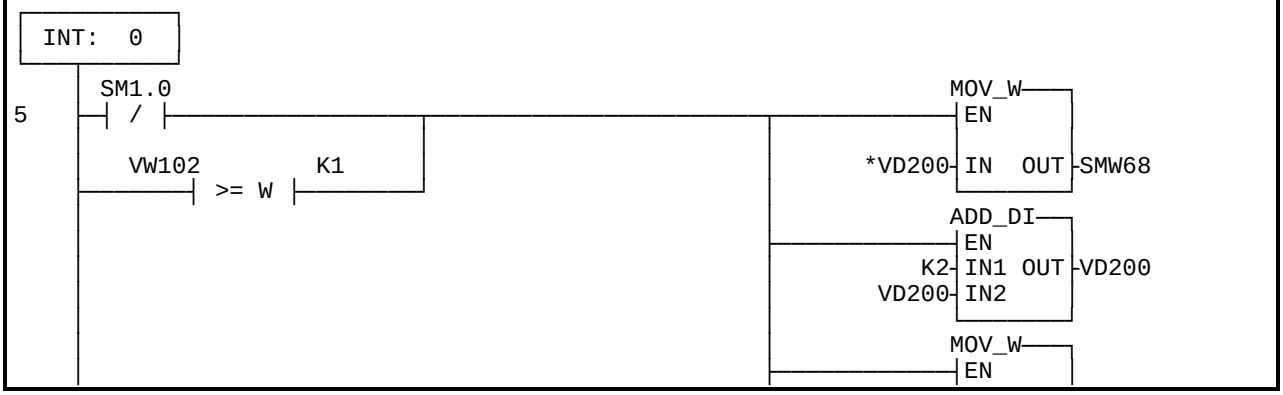
Распечатка программы




```
LD      SM0.1                   // При первом проходе
MOVB    16#85, SMB67            // Установка контрольного бита PTO
                                //      Новое время цикла
                                //      Новый счетчик импульсов
                                //      1 us tick
                                //      Выбор режима PTO
                                //      Допуск PTO
ATCH    0, 19                   // Управление прерываниями для PTO 0
```




```
LD      I0.0                    // Если I0.0
EU                              // И верхний край
MOVW    VW100, VW102            // Инициализация счетчика оставшихся сегментов
MOVD    VD0, AC0                // Очистка AC0
MOVW    VW104, SMW68            // Загрузка 1 сегмента времени цикла
MOVW    VW106, AC0
MOVD    AC0, SMD72              // Загрузка 1 сегмента счетчика импульсов
PLS     0                       // Начало передачи 1 сегмента
MOVW    VW108, SMW68            // Загрузка  2 сегмента времени цикла
MOVW    VW110, AC0
MOVD    AC0, SMD72              // Загрузка 2 сегмента счетчика импульсов
PLS     0                       // Очередь пердачи 2 сегмента
-I      2, VW102                // Уменьшение счетчика оставшихся сегментов
MOVD    &VB112, VD200           // Премещение курсора к следующему сегменту времени
                                //   цикла
ENI                             // Допуск переываний
MEND                            // Конец главной программы
```



```
                                                  *VD200-IN  OUT-AC0
                                                         MOV_DW
                                                        EN
                                                   AC0-IN  OUT-SMD72
                                                         PLS
                                                        EN
                                                     0-Q.X
                                                         ADD_DI
                                                        EN
                                                    K2-IN1 OUT-VD200
                                                 VD200-IN2
                                                         DEC_W
                                                        EN
                                                 VW102-IN  OUT-VW102

6                                                        ( RETI  )
```

```
LDN     SM1.0                    // Если счетчик оставшихся сегментов ! = 0
OW>=    VW102, 1                 // Если счетчик оставшихся сегментов >= 1
MOVW    *VD200, SMW68            // Загрузка семента времени цикла
+D      2, VD200                 // Перемещение курсора к счечику импульсов
MOVW    *VD200, AC0
MOVD    AC0, SMD72               // Загрузка сегмента счетчика импульсов
PLS     0                        // Передача очередного сегмента
+D      2, VD200                 // Премещение курсора к следующему сегменту времени
                                 //  цикла
DECW    VW102                    // Уменьшение счетчика оставшихся сегментов
RETI                             // Конец 0 прерывания
```

Data Block DB1

VW100	13	// Число сегментов в конфигурации
VW102	13	// Число оставшихся сегментов
VW104	2000	// Сегмент 1 время цикла
VW106	500	// # импульсов, Сегмент 1
VW108	1333	// Сегмент 2 время цикла
VW110	500	// # импульсов, Сегмент 2
VW112	1000	// Сегмент 3 время цикла
VW114	500	// # импульсов, Сегмент 3
VW116	800	// Сегмент 4 время цикла
VW118	500	// # импульсов, Сегмент 4
VW120	667	// Сегмент 5 время цикла
VW122	500	// # импульсов, Сегмент 5
VW124	571	// Сегмент 6 время цикла
VW126	500	// # импульсов, Сегмент 6
VW128	500	// Сегмент 7 время цикла
VW130	20000	// # импульсов, Сегмент 7
VW132	571	// Сегмент 8 время цикла
VW134	500	// # импульсов, Сегмент 8
VW136	667	// Сегмент 9 время цикла
VW138	500	// # импульсов, Сегмент 9
VW140	800	// Сегмент 10 время цикла
VW142	500	// # импульсов, Сегмент 10
VW144	1000	// Сегмент 11 время цикла
VW146	500	// # импульсов, Сегмент 11
VW148	1333	// Сегмент 12 время цикла
VW150	500	// # импульсов, Сегмент 12
VW152	2000	// Сегмент 13 время цикла
VW154	500	// # импульсов, Сегмент 13

Приложение: Преобразование Напряжения для Привода шагового мотора

Calex конвертер, используемый в этом примере может производить или 24V понижение или ввод сигнала питания или 5V понижение или сигнал питания . Чтобы использовать имеющиеся в наличии части для создания схемы для подключения PLC к приводу шагового мотора, который требует 5V пониженный сигнал, смотрите рисунок 50.2. (**ПРИМЕЧАНИЕ**: Это - намного больше, чем обычно необходимо.) Чтобы создать схему, которая производит 5V сигнал питания, см. рисунок 50.3.

Внимание:

Время переключения выводов от выключены до включены и от включены до выключены - не то же самое для функций PTO. Это различие во времени переключения проявляется в искажении цикла режима работы. Выводы PTO должны иметь минимальную загрузку по крайней мере 10 % от номинальной загрузки, чтобы обеспечить четкие переходы от выключен до включен и от включен до выключен. Без соответствующей загрузки stepper может пропускать импульс при переключении от одного импульса к другому.

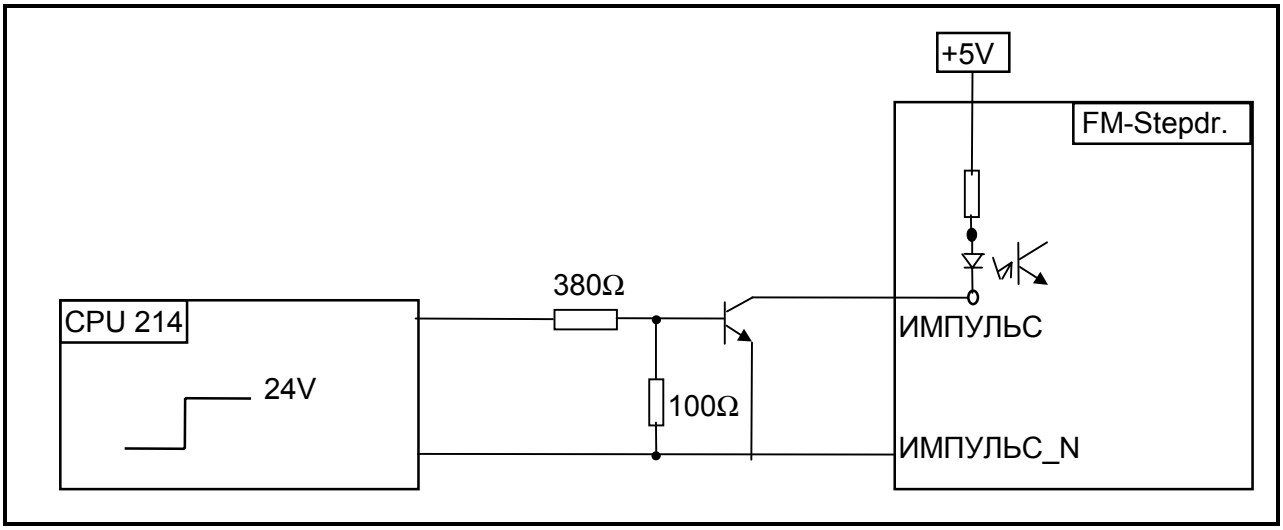


Рисунок 50.2 - Соединение S7-200 PLC и шагового привода требующего 5V пониженного сигнала

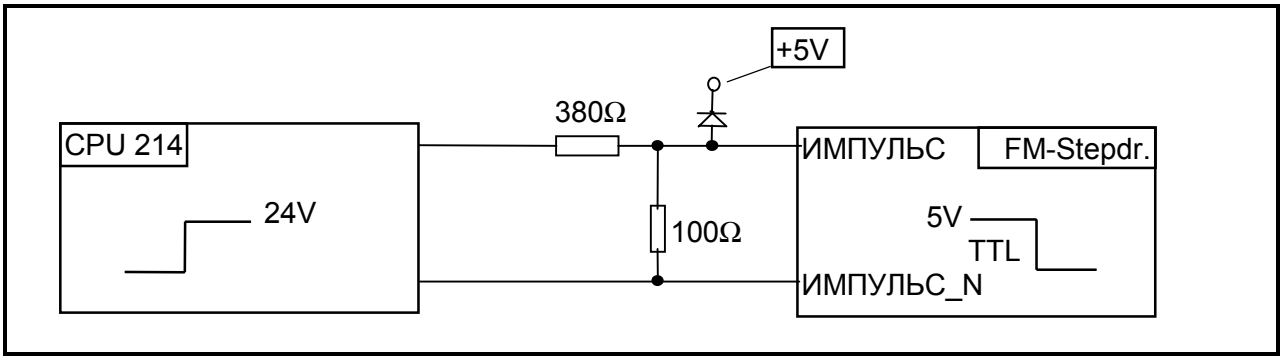


Рисунок 50.3 - Соединение S7-200 PLC и шагового привода требующего 5V сигнала питания

Примечания преобразования

При преобразовании из IEC STL в S7-Micro/DOS STL:

- Добавлять ‘K’ перед всеми нешестнадцатиричными числовыми константами (т.е. 4 ⇒ K4)
- Заменять ‘16#’ на ‘KH’ для всех шестнадцатиричных констант (т.е. 16#FF ⇒ KHFF)
- Запятые обозначают разделение полей. Используйте стрелки или клавишу TAB, чтобы переключаться между полями.
- Чтобы преобразовывать S7-Micro/DOS программу STL в форму LAD, каждая сеть должна начинаться со слова 'СЕТЬ' и номера. Каждая сеть в Application Tip программы обозначена номером на ступенчатой диаграмме. Используйте команду INSNW под меню EDIT, чтобы ввести новую сеть Команды.MEND, RET, RETI, LBL, SBR, и INT каждая получает свою собственную сеть.
- Линия - комментариев, обозначенная '//' невозможна в S7-Micro/DOS, но Сеть - комментариев возможна.

Показанные блоки данных были созданы в Micro / Win; в Miicro / DOS использование редактора V-памяти для ввода блока данных (Обратитесь к руководству пользователя Micro / DOS). Обратите внимание что в Micro / Win:

- 16#.. обозначает шестнадцатиричные значения
- ‘*text*‘ представляет строковые значения

Общие замечания

SIMATIC S7-200 Советы по применению обеспечиваются для того, чтобы дать пользователям S7-200 некоторое представление относительно того, как, некоторые задачи из представления методики программирования, могут быть решены с этим контроллером. Эти инструкции не предполагают отражать все детали или разновидности оборудования, и при этом они не предусматривают любое возможное непредвиденное обстоятельство. Использование S7-200 Советов по применению свободное.
Siemens оставляет право делать изменения в спецификациях, показанных здесь или делать усовершенствования в любое время без примечания или обязательства. Это не освобождает пользователя от ответственности, при использовании звуковых методов в прикладной программе, установке, операции, и сопровождении приобретенного оборудования. Если возникает конфликт между общей информацией, содержащейся в этой публикации, содержанием рисунков и дополнительным материалом, или обоими, последние должны иметь приоритет.
Siemens не несет ответственности, по любой допустимой причине, в повреждениях или персональном ущербе, последующем из использования советов по применению.
Все права защищены. Любая форма дублирования или распространения, включая выдержки, разрешается только с конкретным разрешением SIEMENS.

SIMATIC S7-200 Советы

Группа	Тема
3	**Использование операторного интерфейса TD200 с S7-200 PLC**

Требуемые для совета CPU					
CPU 210 ☐	CPU 212 ☑	CPU 214 ☑	CPU 215 ☑	CPU 216 ☑	ДРУГИЕ ☐

Обзор

Этот пример программы показывает новые свойства TD200 (из версии 1.1) чтобы помочь пользователю ознакомиться с текстовым дисплеем.
Дальнейшая информация относительно текстового дисплея обеспечивается в Руководстве пользователя интерфейса оператора TD200.

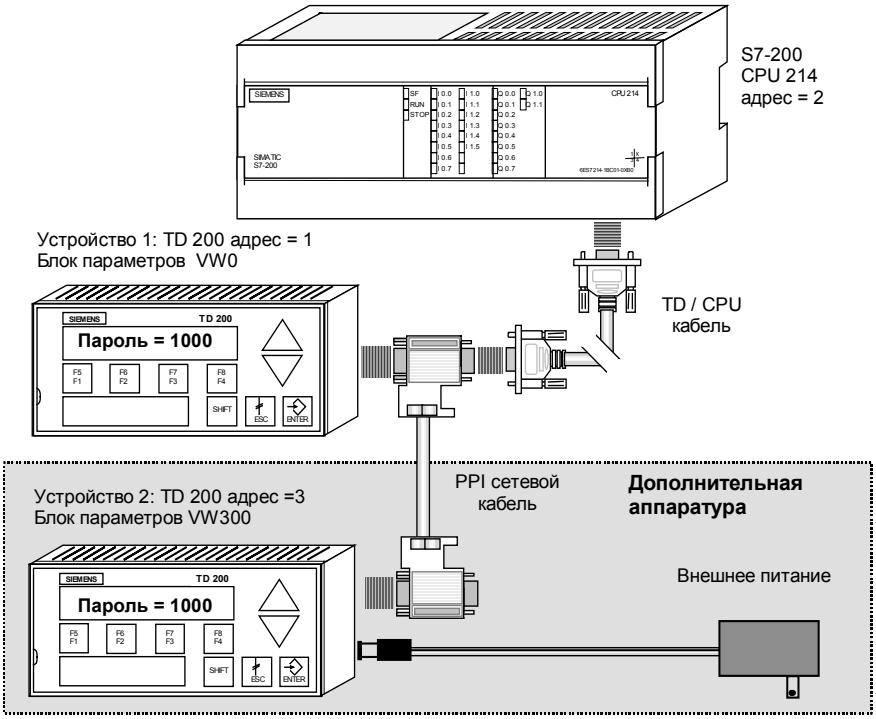


Рисунок 51.1 Подключение нескольких TD200 к S7-200 CPU 214

Аппаратные требования

1 S7-200 PLC
2 TD200 (версия 1.1 или выше)
2 TD / CPU Кабель (included with TD 200)
1 Siemens PC / PPI Кабель
1 PPI Сетевой кабель (дополнительно)
1 Внешнее питание для TD 200 (115V или 220V) (дополнительно)

Общее описание

Обзор
1. Соединение TD с S7-200 CPU
2. Компоненты TD 200
3. Черты TD200 (версия 1.1)
4. Праметры конфигурации TD200
5. Пример программы

1. Соединение TD с S7-200 CPU
Текстовый дисплей 200 (TD 200) - дисплей и интерфейс оператора для семейства программируемых логических контроллеров S7-200.
TD связывается с S7-200 CPU через TD/CPU-кабель. Вы можете подключить один или несколько TD200S к S7-200 CPU, используюя PPI сетевой кабель.
Если Вы подключаете несколько TDS к S7-200 CPU, первый TD может получать мощность от S7-200. Другие TDS должны быть подлючены к внешним источникам питания (AC адаптер).

2. Компоненты TD200
TD200 - маленькое, компактное устройство, которое обеспечивает все необходимые компоненты для связи с помощью интерфейса с вашим S7-200 CPU.
Главные компоненты TD200:

- Текстовая Область Дисплея

Дисплей (две 20-символьных линии) позволяет Вам видеть сообщения, полученные из S7-200 CPU.

- Коммуникационный порт

Порт - с 9 штырьками, D-соединитель, который позволяет Вам подключать TD200 к S7-200 CPU, используюя TD/CPU-кабель.

- Подключение питания

Вы можете подключить внешнее питание к TD200 через доступ подключения питания, размещенный справа от TD200. Это подключение не требуется, когда Вы используете TD/CPU кабель (для одного TD200).

- Клавиатура

TD200 клавиатура имеет девять клавиш. Пять из этих клавиш обеспечивают предопределенные, контекстно - чувствительные функции, и четыре клавиши обеспечивают функции определяемые пользователем. Клавиатура имеет сменную вставку, так что пользователь может получать заказанный шаблон клавиатуры.

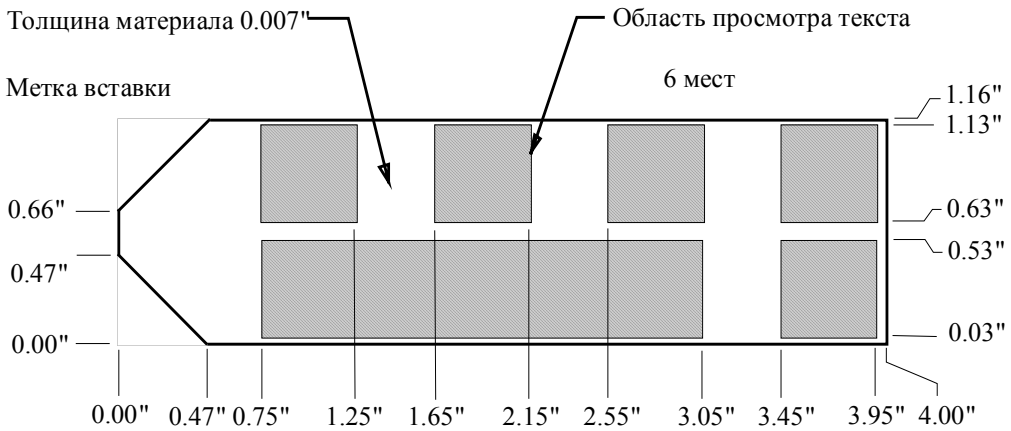


Рисунок 51.2 TD200 Шаблон метки вставки.

3. Черты TD200 (версия 1.1)

Версия 1.1 TD200 включает следующие новые черты:

- Поддержка реальных чисел(с плавающей точкой)
- Защита по паролю
- Использование нескольких TD200 с одной PLC
- Поддержка скорости связи19.2 Кбод
- Область дисплея
- Управление клавишами курсора

4. Параметры конфигурирования TD 200

См. Руководство Интерфейса Оператора TD200 для информации относительно конфигурирования параметров TD200.

5. Пример программы

Этот пример программы покажет пользовтелю несколько возможностей для использованияTD200. Он показывает:

- как использовать клавиши курсора (UP/DOWN), чтобы отображать или выбирать различные сообщения (простая система меню),
- как использовать функциональные клавиши для различных действий,
- как обрабатывать и отображать вложенные данные,
- как обрабатывать гистограммы,
- как листать сообщение в строке дисплея,
- как отображать и редактировать реальное значение (с плавающей запятой),
- how to control text messages with bit values.

Пример программы включает различные маленькие программы S7-200, которые являются вложенными в подпрограммы.
Вставлены следующие программы:

⇨ Отображение сообщение с более чем четырьмя вложенными значениями на одной линии.

⇨ Отображение вложенного значения двойного слова, которое может управляться функциональными клавишами.

⇨ Конфигурация вывода PLC с функциональными клавишами

⇨ Display a bar chart graphic of the analog pot value

⇨ Листание сообщения на первой строке дисплея справа налево

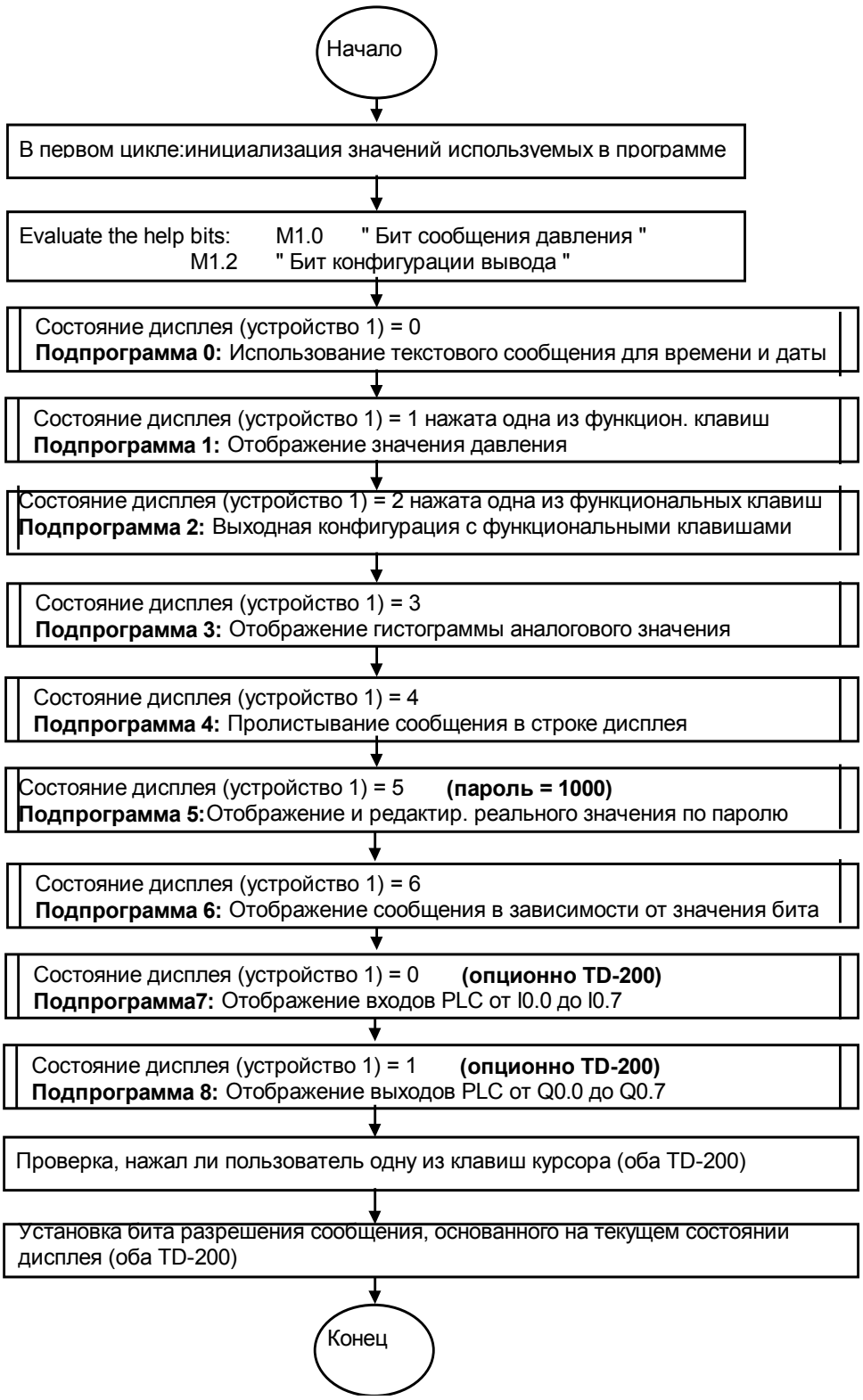
⇨ Отображение и редактирование реального значения с защитой по паролю (пароль = '1000').

⇨ Управление текстовым сообщением, основанным на значении бита (то есть входного бита)

⇨ Отображение PLC-входов от I0. 0 до I0. 7 в двоичной, шестнадцатеричной и десятичной форме.

⇨ Отображение PLC-выходов от Q0. 0 до Q0. 7 в двоичной, шестнадцатеричной и десятичной форме.

Структура программы


Начало
В первом цикле:инициализация значений используемых в программе
Evaluate the help bits: M1.0 " Бит сообщения давления "
M1.2 " Бит конфигурации вывода "
Состояние дисплея (устройство 1) = 0
Подпрограмма 0: Использование текстового сообщения для времени и даты
Состояние дисплея (устройство 1) = 1 нажата одна из функцион. клавиш
Подпрограмма 1: Отображение значения давления
Состояние дисплея (устройство 1) = 2 нажата одна из функциональных клавиш
Подпрограмма 2: Выходная конфигурация с функциональными клавишами
Состояние дисплея (устройство 1) = 3
Подпрограмма 3: Отображение гистограммы аналогового значения
Состояние дисплея (устройство 1) = 4
Подпрограмма 4: Пролистывание сообщения в строке дисплея
Состояние дисплея (устройство 1) = 5 (пароль = 1000)
Подпрограмма 5: Отображение и редактир. реального значения по паролю
Состояние дисплея (устройство 1) = 6
Подпрограмма 6: Отображение сообщения в зависимости от значения бита
Состояние дисплея (устройство 1) = 0 (опционно TD-200)
Подпрограмма7: Отображение входов PLC от I0.0 до I0.7
Состояние дисплея (устройство 1) = 1 (опционно TD-200)
Подпрограмма 8: Отображение выходов PLC от Q0.0 до Q0.7
Проверка, нажал ли пользователь одну из клавиш курсора (оба TD-200)
Установка бита разрешения сообщения, основанного на текущем состоянии дисплея (оба TD-200)
Конец

LAD (S7-MicroDOS)	STL (IEC)

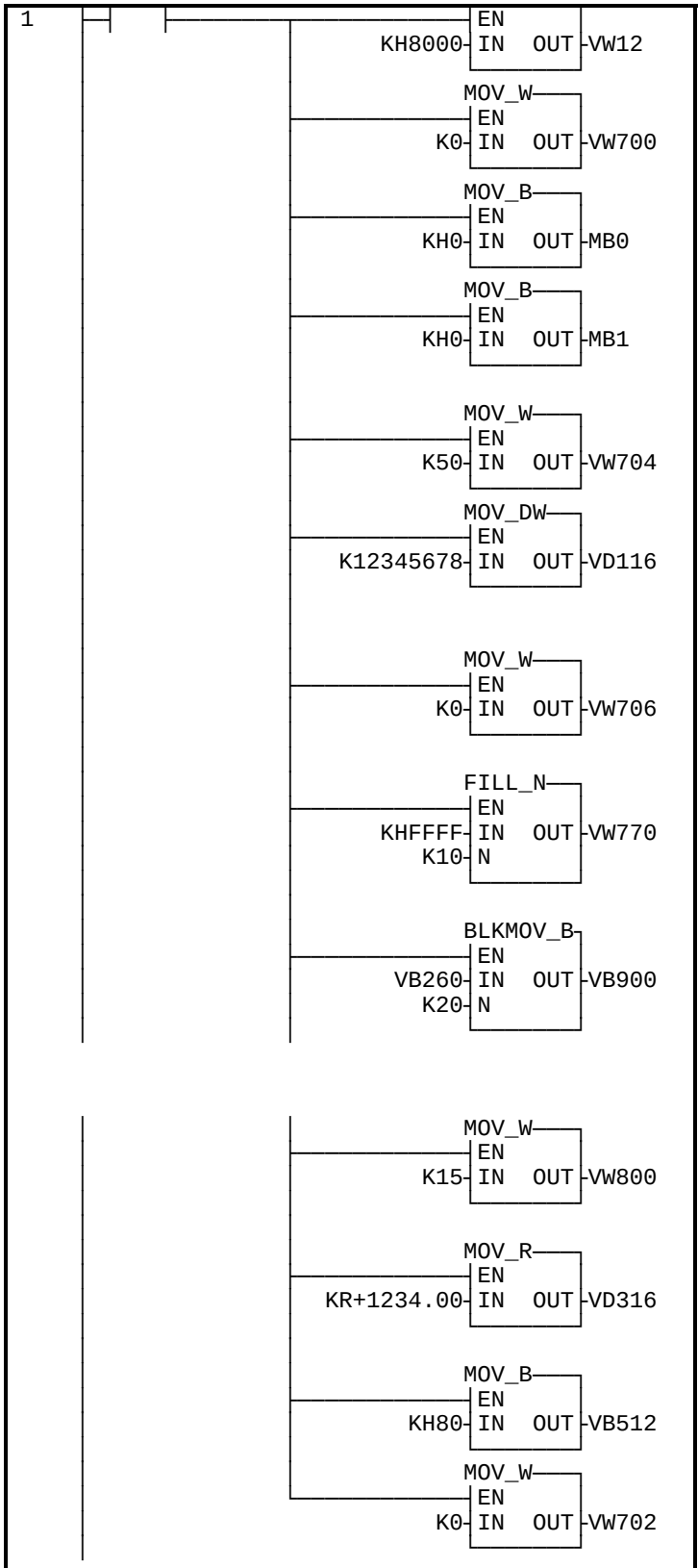
Описание программы и главная программа

// Эта программа показывает новые свойства TD200 (из версии 1.1).
// Возможно подключить несколько TDS к S7-200 CPU - следующая программа обеспечивает два TD200 с различными
// сообщениями.
// Если Вы не имеете два TD200S, чтобы соединиться с S7-200, Вы можете изменять адрес для блока параметра в
// меню установки TD200. (См. Руководство Интерфейса Оператора TD200 для дополнительной информации относительно конфигурирования
// параметров TD200)
// **TD200 устройство 1: Адрес TD200 = 1 Адрес для блока параметров - VW0**
// **(Дополнительный) TD200 устройство 2: Адрес TD200 = 3 Адрес для блока параметров - VW300**
/// **S7-200 PLC:** Адрес PLC = 2
// Устройство 1 предлагает выбор между семью различными меню (сообщениями).
// С дополнительным устройством 2, возможно выбирать между двумя меню.
// Клавиши курсора (UP/DOWN) используются, чтобы отобразить следующее сообщение (меню). Соответствующие биты для клавиш стрелок
// назначаются программой.
// Могут быть выбраны следующие сообщения (меню):
// **1. TD200:**
// меню 1 > Отображение сообщения с более чем четырьмя вложенными
// значениями.
// Подпрограмма 0: Использование текстового сообщения для
// отображения текущих даты и времени
// **(состояние дисплея VW700 = 0)**
// меню 2 > Отображение сообщения со вложенным двойного значением слова,
// которое может управляться функциональными клавишами. Значение
// двойного слова – например значение давления. С функциональными
// клавишами STOP (F1), RUN (F2), SLOW (F3), FAST (F4) - можно изменять
// значение давления.
// Подпрограмма 1: Назначение функциональных клавиш для
// значения давления
// **(состояние дисплея VW700= 1)**
// меню 3 > Конфигурация выхода с функциональными клавишами.
// С функциональными клавишами OFF(F1), ON(F2), CYCL (F3) -возможно // устанавливать, сбрасывать или зациклавать выходы от Q0. 0 до Q0. 7.
// Подпрограмма 2: Назначение функциональных клавиш
// для конфигурирования выхода
// **(состояние дисплея VW700= 2)**
// меню 4 > Display a bar chart graphic - for example the analog pot value.
// The value of the analog pot and the bar chart graphic can be changed by a
// screwdriver.
// **Подпрограмма** 3: Updating the pot 0 value as a bar chart graphic and in
// decimal form. **(состояние дисплея VW700=3)**
// menu 5 > Листание сообщения в строке дисплея справа налево.
// Текст в первой строке дисплея прокручивается справа налево, в
// зависимости от выбранной скорости листания (F1-F4).
// **Подпрограмма 4:** Сдвиг и отображение сообщения
// **(состояние дисплея VW700= 4)**
// menu 6 > Отображение и редактирование реального значения с защитой по
// паролю.Нажимая клавиши ENTER, Вы можете включать режим
// редактирования реальное значение. После того, как правильный пароль
// (1000) печатается - курсор будет помещен внутри реального значения. С
// использованием клавиш курсора текущая цифра может быть изменена.
// Нажмите SHIFT+ клавиши курсора, чтобы изменить позицию курсора
// внутри переменной.

```
//                          Подпрограмма 5:      Очистка битов маркера функциональных клавиш
//                          (подпрограмма системы делает edit-stuff) (состояние дисплея VW700= 5)
//          menu 7 >        Показывает, как управлять сообщением, основанным на значении бита
//                          (т.е. входного бита)
//                          Подпрограмма 6:      Эта подпрограмма пишет на дисплее, в
//                          зависимости от входного бита 10. 0, различный текст.
//                          I0.0 = 1: Отображает на дисплее текст " Двигатель включён  I0.0=1 ".
//                          I0. 0= 0: Отображает на дисплее текст " Двигатель выключен I0.0=0 ".
//                                          (состояние дисплея VW700= 6)
// 2. TD200: (дополнительный)
//          menu 1 >        Отображение PLC-входов от I0. 0 до I0. 7
//                          Входы PLC отображаются в двоичной, шестнадцатеричной и десятичной
//                          форме.
//                          Подпрограмма 6:      Модификация, и отображение PLC-входов
//                                               (состояние дисплея VW702= 0)
//          menu 2 >        Отображение PLC-выходов от Q0. 0 до Q0. 7
//                          Выходы PLC отображаются в двоичной, шестнадцатеричной и
//                          десятичной форме.
//                          Подпрограмма 7:      Модификация, и отображение PLC-выходов
//                                               (состояние дисплея VW702=1)
//
// Дальнейшая информация относительно TD200 обеспечивается в Руководстве Интерфейса Оператора
TD200.
//
```

// Инициализация используемых значений

| SM0.1 MOV_W──┐	**LD SM0.1 // В первом цикле**



// инициализация TD200 устройство 1
MOVW 16#8000, VW12
// разрешено первое ссобщение (устройство 1)

MOVW 0, VW700 // состояние дисплея
// устройство 1 устанавливается в 0

MOVB 16#0, MB0
// очистка области функциональных клавиш
MOVB 16#0, MB1 // очистка области бито справки

// Инициализация вложенных значений для давления
// и скорости модификации

MOVW 50, скорость модификации (давление)
VW704 // Инициализация
// скорости модификации (давление)
MOVD 12345678, VD116 // Установка значения давления

// Инициализация конфигурации выхода с
// Функциональными клавишами
MOVW 0, VW706 // Инициализация циклов I/O

// Инициализация заполнения гистограммы
FILL 16#FFFF, VW770, 10
// заполнение сообщения с полными блоками

// Инициализация прокрутки сообщения

BMB VB260, VB900, 20
// Копирование сообщений для прокрутки

MOVW 15, VW800 // Инициализация скорости модификации
// для прокрутки (медленная)
// Инициализация отображения и редактирования реального значения

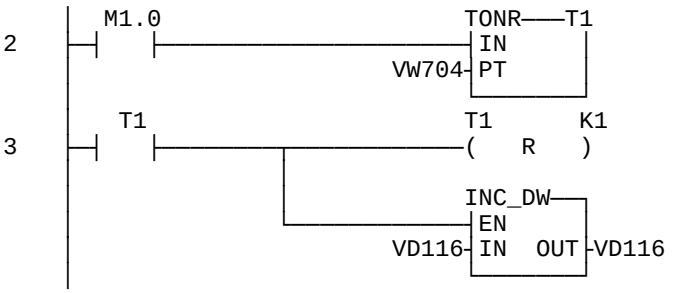
MOVR 1234.0, V316
// Инициализация реального значения

// Инициализация TD200 устройство 2 (дополнительное)

MOVB 16#80, VB512 Допуск первого
// сообщения Устройства 2

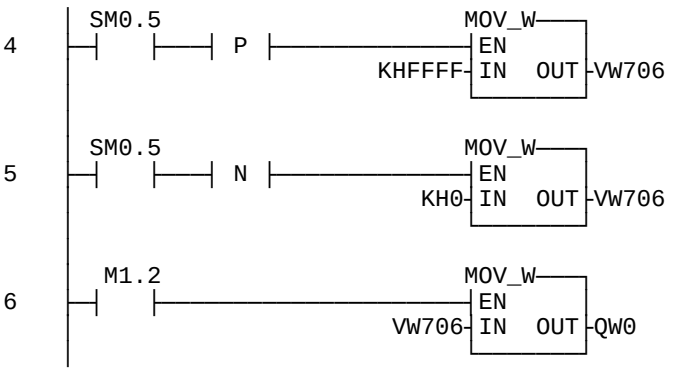
```
MOVW  0, VW702
      // Установка состояния дисплея на 0 (устройство 2)
```

// Запуск таймера для увеличить значений данных (устройство 1)



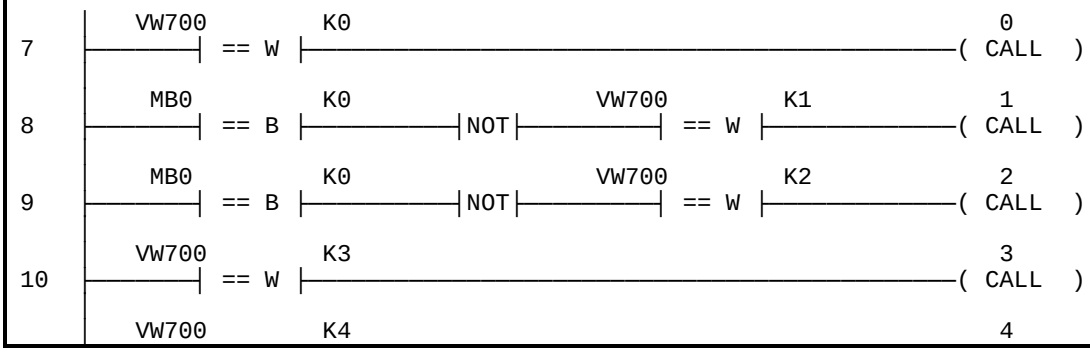

```
LD    M1.0     // run timer only when
               // pressure running
TONR  T1, VW704        // ... 10ms таймер

LD    T1       // Если таймер истек,
R     T1, 1    // ... сброс таймера

INCD  VD116    // увеличение давления
```

```
//
// Manage the cycling of the outputs
//
```




```
LD    SM0.5    // if half second tick изменился
EU             // ...от 0 до 1
MOVW  16#FFFF, VW706
               // ... включается цикл выходов

LD    SM0.5    // if half second tick изменился
ED             // ... от 1 до 0
MOVW  16#0, VW706
               // ... выключается цикл выходов

LD    M1.2     // если допущен цикл I/O ,

MOVW  VW706, QW0     // пишутся выходы
```

```
//
// Переход в соответствующую подпрограмму:
// Проверка, имелась ли нажатая функциональная клавиша и вызов управления, основанного на состоянии дисплея ( Сообщение которого отображается).
//
// ******* Подпрограмма для первого TD 200 (устройство 1)
```



```
11  ├────────┤ == W ├──────────────────────────────────────────────( CALL  )
    │
    │   VW700        K5                                                 5
12  ├────────┤ == W ├──────────────────────────────────────────────( CALL  )
    │
    │   VW700        K6                                                 6
13  ├────────┤ == W ├──────────────────────────────────────────────( CALL  )
    │
```

LDW= VW700, 0 // Состояние дисплея = 0?
CALL 0 // Установка состояния в 0 (отображение даты и времени)

LDB= MB0, 0 // Нажаты ли функциональные клавиши...
NOT
AW= VW700, 1 // ... и отображение состояния = 1?
CALL 1 // Установка состояние дисплея один (назначение функциональным клавишам сообщения давления)

LDB= MB0, 0 // Если нажата функциональная клавиша...
NOT
AW= VW700, 2 // ... и состояние дисплея = 2?
CALL 2 // Установка состояния дисплея два (назначение функциональных клавиш конфигурированию выходов)

LDW= VW700, 3 // если состояние дисплея = 3?
CALL 3 // Установка состояния дисплея три (отображение гистограммы of the)

LDW= VW700, 4 // Если состояние дисплея = 4?
CALL 4 // Установка состояния дисплея четыре (прокрутка сообщения на дисплее)

LDW= VW700, 5 // Если состояние дисплея = 5?
CALL 5 // Установка состояния дисплея пять (отображение и редактирование реального значения с защитой по паролю)
LDW= VW700, 6 // Если состояние дисплея = 6?
CALL 6 // Установка состояния дисплея шесть (отображение сообщения в зависимости от значения бита)

```
//
// ******* Подпрограмма для второго TD200 (устройство 2, дополнительное))
//
```

```
14  VW702        K0                                    7
    ──────┤ == W ├──────────────────────────────────( CALL   )

15  VW702        K1                                    8
    ──────┤ == W ├──────────────────────────────────( CALL   )
```

```
LDW=   VW702, 0          // Если состояние дисплея = 0?
CALL   7                 // Установка состояния дисплея ноль (отображение PLC-входов от
I0.0 до I0.7)

LDW=   VW702, 1          // Если состояние дисплея = 1?
CALL   8                 // Установка состояния дисплея один (отображение PLC-выходов
отQ0.0 до Q0.7)
```

```
//
// Назначение клавиш курсора усстройства  1:
// Первый блок параметров начинается с VW0. Клавиши курсора возвращены в четвертый байт
// Блока параметров.                        "UP"-стрелка :  V3.2
//                                          "DOWN"-стрелка: V3.3
// Эти биты, используются, чтобы прокручивать сообщения (системное меню).
//
```

```
16  V3.3                    V3.3     K1
    ─┤   ├──────────────────(   R   )
                          INC_W
                        ┌──────┐
                        │EN    │
                VW700───│IN  OUT│──VW700
                        └──────┘

17  VW700        K7         MOV_W
    ──────┤ == W ├──────────┌──────┐
                          │EN    │
                    K0────│IN  OUT│──VW700
                          └──────┘

18  V3.2                    V3.2     K1
    ─┤   ├──────────────────(   R   )
                          DEC_W
                        ┌──────┐
                        │EN    │
                VW700───│IN  OUT│──VW700
                        └──────┘

19  VW700        K-1        MOV_W
    ──────┤ <= W ├──────────┌──────┐
                          │EN    │
                    K6────│IN  OUT│──VW700
                          └──────┘
```

```
LD      V3.3     // Нажата стрелка-"UP"?
R       V3.3, 1  // ...сброс флага

INCW    VW700    // ..следующее состояние
дисплея

LDW=    VW700, 7         // состояние
дисплея=7?
MOVW    0, VW700
        // ... установка состояния дисплея= 0


LD      V3.2
        // Нажата стрелка "DOWN"?
R       V3.2, 1  // ... сброс флага

DECW    VW700    // ...предыдущее состояние


LDW<=   VW700, -1 // состояние дисплея= -1?
MOVW    4, VW700
        // ... установка состояния дисплея= 6
```

```
//
// Назначение клавиш курсора устройства 2 (дополнительного):
// Второй блок параметров начинается с VW300. Клавиши курсора возвращают в четвертый байт
блока параметров.                        "UP"-стрелка :  V353.2
//                                       "DOWN"-стрелка: V353.3
// Эти биты используют для прокрутки сообщений (системное меню).
//
```

```
20  V503.3                    V503.3    K1
 ─┤   ├──────────────────────(   R   )
                              INC_W
                             ┌EN
                      VW702─┤IN   OUT├─VW702

21  VW702        K2          MOV_W
 ──────┤ == W ├──────────────┌EN
                          K0─┤IN   OUT├─VW702

22  V503.2                    V503.2    K1
 ─┤   ├──────────────────────(   R   )
                              DEC_W
                             ┌EN
                      VW702─┤IN   OUT├─VW702

23  VW702        K-1         MOV_W
 ──────┤ <= W ├──────────────┌EN
                          K1─┤IN   OUT├─VW702
```

LD V503.3 //Нажата стрелка "UP"?
R V503.3, 1 // ... сброс флага
INCW VW702
// ... следующее состояние дисплея (устройство 2)

LDW= VW702, 2 // состояние дисплея= 2?
MOVW 0, VW702
// ... установка состояния дисплея= 0

LD V503.2 // Нажата стрелка "UP"?
R V503.2, 1 // ... сброс флага

DECW VW702 // предыдущее состояние дисплея

LDW<= VW702, -1 // состояние дисплея= -1?
MOVW 1, VW702
// ... установка состояния дисплея= 1

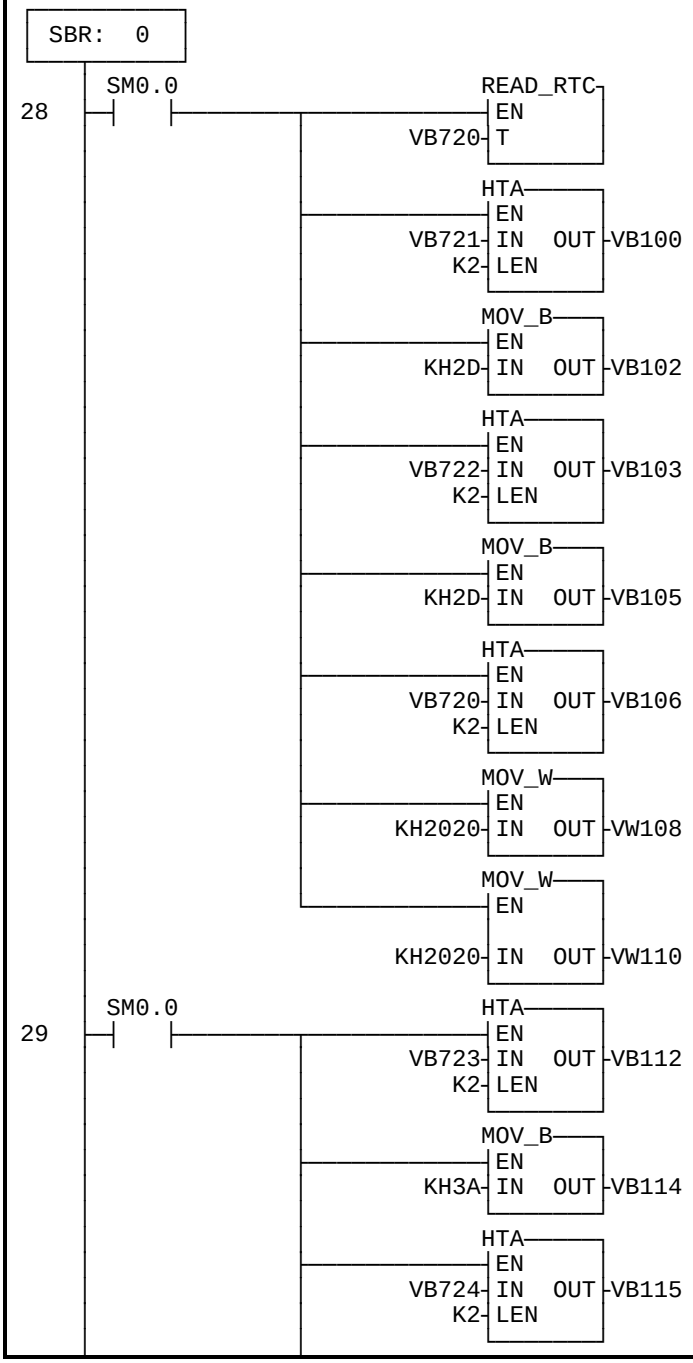
//
// Установка бита разрешения сообщения основанного на текущем состоянии дисплея.
//

```
24  SM0.0                     SHR_W
 ─┤   ├──────────────────────┌EN
                       KH8000─┤IN   OUT├─VW12
                         VB701─┤N

25  SM0.0                     SHR_W
 ─┤   ├──────────────────────┌EN
                       KH8000─┤IN   OUT├─VW512
                         VB703─┤N

26 ──────────────────────────( MEND  )
```

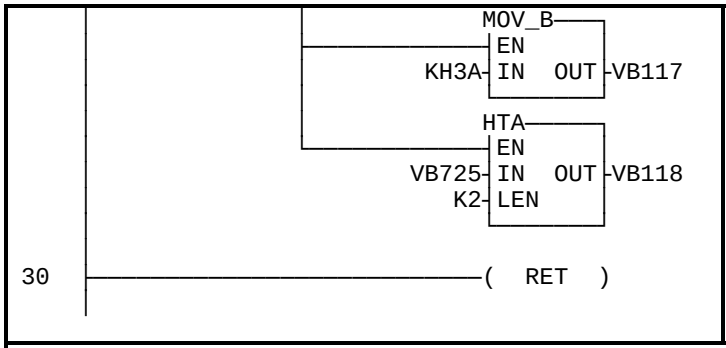
LD SM0.0 // На каждом проходе:
MOVW 16#8000, VW12 // Установка //состояния дисплея для первого //сообщения (MSB) устройства 1
SRW VW12, VB701 // Сдвиг // основанный на состоянии дисплея LS // байт
LD SM0.0 // При каждом // проходе:
MOVW 16#8000, VW512 // Установка // состояния дисплея для первого // сообщения (MSB) устройства 2
SRW VW512, VB703// Сдвиг, основанный // на состоянии дисплея LS байт
MEND

Подпрограммы

```
//
// Подпрограмма 0: Использование сообщений для отображения времени и даты.
// состояние дисплея (устройство 1) = 0
// Подпрограмма выбирает дату и время часов реального времени. После преобразования,
//  использующего HTA  (Шестнадцатеричный к ASCII) текстовое сообщение показывается на
//дисплее.
//
```

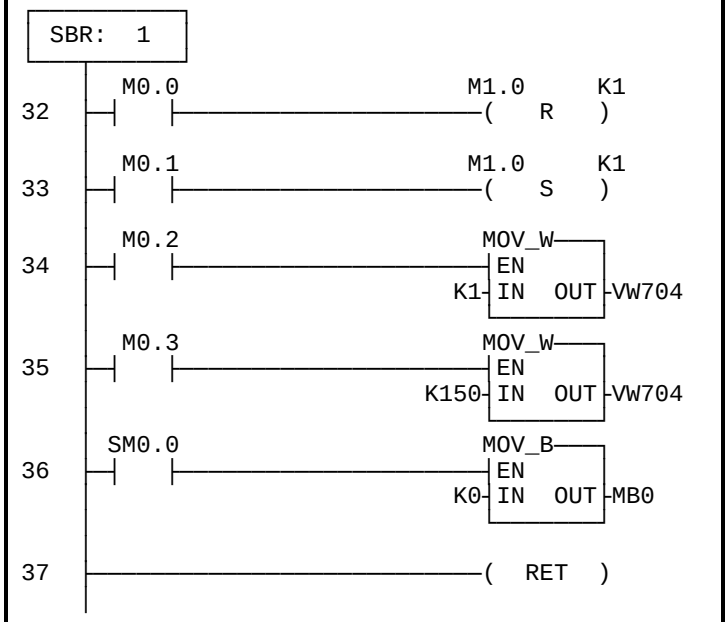



```
LD      SM0.0          // прои каждом
                       // проходе:
TODR    VB720          // ... Получение
                       // данных часов

HTA     VB721, VB100, 2
        // преобразование месяца

MOVB    16#2D, VB102   // загрузка "-"

HTA     VB722, VB103, 2
        // преобразование дня месяца

MOVB    16#2D, VB105   // загрузка "-"

HTA     VB720, VB106, 2
        // преобразование года

MOVW    16#2020, VW108
        // Загрузка двух пробелов

MOVW    16#2020, VW110
        // Загрузка двух пробелов

LD      SM0.0          // при каждом
                       //проходе:
HTA     VB723, VB112, 2
        // преобразование часа

MOVB    16#3A, VB114   // загрузка ":"

HTA     VB724, VB115, 2
        // преобразование минут
```



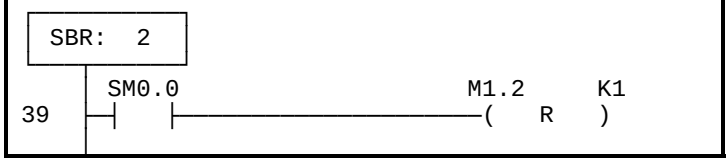

```
MOVB   16#3A, VB117    // загрузка":"

HTA    VB725, VB118, 2
       // преобразование секунд

RET
```

```
//
// Подпрограмма 1: Отображение вложенного значения (формат: двойное слово), которое может
// быть изменено функциональными клавишами
// состояние дисплея (устройство 1) = 1
// Эта подпрограмма обрабатывает функциональные клавиши для дисплея, устанавливают 1.
//
```

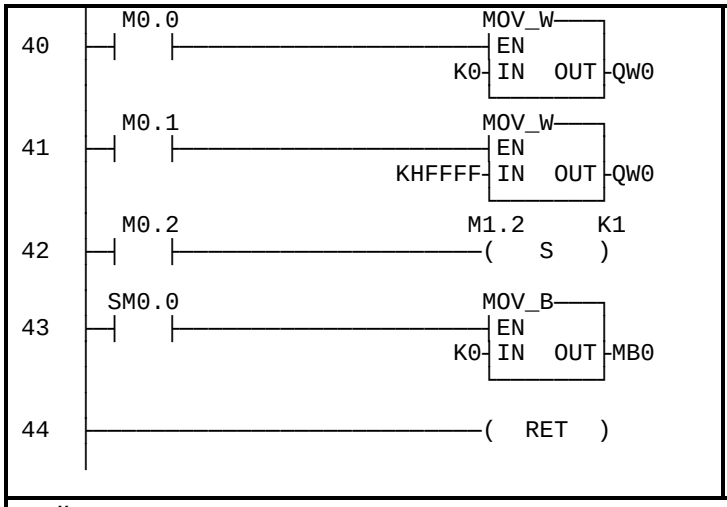



```
SBR    1

LD     M0.0     // если нажата клавиша F1,
R      M1.0, 1  // ... завершение изменения
                //давления

LD     M0.1     // если нажата клавиша F2,
S      M1.0, 1  // ... начало изменения
                //давления

LD     M0.2     // если нажата клавиша F3,
MOVW   1, VW704
       // Установка давления для быстрой
       //модификации

LD     M0.3     // если нажата клавиша F4,
MOVW   150, VW704
       // установка давления для
       // медленной модификации

LD     SM0.0    // при каждом проходе...
MOVB   0, MB0
       // ... очистка области
       // функциональных клавиш

RET
```

```
//
// Подпрограмма 2: Конфигурирование выходов с функциональными клавишами
// состояние дисплея (устройство 1) = 2
// Эта подпрограмма обрабатывает функциональные клавиши для конфигурирования вывода
//(состояние дисплея 2).
//
```

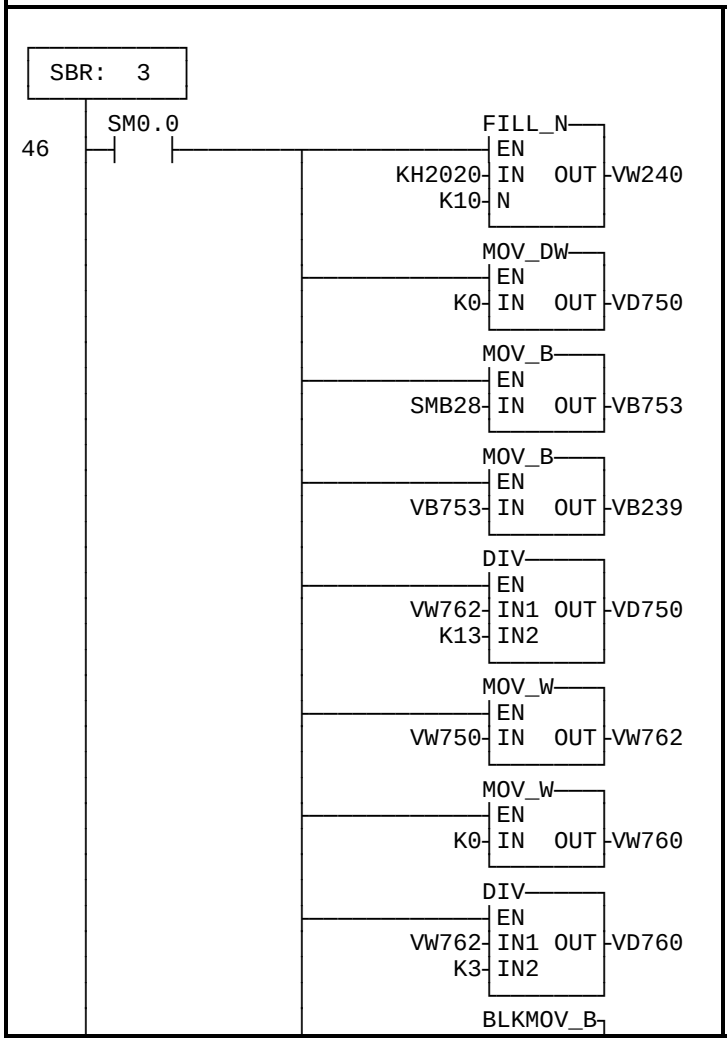



```
SBR    2

LD     SM0.0    // при каждом проходе:
R      M1.2, 1  // ... остановка цикла I/0
```

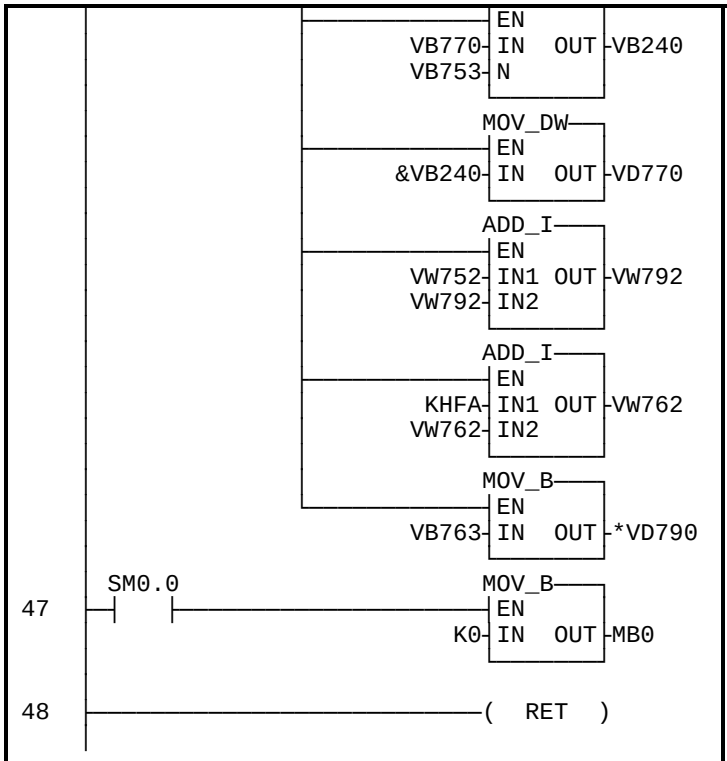



```
LD      M0.0     // если нажата клавиша F1,
MOVW    0, QW0   // ... Отключение всех
                 // выходов

LD      M0.1     // если нажата клавиша F2,
MOVW    16#FFFF, QW0
                 // ... установка всех выходов

LD      M0.2     // если нажата клавиша F3,
S       M1.2, 1  // ... разрешение циклов I/O

LD      SM0.0    // при каждом проходе...
MOVB    0, MB0   // ...очистка области
                 // функциональных клавиш

RET
```

```
//
// Подпрограмма 3: Отображение 0 значения горшка в форме гистограммы и десятичного значения
// состояние дисплея(устройство1) = 3
// Эта подпрограмма отображает текущее значение аналогового горшка 0 в форме гистограммы.
// Значение горшка может быть изменено отверткой.
//
```




```
SBR     3

LD      SM0.0            // при каждом
проходе:
FILL    16#2020, VW240, 10
        // Заполните сообщение пробелами

MOVD    0, VD750
        // Очистка пространства для работы

MOVB    SMB28, VB753     // получение 0
значения горшка

MOVB    VB753, VB239     // отображение
// значения горшка в первой строке дисплея

DIV     13, VD750        // деление значения
горшка на 13 ( 20 символов * 13 > 256)

MOVW    VW750, VW762     // Получение остатка

MOVW    0, VW760         // очистка MSB

DIV     3, VD760         // деление остатка на
3 для получения  fifths
```



BMB VB770, VB240, VB753
// Перемещение полных областей
// для отображения буфера

MOVD &VB240, VD770
// Указание на начало буфера

+I VW752, VW792
// Смещение к первому пробелу

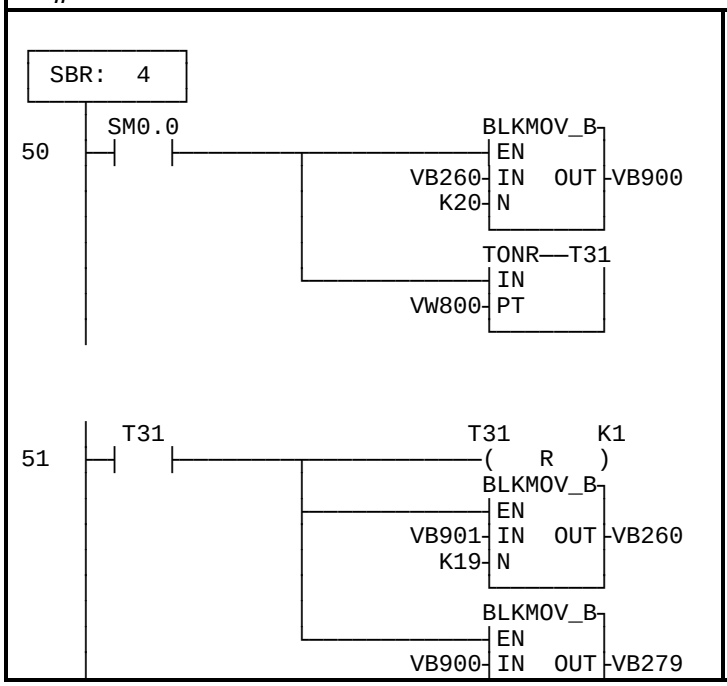
+I 16#FA, VW762
// Создание символа

MOVB VB763, *VD790
// Сохранение частичного блока в
// буфере

LD SM0.0 // при каждом
// проходе:
MOVB 0, MB0 // ...очистка области
// функциональных клавиш

RET

//
// Подпрограмма 4: Прокрутка текстового сообщения на строке дисплея справа налево
// состояние дисплея (устройство1) = 4
// Эта подпрограмма пишет текстовое сообщение на первой строке дисплея и листает его налево, в
// зависимости от выбранной скорости.
//



SBR 4

LD SM0.0 // при каждом
// проходе:
BMB VB260, VB900, 20
// Копирование текста сообщения в
// рабочую область

TONR T31, VW800 // Пуск таймера T31

LD T31 // Если таймер истек,
R T31, 1 // ... сброс таймера

BMB VB901, VB260, 19 // ...Отображение
// текста сообщения (от 2 до 20 символов)

BMB VB900, VB279, 1// ... Отображение
// первого символа в конце сообщения

```
                                K1-N
     M0.0                       MOV_W
52 ─┤  ├──────────────────────EN
                              K1-IN  OUT-VW800
     M0.1                       MOV_W
53 ─┤  ├──────────────────────EN
                              K5-IN  OUT-VW800
     M0.2                       MOV_W
54 ─┤  ├──────────────────────EN
                             K10-IN  OUT-VW800
     M0.3                       MOV_W
55 ─┤  ├──────────────────────EN
                             K15-IN  OUT-VW800
    SM0.0                       MOV_B
56 ─┤  ├──────────────────────EN
                              K0-IN  OUT-MB0

57 ──────────────────────────(  RET  )
```

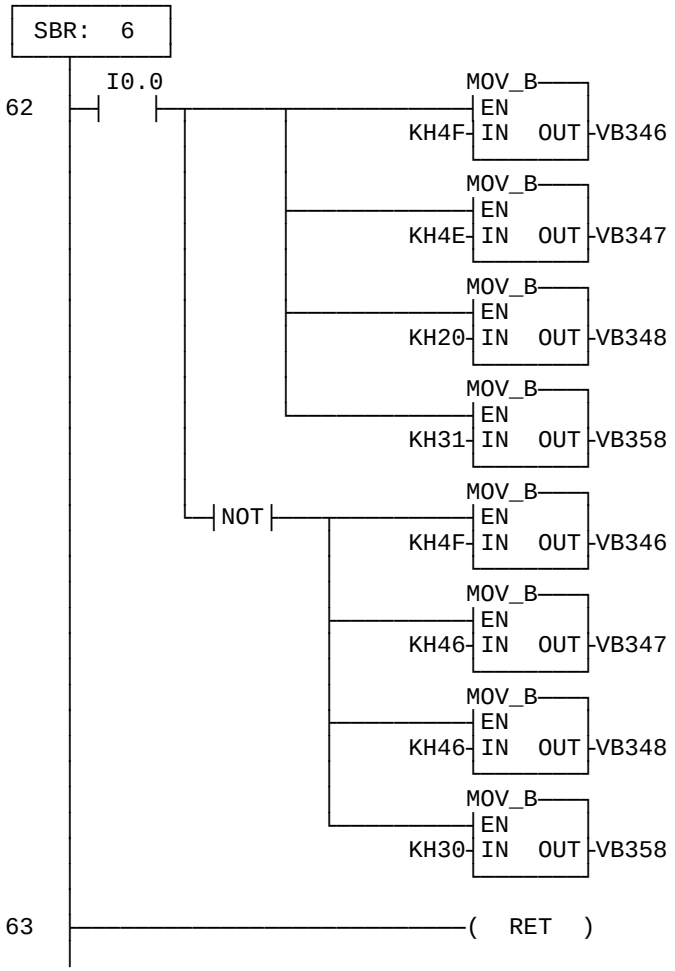
```
LD      M0.0    // Если нажата клавиша F1 ,
MOVW    1, VW800
        // установка прокрутки " FAST "

LD      M0.1    // если нажата клавиша F2,
MOVW    5, VW800
        // установка прокрутки “FAST-”

LD      M0.2    // если нажата клавиша  F3,
MOVW    10, VW800
        // установка прокрутки “SLOW+”
LD      M0.3    // если нажата клавиша  F4,
MOVW    15, VW800
        // установка прокрутки “SLOW”

LD      SM0.0   // при каждом проходе:
MOVB    0, MB0
        // ... очистка области
// функциональных клавиш

RET
```

```
//
// Подпрограмма 5: Отображение и редактирование реального значения с защитой по паролю
// состояние дисплея (устройство 1) = 5
// Эта подпрограмма очищает биты маркера функциональных клавиш.
// Редактирование реального значения после нажатия клавиши ENTER , будет выполнено
// подпрограммой системы TD200.
// Нажимая клавишу ENTER, Вы можете редактировать реальное значение. После того, как
// печатается правильный пароль (1000) , курсор будет помещен в реальное значение. С
// использованием стрелок ВВЕРХ и ВНИЗ символ у курсора может быть изменен. Чтобы изменять
// десятичную позицию  курсора реального значения необходимо нажать SHIFT +стрелки.
//
```

```
SBR:  5

    SM0.0                       MOV_B
59 ─┤  ├──────────────────────EN
                              K0-IN  OUT-MB0

60 ──────────────────────────(  RET  )
```

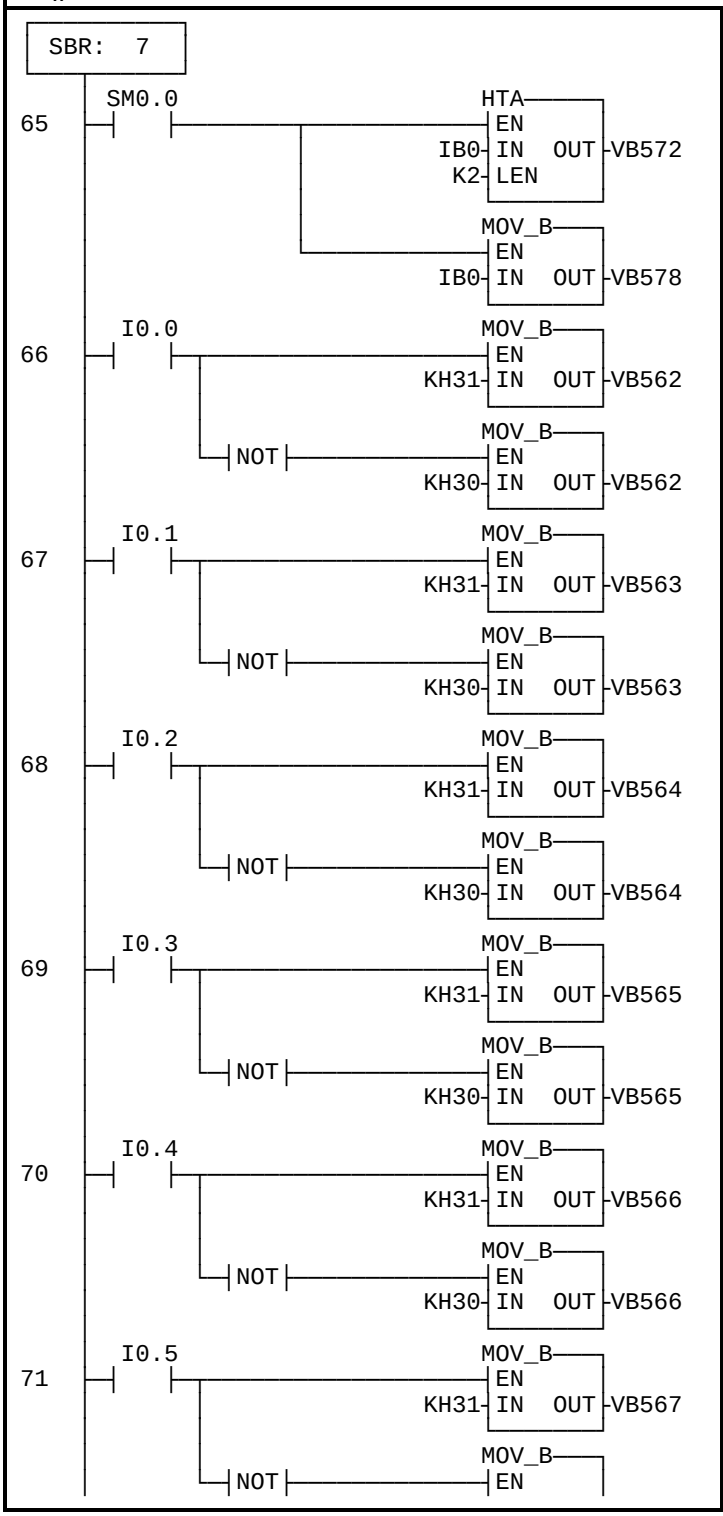
```
SBR     5

LD      SM0.0   // при каждом проходе:
MOVB    0, MB0
        // ... очистка области
// функциональных клавиш

RET
```

```
//
// Подпрограмма 6: Показывает, как управлять текстовым сообщением, основанным на значении
// разряда (например а бита ввода)
// состояние дисплея 6 (устройство 1) = 6
// Эта подпрограмма пишет, в зависимости от Входного бита 10. 0, различный текст на дисплей.
// I0.0 = 1: Отображает на дисплее текст “Мотор включен  I0.0 = 1” .
// I0.0 = 0: Отображает на дисплее текст “Мотор выключен I0.0 = 0”.
//
```

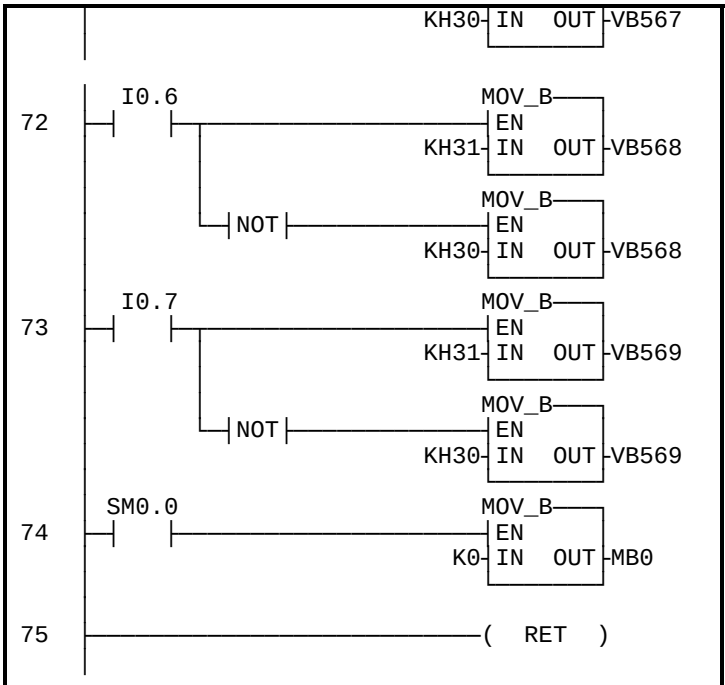



```
SBR    6

LD     I0.0      // если включен ввод,
MOVB   16#4F, VB346
       // пишет на дисплее ‘O’

MOVB   16#4E, VB347
       // пишетна дисплее ‘N’

MOVB   16#20, VB348
       // пишет на дисплее ‘ ‘ (пробел)

MOVB   16#31, VB358
       // пишет на дисплее ‘1’

NOT    // если ввод выключен,

MOVB   16#4F, VB346
       // пишет на дисплее ‘O’

MOVB   16#46, VB347
       // пишет на дисплее ‘F’

MOVB   16#46, VB348
       // пишет на дисплее ‘F’
MOVB   16#30, VB358
       // пишет на дисплее ‘0’

RET
```

```
//
// Подпрограмма 7: Отображает входы PLC от I0.0 до I0.7
// состояние дисплея (дополнительное устройство 2) = 0
// Эта подпрограмма читает PLC Входы от I0. 0 до I0. 7 и отображает их в двоичном,
// шестнадцатеричном и десятичном виде.
//
```

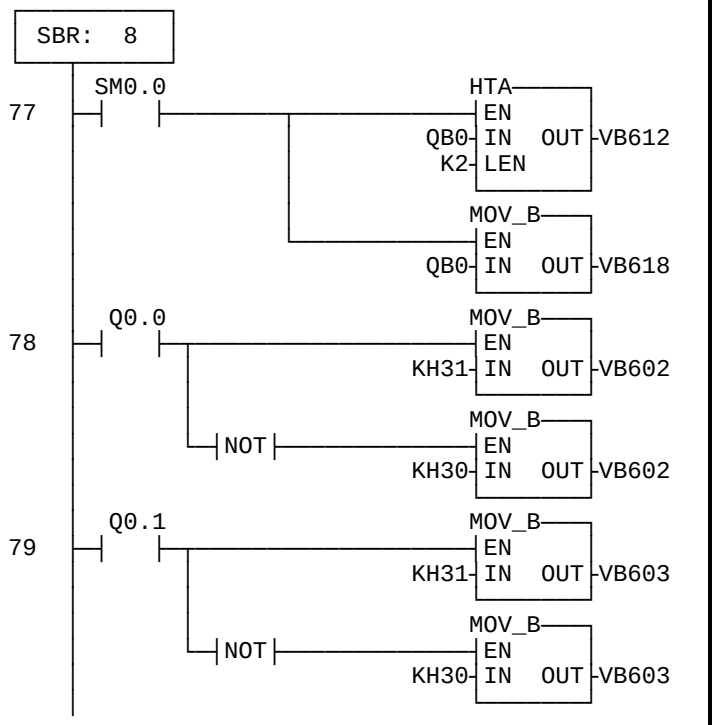



```
SBR     7

LD      SM0.0   // при каждом проходе:
HTA     IB0, VB572, 2     // Преобразование
// входного байта из шестнадцатиричного в
// ASCII

MOVB    IB0, VB578
        // Входной байт дисплея в
// десятичном виде

LD      I0.0                // если ввод
// включен,
MOVB    16#31, VB562      // ... пишет “1” в
// соответствующее место дисплея

NOT             // если ввод выключен,
MOVB    16#30, VB562      // ... пишет “0” в
// соответствующем месте дисплея

LD      I0.1
MOVB    16#31, VB563

NOT
MOVB    16#30, VB563

LD      I0.2
MOVB    16#31, VB564

NOT
MOVB    16#30, VB564

LD      I0.3
MOVB    16#31, VB565

NOT
MOVB    16#30, VB565

LD      I0.4
MOVB    16#31, VB566

NOT
MOVB    16#30, VB566

LD      I0.5
```

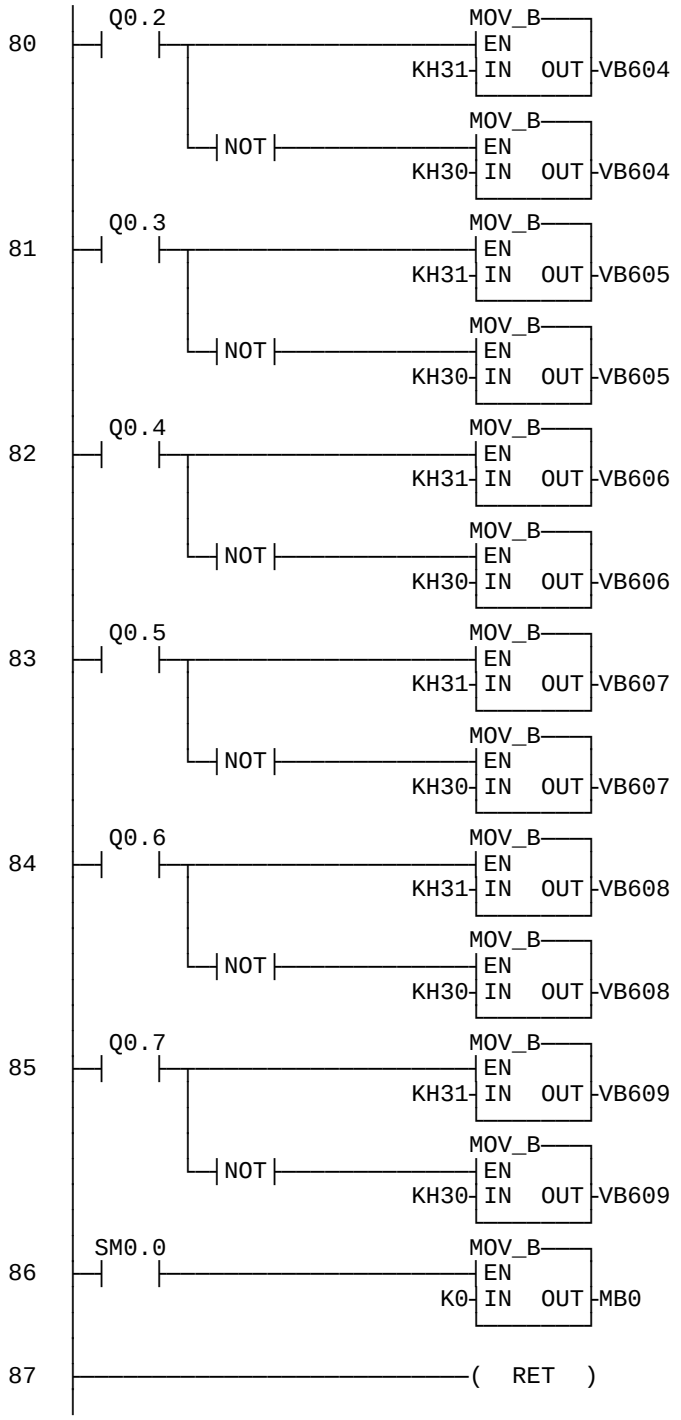



```
MOVB   16#31, VB567

NOT
MOVB   16#30, VB567

LD     I0.6
MOVB   16#31, VB568

NOT
MOVB   16#30, VB568

LD     I0.7
MOVB   16#31, VB569

NOT
MOVB   16#30, VB569

LD     SM0.0   // при каждом проходе:
MOVB   0, MB0  // ... очистка области
// функциональных клавиш

RET
```

```
///
// Подпрограмма 8: Отображение выходов PLC от Q0.0 до Q0.7
// состояние дисплея (дополнительное усройство 2) =1
// Эта подпрограмма читает  PLC Выходы от Q0. 0 до Q0. 7 и отображает их в двоичном,
// шестнадцатеричном и десятичном виде.
//
//
```




```
SBR    8

LD     SM0.0   // при каждом проходе:
HTA    QB0, VB612, 2   // Преобразование
// выходного байта из шестнадцатиричного
//в ASCII

MOVB   QB0, VB618
       // Отображение выходного байта в
       // десятичном виде

LD     Q0.0            // если выход
                       // включен,
MOVB   16#31, VB602    // пишет “1” на
// соответствующем месте дисплея

NOT                    // если выход
                       // выключен,
MOVB   16#30, VB602    // пишет“0” на
// соответствующем месте дисплея

LD     Q0.1
MOVB   16#31, VB603
```




```
NOT
MOVB   16#30, VB603

LD     Q0.2
MOVB   16#31, VB604

NOT
MOVB   16#30, VB604

LD     Q0.3
MOVB   16#31, VB605

NOT
MOVB   16#30, VB605

LD     Q0.4
MOVB   16#31, VB606

NOT
MOVB   16#30, VB606

LD     Q0.5
MOVB   16#31, VB607

NOT
MOVB   16#30, VB607

LD     Q0.6
MOVB   16#31, VB608

NOT
MOVB   16#30, VB608

LD     Q0.7
MOVB   16#31, VB609

NOT
MOVB   16#30, VB609

LD     SM0.0   // при каждом проходе:
MOVB   0, MB0  // ... очистка области
функциональных клавиш

RET
```

Блоки данных DB1 (V Память):

```
// Начало TD200_BLOCK 0
// (Комментарии внутри этого блока не должны быть отредактированы или удалены)
VB0    'TD'          // Идентификация TD 200
VB2    16#90         // Установка языка на Английский, установка такой быстрой
                     // модификации какой только возможно
VB3    16#71         // Установка дисплея в режим 40 символов ; лкавиша Up  V3.3;
клавиша
                     // Down  V3.2
VB4    7             // Установка числа сообщений
VB5    0             // Установка битов сообщения функциональных клавиш в M0.0 - M0.7
VW6    100           // Установка начального адреса сообщения в VW100
VW8    12            // Установка стартового адреса бита разрешения сообщения в VW12
VW10  1000           // Глобальный пароль
// СООБЩЕНИЕ 1
// Бит разрешения сообщения V12.7
VB100 '              ARROW KEYS TO SCROLL'
// СООБЩЕНИЕ 2
// Бит разрешения сообщения V12.6
VB140        'PRESSURE =    '
VB154        16#00        // Нет редактирования ; Нет подтверждения; Нет пароля;
VB155        16#43        // Двойное слово без знака;  3 цифры после запятой;
VD156        16#0000      // Вложенное значение данных: Перемещение данных для
                          // дисплея сюда.
VB160        'STOP  RUN  FAST SLOW'
// СООБЩЕНИЕ 3
// Бит разрешения сообщения V12.5
VB180        'OUTPUT CONFIGURATIONOFF   ON   CYCL     '
// СООБЩЕНИЕ 4
// Бит разрешения сообщения V12.4
VB220        '0   POT VALUE   '
VB236        16#00        // Нет редактирования ; Нет подтверждения; Нет пароля;
VB237        16#30        // Двойное слово без знака;  0 цифры после запятой;
VW238        16#00        //Вложенное значение данных: Перемещение данных для
                          // дисплея сюда.
VB240 '              '
// СООБЩЕНИЕ 5
// Бит разрешения сообщения V12.3
VB260        '  This is a test!   FAST  FA-  SL+  SLOW'
// СООБЩЕНИЕ 6
// Бит разрешения сообщения V12.2
VB300        'REAL VALUE:   '
VB314        16#18        // Сообщение редактирования V314.2; Нет подтверждения;
                          // редактирование требует пароль;
VB315        16#53        // Реальное двойное слово; 3 цифры после запятой;
VD316        16#0000      // Вложенное значение данных: Перемещение данных для
                          // дисплея сюда.
```

```
VB320          'Press ENTER to EDIT '
// СООБЩЕНИЕ 7
// Бит разрешения сообщения V12.1
VB340          'Motor        I0.0 =    Use I0.0 to toggle  '
// КОНЕЦ TD200_BLOCK 0

//НАЧАЛО TD200_BLOCK 500
// (Комментарии внутри этого блока не должны быть отредактированы или удалены)
VB500          'TD'              // Идентификация TD 200
VB502          16#90             // Установка языка - Английский, установка такой быстрой
                                 // модификации какая только возможна
VB503          16#71             // Установка режима дисплея 40 символов;клавиша Up V503.3;
                                 //клавиша Down V503.2
VB504          2                 // Установка количества сообщений
VB505          0                 // Установка битов сообщения функциональных клавиш в M0.0
–
                                 // M0.7
VW506          540               // Установка стартового адреса для сообщений в VW540
VW508          312               // Установка  стартового адреса бита разрешения сообщений в
                                 //  VW512
VW510          1000              // Глобальный пароль
// СООБЩЕНИЕ 1
// Бит разрешения сообщеня V512.7
VB540          'Display Inputs      I:XXXXXXXXb XXh'
VB575          16#00             // Нет редактирования ;Нет подтверждения; Нет пароля;
VB576          16#30             // Слово без знака;  0 цифр после запятой;
VW577          16#00             // Вложенное значение данных: Перемещение данных для
                                 // дисплея сюда.
VB579          'd'
// СООБЩЕНИЕ 2
// Бит разрешения сообщения V512.6
VB580          'Display Outputs     Q:XXXXXXXXb XXh'
VB615          16#00             // Нет редактирования ;Нет подтверждения; Нет пароля
VB616          16#30             // Слово без знака;  0 цифр после запятой;
VW617          16#00             // Вложенное значение данных: Перемещение данных для
                                 // дисплея сюда.
VB619          'd'
// КОНЕЦ TD200_BLOCK 500
```

Примечания преобразования

При преобразовании из IEC STL в S7-Micro/DOS STL:

- Добавлять ‘K’ перед всеми нешестнадцатиричными числовыми константами (т.е. 4 ⇒ K4)
- Заменять ‘16#’ на ‘KH’ для всех шестнадцатиричных констант (т.е. 16#FF ⇒ KHFF)
- Запятые обозначают разделение полей. Используйте стрелки или клавишу TAB, чтобы переключаться между полями.
- Чтобы преобразовывать S7-Micro/DOS программу STL в форму LAD, каждая сеть должна начинаться со слова 'СЕТЬ' и номера. Каждая сеть в Application Tip программы обозначена номером на ступенчатой диаграмме. Используйте команду INSNW под меню EDIT, чтобы ввести новую сеть Команды.MEND, RET, RETI, LBL, SBR, и INT каждая получает свою собственную сеть.
- Линия - комментариев, обозначенная '//' невозможна в S7-Micro/DOS, но Сеть - комментариев возможна.

Показанные блоки данных были созданы в Micro / Win; в Miicro / DOS использование редактора V-памяти для ввода блока данных (Обратитесь к руководству пользователя Micro / DOS). Обратите внимание что в Micro / Win:

- 16#.. обозначает шестнадцатиричные значения
- ‘*text*‘ представляет строковые значения

Общие замечания

SIMATIC S7-200 Советы по применению обеспечиваются для того, чтобы дать пользователям S7-200 некоторое представление относительно того, как, некоторые задачи из представления методики программирования, могут быть решены с этим контроллером. Эти инструкции не предполагают отражать все детали или разновидности оборудования, и при этом они не предусматривают любое возможное непредвиденное обстоятельство. Использование S7-200 Советов по применению свободное.
Siemens оставляет право делать изменения в спецификациях, показанных здесь или делать усовершенствования в любое время без примечания или обязательства. Это не освобождает пользователя от ответственности, при использовании звуковых методов в прикладной программе, установке, операции, и сопровождении приобретенного оборудования. Если возникает конфликт между общей информацией, содержащейся в этой публикации, содержанием рисунков и дополнительным материалом, или обоими, последние должны иметь приоритет.
Siemens не несет ответственности, по любой допустимой причине, в повреждениях или персональном ущербе, последующем из использования советов по применению.

Все права защищены. Любая форма дублирования или распространения, включая выдержки, разрешается только с конкретным разрешением SIEMENS.

SIMATIC S7-200Советы

Группа	Тема
3	**Использование XMT и RCV команды для связи с интерфейсом оператора**

CPU – необходимые для этого совета					
CPU 210 ☐	CPU 212 ☐	CPU 214 ☐	CPU 215 ☑	CPU 216 ☑	ДРУГИЕ ☐

Обзор

Этот пример объяснет, как использовать XMT и RCV команды для связи с интерфейсом оператора. Программа S7-200 выполняется на S7-216, связанном с PC (выполняя программу терминала) через PPI кабель. XMT команда используется, чтобы передать меню команды и информацию об ошибке. RCV команда используется, чтобы получить команды от программы терминала. Эта программа поддерживает две команды ON и OFF. Если введен ON, включаются первые 8 выводов на S7-216. Если выполнен OFF выключаются первые 8 выводов .

Обратите внимание: При этой записи RCV команда поддерживается только на 0 порте CPU S7-215 и 0 и 1 портах CPU.S7-216

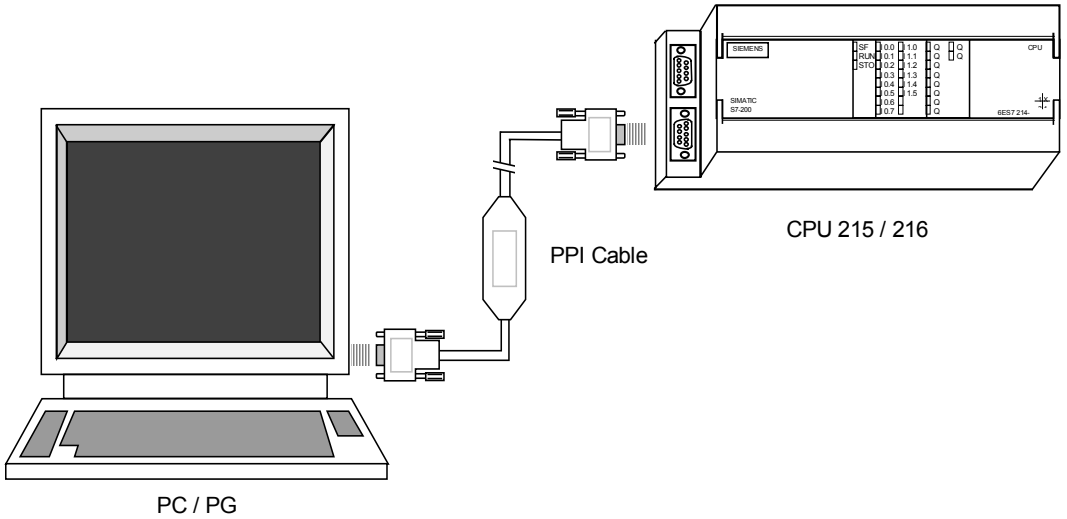


Рисунок 52.1 – Установка сети

Описание программы

Часть первая: Аппаратура

Интерфейс оператора
Эта программа использует программу терминала, которая отправлена с Windouws 3.1. Программа терминала должна быть конфигурирована следующим образом:

- DEC VT100 (ANSI)
- Локальное эхо
- 9600, 7 бит, контрольчетности, 1 стоповый бит
- Flow control -- none

Хост

- SIMATIC S7-200 CPU 215 или CPU 216
- PC/PPI Кабель соединенный с 0 портом S7216 и другим концом к одному из портов персонального компьютера RS-232 .
- PC/PPI кабель должен быть конфигурирован на 9600 бод. Переключатель, устанавливающий 0100 (т.е. SW2 вверх, все другие вниз)

Часть вторая: RCV команды

Специальные биты памяти, не определенные в текущем руководстве.
SM4.5 - Порт 0: Передатчик неактивен ('0' – идет передача; '1' – нет передачи)
SM4.6 - Порт 1: Передатчик неактивен ('0' - идет передача; '1' - нет передачи)

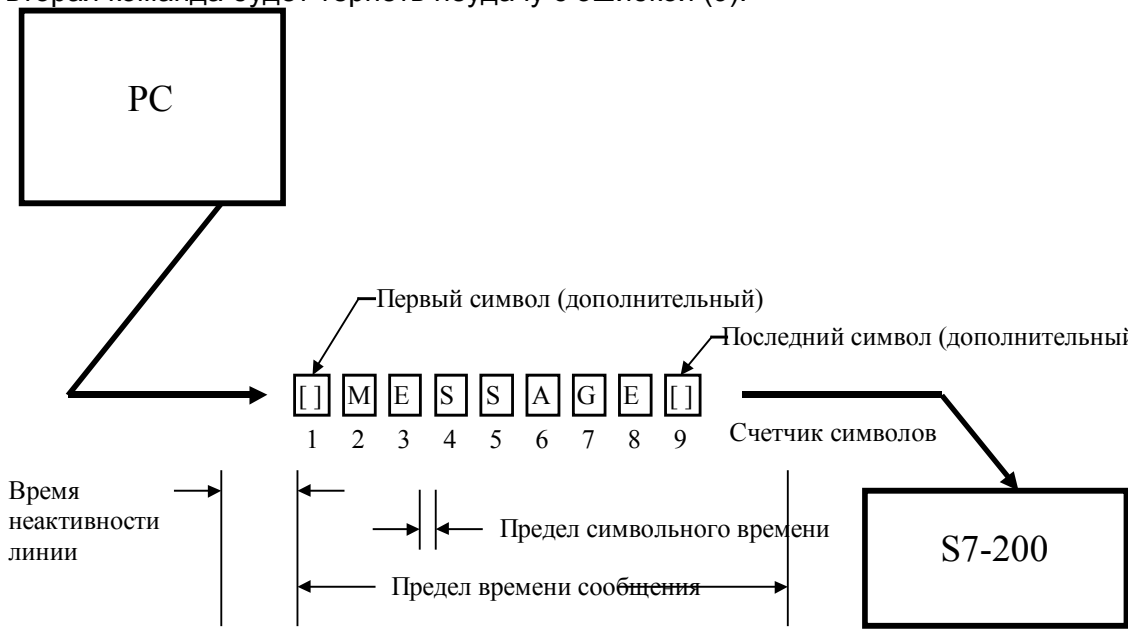
Каждый раз при выполнении RCV команды, поле счета в обозначенном буфере получения очищается. Все другие данные в буфере будут записаны как только будут получены новые данные.
PC/PPI кабель содержит допускаемый данными RS-232 для RS-485 конвертера. Если Вы используете этот кабель для Freeport связи, Вы должны позволить минимум 2 символьных промежутка между сообщениями противоположных направлений (получение напротив передачи). Вычисления в таблице основаны на 10 битовых символах (1 стартовый бит, 7 битов данных, 1 бит четности, 1 стоповый бит). Таблица ниже дает символьные времена для различных скоростей в бодах.

Скорость **В бодах**	**время1 символа Мсек**	**время 2 символов Мсек**
38400	0.26	0.52
19200	0.52	1.04
9600	1.04	2.08
4800	2.08	4.17
2400	4.16	8.34
1200	8.34	16.67
600	16.67	33.34
300	33.34	66.67

Чтобы вычислить время сообщения в миллисекундах используют следующие формулы:

Символьное_время = 1000 ms (скорость/число_бит_на_символ)
Время_сообщения = символьное_время * число_символов

Если имеется изменение в конфигурации порта (т.е.,четности, битов данных, или скорости в бодах), RCV команда сбросит порт и установит новую конфигурацию перед установкой получения символов.
Выполнение XMT и RCV команд взаимно исключается. Следующие комбинации команд дают ошибку если выполняются одновременно на одном и том же порте: XMT - XMT, RCV - RCV, XMT - RCV и RCV - XMT. Во всех случаях первая команда выполнится правильно а вторая команда будет терпеть неудачу с ошибкой (9).



VB640	Число полученных символов
VB641	Начальный символ (если выбран)
VB642	С
VB643	О
VB644	О
VB645	Б
VB646	ЩЕ
VB647	НИ
VB648	Е
VB649	Последний символ (если выбран)

Рисунок 52.2 - Условия сообщения

Должно быть по крайней мере одно условие начала и одно условие останова, определенное для выполнения RCV команды. Как только условие начала сообщения было выполнено, завершение сообщения должно произойти прежде, чем может быть получено другое сообщение. Условие рестарта не поддерживается.
Начальное и конечное условие:

- Начало_сообщения: Перерыв_неактивности_ линии_и_начальный_символ
- Конец_сообщения:Последний_символ ИЛИ Перерыв_сообщения/символа ИЛИ максимум_счетчика_символов ИЛИ ошибка_приоритета ИЛИ невозможная_команда

Чтобы прервать выполняющуюся RCV команду, очищается бит n (7) в байте управления для получения сообщения (SMB87 или SMB187, порт 0, порт, 1 соответственно) и затем выполняется RCV команда для рассматриваемого порта. Предыдущая RCV команда будет прервана, и будет установлен бит n (7) " Получение сообщения, завершено пользователем командой отключения " в байте получаемого сообщения (SMB86 или SMB186, порт 0, порт, 1 соответственно). Чтобы получить другое сообщение от рассматриваемого порта, будет необходимо повторно допустить функцию RCV для того порта, устанавливая бит n (7) в байте управления получаемого сообщения и выполняя RCV команду для определенного порта.

ПОРТ 0 ПОРТ 1	ИМЯ	ИСПОЛЬЗОВАНИЕ
SM4.5 SM4.6	Порт 0 нет передачи Порт 1 нет передачи	Используется для определения, является ли передатчик занятым. Используйте Этот бит используется для предотвращения выполнения двух XMT команд одновременно на одном и том же порте. Если две XMT команды выполняются одновременно на одном и том же порте, происходит ошибка.
SMB30 SMB130	Регистр управления режимом Freeport	Set up protocol and port communication parameters. XMT and RCV can only be used with the Freeport protocol.
SMB86 SMB186	Байт состояния получаемого сообщения	Use to determine which termination condition terminated the RCV instruction. MSB 7 … LSB 0 n 0 e 0 0 t c p
SMB87 SMB187	Байт состояния получаемого сообщения	Used to configure the start and stop conditions for receiving a message. Also used to abort a message. MSB 7 … LSB 0 n x y z m t 0 0
SM87.7 *SM187.7*	Допуск / отключение функции получения	The RCV instruction can only operate if it is enable. If the RCV instruction is in operation and this bit is cleared (Disabled) the execution of the next RCV instruction will terminate (abort) the Receive function. This termination will occur even if the message is in the process of being received.
SM87.6 *SM187.6*	Обнаружение первого символа сообщения (Условие старта)	Use the Value in SMB88/SM188 to detect start of message. This could be a character used to detect start of message or possibly the address of your node in a network (i.e., if this CPU is node 5, load a 5 into SMB88). Any message that does not start with a 5 will not be received.
SM87.5 *SM187.5*	Обнаружение последнего символа сообщения (Завершение сообщения)	Use the Value in SMB89/SMB189 to detect end of message. In an application where an individual was entering data the <carriage return character> might make a good end of message character. Make sure to use another form of termination such as inter-character timeout or end of message timeout just in case the end of message character isn't received.
SM87.4 *SM187.4*	Обнаружение условия неактивной линии (Условие старта)	Use the Value in SMW90 to detect an idle line condition. The first character received after idle line time has expired is the start of message. To further refine the start of message one can also specify a start of message character.
SM87.3 *SM187.3*	Печать таймера	Indicates timer type: Timer can either be a Inter-character timer or an End-of-Message timer. Typically shorter timeout values can be used if the Inter-Character-Timer is used.
SM87.2 *SM187.2*	Запуск таймера (Завершение сообщения)	Terminate receive if the time period in SMW92 or SMW192 is exceeded.
SMB94 SMB194	Максимальный размер буфера (Завершение сообщения)	Maximum number of buffers to be received. At least a count of 1 byte must be used. If a zero appears in this field the RCV instruction performs as a NOP.
Для дополнительной информации см. Приложение D, Биты Специальной Памяти (SM), в		

Часть третья: Программа

Эта программа связывается с интерфейсом оператора, используюя XMT и RCV команды. Машина состояний была выполнена для осуществления связи. Машина состояний состоит из 5 состояний связи (Сост. Связи):

- НЕАКТИВНАЯ (0)
- RCV (1)
- XMT (2)
- ОЖИДАНИЕ (3)
- ОШИБКА (4)

Обратитесь к рисунку 52-3 для графического представления как работает машина состояний.

При первом проходе инициализируются параметры порта , событие 'XMT совершена' соотносится с прерванием 0, событие 'RCV совершена' соотносится с прерванием 1, прерывания допускаются, машина инициализирована к ПРОСТОЮ, и меню пользователя передается к интерфейсу оператора.

Когда передача завершена происходит прерывание выполняемой RCV команды ,если машина состояний не находится в состоянии ошибки. Если машина состояний находится в ошибке, происходит возврат из прерывания. Так как XMT и RCV команды не могут быть выполнены одновременно, RCV команда должна быть защищена от выполнения, пока меню команды не будет передано. Только сообщения об ошибках передается в состоянии ошибки. Однако, сообщение об ошибках будет всегда сопровождаться командным меню. Если RCV команда должна была быть выполнена, в то время как машина состояний находится в ошибке, меню команды не может быть передано, так как XMT и RCV команды не могут выполняться одновременно. RCV команда должна победить, так как она будет выполнена сначала и меню команды никогда не будет передано.

Нормальная работа машины состояний простая и прямая:

- командное меню передается наружу,
- при полной передаче происходит прерывание,
- допускается функция получения,
- Переход машины состояний к НЕАКТИВНОМУ состоянию, ждет команду
- Команда получена,
- Получение завершено происходит прерывание
- Сообщение проверяется на ошибки получения
- Переход машины состояний в состояние получения
- Сообщение проверяется на команду поддержки
- Команда выполняется
- Переход машины состояний в состояний передачи
- передается меню команды

Обратите внимание: Состояние Связи 3 "ожидание 2 символов" необходим, только если используется PPI кабель.Ожидание 2 символов необходимо для того, чтобы направить связь вокруг. Если используется связь напрямую через RS485 кабель, состояние 3, может быть опущено.

Для дополнительной справки о XMT и RCV командах, см. Руководство Программируемого Контроллера Системы S7-200. Размер программы - 430 байтов.

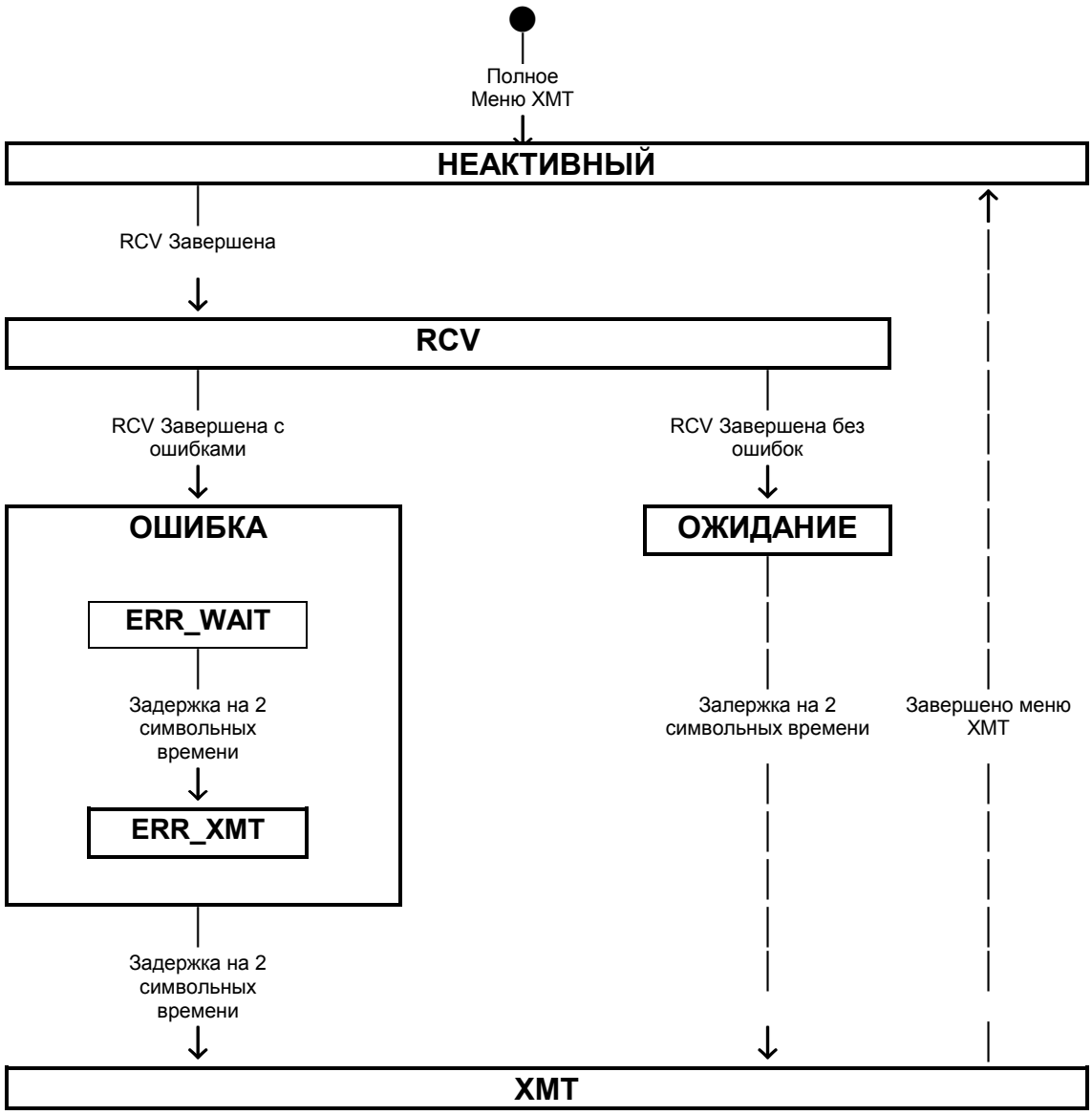
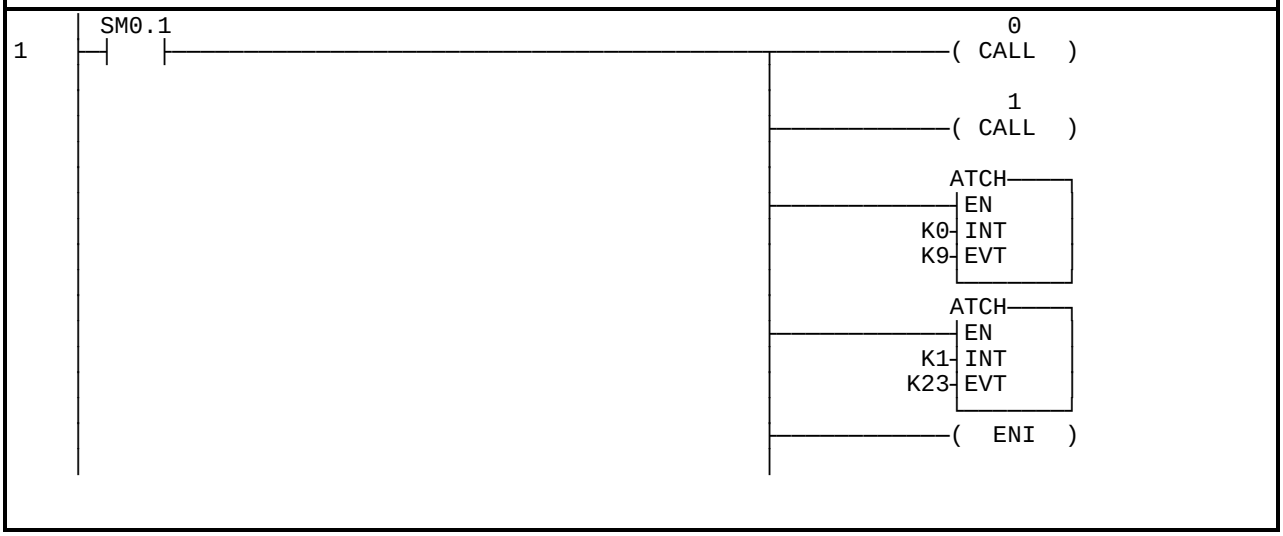


Рисунок 52.3 – машина состояний

Программа

```
//
// ЗАГОЛОВОК: Командный процессор
//
// ЦЕЛЬ:        Эта программа связывается с интерфейсом оператора
//              используя передчу и получение команд.
//
// Использование Памяти:
//   MB0     :  Байт ошибки
//   M0.7    :  Большой бит ошибки
//   M0.0    :  Недопустимая команда
//   SM0.1   :  Первый бит просмотра
//   SM4.5   :  Бит- передатчик занят
//   SMB30   :  Порт 0 Freeport байт управления режимом
//   SMB86   :  Порт 0 Байт состояния получаемого сообщения
//   SMB87   :  Порт 0 Байт управления получаемого сообщения
//   SMB89   :  Порт 0 " конечный символ сообщения " ( <CR> )
//              Условие останова
//   SMW90   :  Порт 0 Время простоя линии в миллисекундах.
//              Условие начала.
//   SMW92   :  Порт 0 Значение меж-символьного перерыва (1 секунда)
//              Условие ошибки
//   VB0     :  Сост.связи, 0=простой,1=получение,2=xmt,3=ожиданиет, 4=ошибка
//   VB1     :  Подсостояние ошибки, 1=ожидание, 2=xmt.
//   VB100   :  Сообщение об ошибках - завершение получения сообщения
//              по команде отключения пользователем
//   VB180   :  Сообщение об ошибках - получен последний символ
//   VB260   :  Сообщение об ошибках - Получаемое сообщение завершено:
//              таймер истек
//   VB340   :  Сообщение об ошибках - Получаемое сообщение завершено:
//              счетчик достиг максимального количества символов
//   VB420   :  Сообщение об ошибках - Получаемое сообщение завершено:
//              из-за ошибки четности
//   VB500   :  Сообщение об ошибках -недопустимая команда
//   VB580   :  Буфер команд - ввод команды (ON|OFF + <CR>):
//   VB640   :  Получение буфера
//   T32     :  Задержка на 2 символьных времени (5 msec)
//
```



```
                                                  MOV_B
                                                 EN
                                             K0 IN   OUT MB0

                                                  MOV_B
                                                 EN
                                             K0 IN   OUT VB1

                                                  MOV_B
                                                 EN
                                             K0 IN   OUT VB0

                                                  XMT
                                                 EN
                                          VB580 TBL
                                              0 POR
```

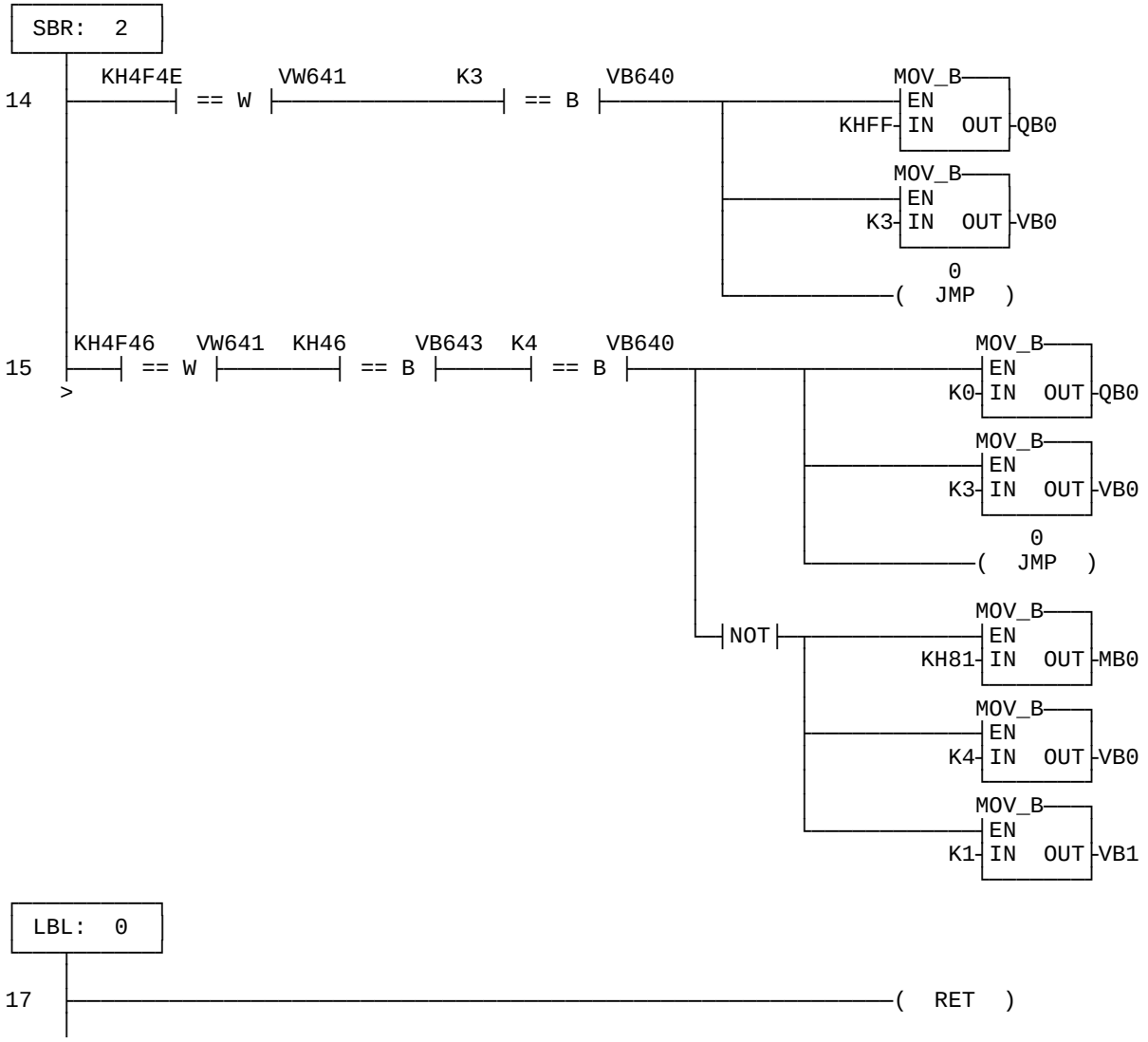
```
LD    SM0.1          // если(первый проход) {
CALL  0              //      Инициализация параметров связи порта 0
CALL  1              //      Инициализация параметров связи порта 0
ATCH  0,9            //      Прерывание 0 --> xmt завершена
ATCH  1,23           //      Прерывание 1 --> rcv завершена
ENI                  //      допуск прерываний
MOVB  0,MB0          //      Нет ошибок
MOVB  0,VB1          //      инициализация подсостояния ошибок в NULL
MOVB  0,VB0          //      инициализация состояния связи в IDLE
XMT   VB580,0        //      передача меню
                     // }
```

```
     VB0          K1                                          2
2  ──────┤ == B ├──────────────────────────────────────( CALL  )

     VB0          K2                                          3
3  ──────┤ == B ├──────────────────────────────────────( CALL  )

     VB0          K3                                          4
4  ──────┤ == B ├──────────────────────────────────────( CALL  )

     VB0          K4                                          5
5  ──────┤ == B ├──────────────────────────────────────( CALL  )

6  ──────────────────────────────────────────────────────( MEND  )
//
```

```
// Процессор машины состояний. Имеются только 5 состояний.
// Простой, Передача, Получение, Ожидание, и Ошибка.
//
LDB=  VB0,1          // если ( сост.связи = получение )
CALL  2              // процесс сообщения
LDB=  VB0,2          // если ( сост.связи = передача )
CALL  3              // процесс сообщения
LDB=  VB0,3          // если ( сост.связи = ожидание )
CALL  4              //     процесс ожидания 2 символов
LDB=  VB0,4          // если ( сост.связи = ошибка)
CALL  5              //     процесс ошибки
MEND
```

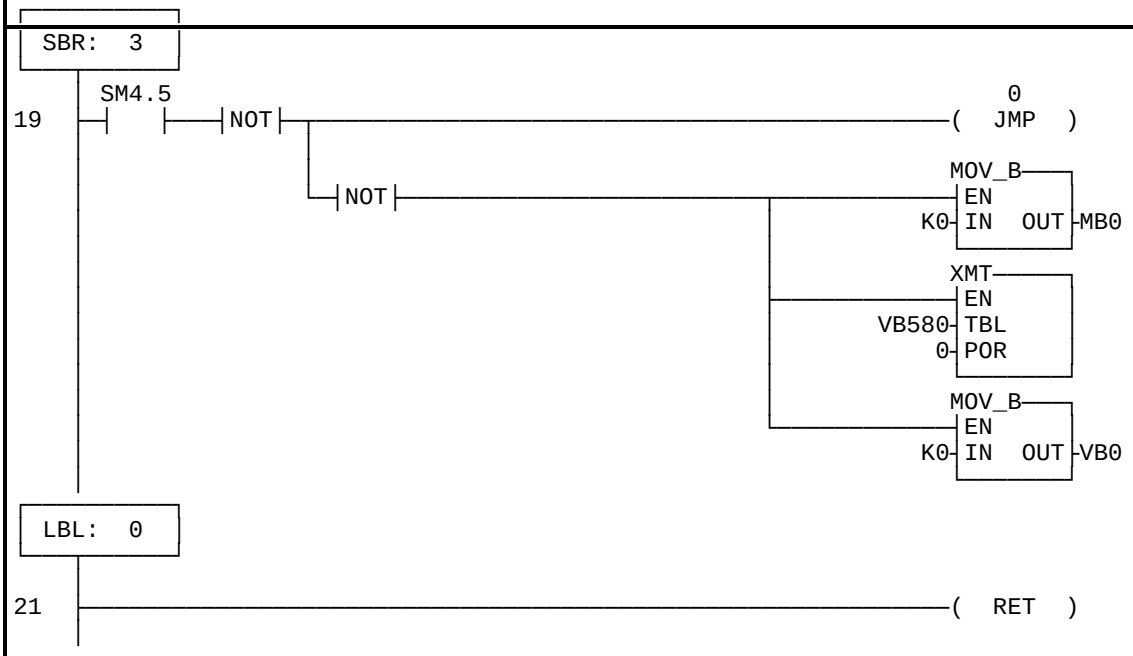
```
SBR:  0

8   ──────────────
```

```
MOVB   16#0D,SMB89          // Символ конца сообщения <CR>
MOVW   50,SMW90             // Перерыв простоя линии = 50 мсек
MOVW   4000,SMW92           // Меж-символьный перерыв = 4000 Мсек
MOVB   6,SMB94              // максимальное число символов = 6
RET
```




```
//
// Подпрограмма 2 – Процесс получения сообщения – СОСТ.СВЯЗИ 1
//
// Функция:   This state processes valid messages. Определяет поддерживается
//            команда или нет. Поддерживаются только 2 команды.
//            "ON" - включить 8 выходов
//            "OFF" - выключить 8 выходов.
//

SBR    2                    // В случае (КОМАНДА) {
LDW=   16#4F4E,VW641        //     ON : (команда==ON)  AND
AB=    3,VB640              //          (счетчик символов = 3)
MOVB   16#FF,QB0            //          включены 8 выходов
MOVB   3,VB0                //          следующее состояние-ожидание
```

```
JMP   0                    //
LDW=  16#4F46,VW641        //    OFF :(команда==OFF)
AB=   16#46,VB643          //        AND
AB=   4,VB640              //        (счеттчик символов = 4)
MOVB  0,QB0                //        выключены 8 выходов
MOVB  3,VB0                //        следующее состояние-ожидание
JMP   0                    //
NOT                        //    ERROR :
MOVB  16#81,MB0            //        ошибка = (ошибка + недопустимая команда)
MOVB  4,VB0                //        следующее состояние ошибки
MOVB  1,VB1                //        следующее подсостояние ошибки-ожидание
LBL   0                    // }      конец
RET                        // возврат
```




```
//
// Подпрограмма 3 –Передача сообщения – СОСТ.СВЯЗИ 2
//
// Функция:   Если передатчик не занят передает
//            command menu.
//
SBR   3
LD    SM4.5                // если (передатчик НЕ передает)
NOT                        // {
JMP   0                    //
```

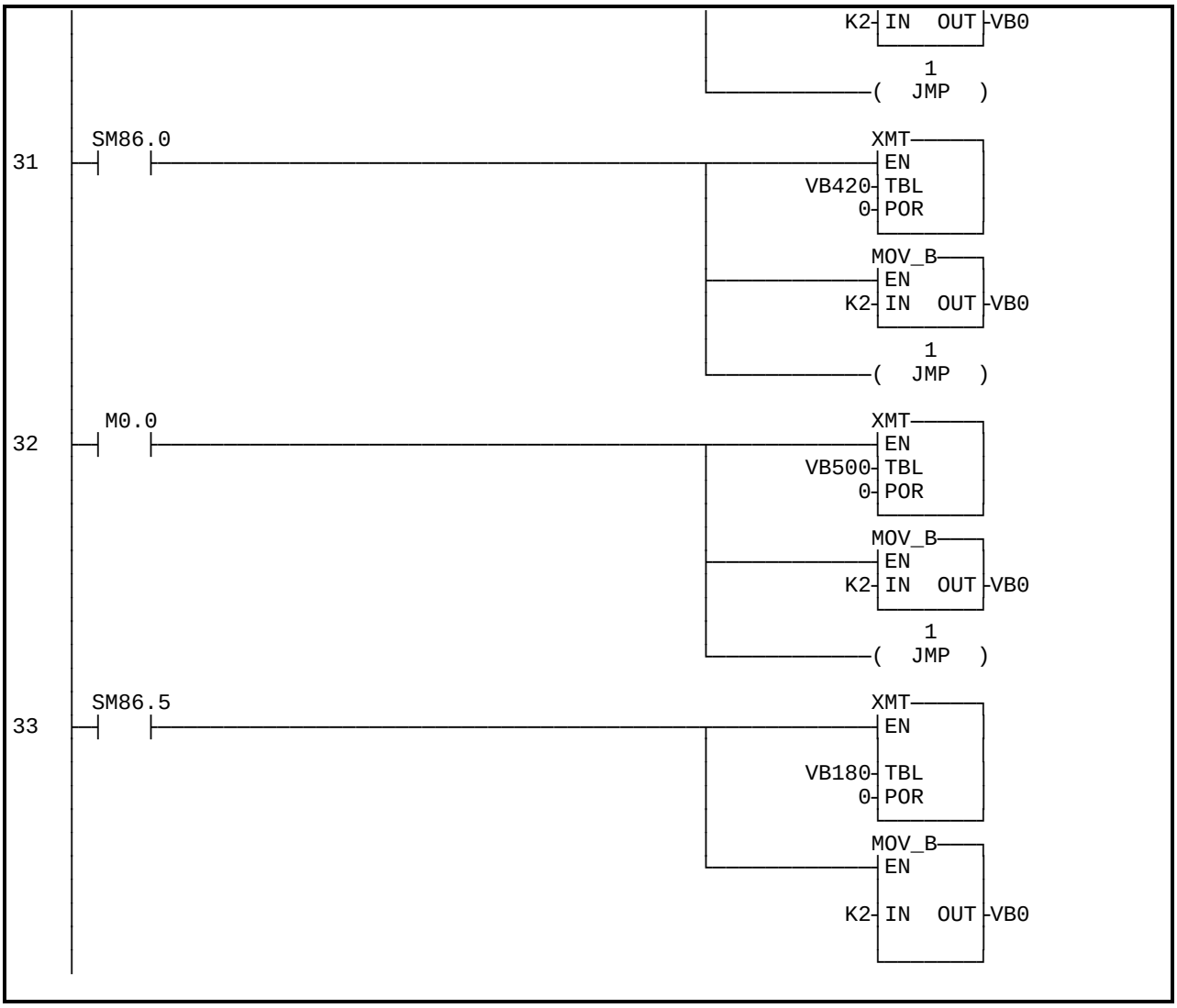
```
NOT                        //
MOVB  0,MB0                //    бит ошибки = Нет ошибки
XMT   VB580,0              //    передача меню
MOVB  0,VB0                //    следующее состояние-простой
LBL   0                    // }
RET                        // возврат
```

```
SBR:  4
    T32                                                    MOV_B
23 ─┤   ├──────────────────────────────────────────────────┤EN
                                                       K2─┤IN  OUT├─VB0

24 ──────────────────────────────────────────────────────────( RET )
```

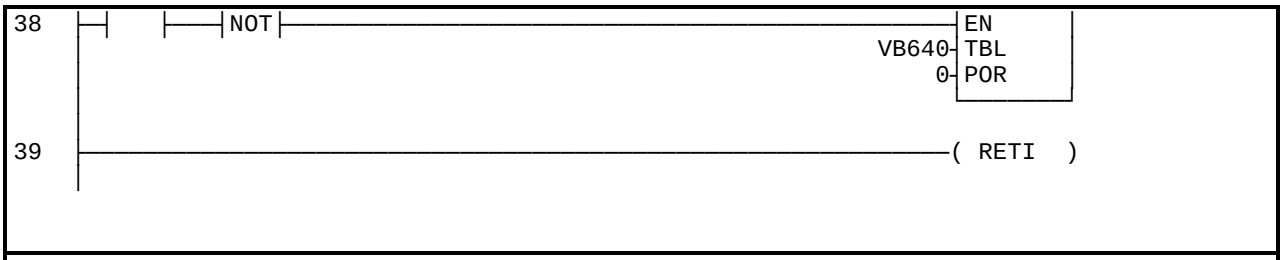
//

```
// Подпрограмма 4 --ОЖИДАНИЕ – СОСТ.СВЯЗИ 3
//
// Функция: Эта подпрограмма продвигают состояние связи в
//  передачу если истекла задержка 2 символов
//
SBR   4
LD    T32                    // если (перерыв)
MOVB  2,VB0                  //  Следующее состояние связи-передача
RET                          // возврат
```

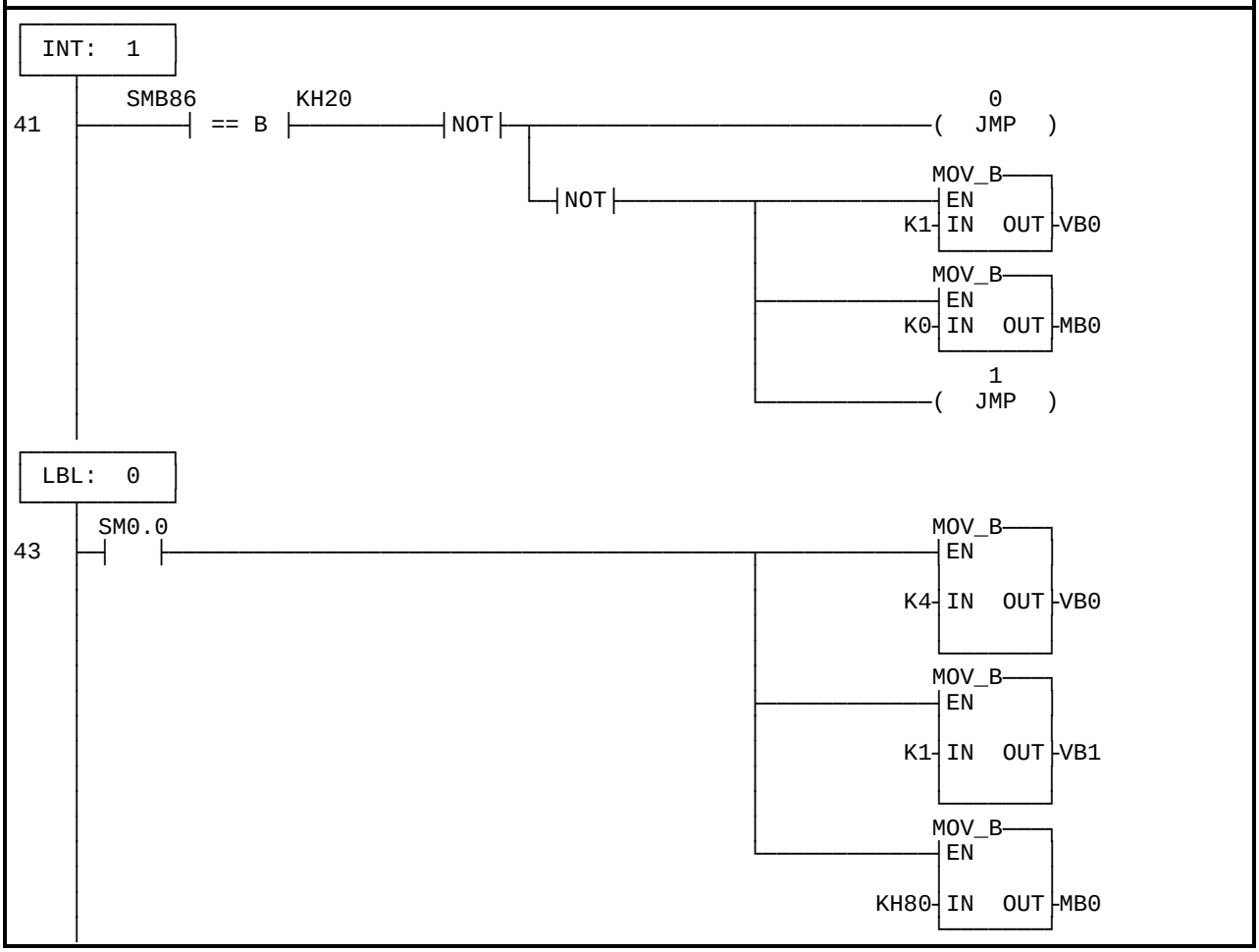
```
SBR:  5
    VB1          K1          T32                           MOV_B
26 ─────┤ == B ├──────────┤   ├──────────────────────────┤EN
                                                       K2─┤IN  OUT├─VB1

    VB1          K2                                          0
27 ─────┤ == B ├──────────┤NOT├──────────────────────────( JMP )

    SM86.7                                                 XMT
28 ─┤   ├──────────────────────────────────────────────┤EN
                                                   VB100─┤TBL
                                                       0─┤POR

                                                       MOV_B
                                                       ├─┤EN
                                                       K2─┤IN  OUT├─VB0

                                                           1
                                                       └─( JMP )

    SM86.2                                                 XMT
29 ─┤   ├──────────────────────────────────────────────┤EN
                                                   VB260─┤TBL
                                                       0─┤POR

                                                       MOV_B
                                                       ├─┤EN
                                                       K2─┤IN  OUT├─VB0

                                                           1
                                                       └─( JMP )

    SM86.1                                                 XMT
30 ─┤   ├──────────────────────────────────────────────┤EN
                                                   VB340─┤TBL
                                                       0─┤POR

                                                       MOV_B
                                                       ├─┤EN
```


K2 IN OUT VB0
1
(JMP)
31
SM86.0
XMT
EN
VB420 TBL
0 POR
MOV_B
EN
K2 IN OUT VB0
1
(JMP)
32
M0.0
XMT
EN
VB500 TBL
0 POR
MOV_B
EN
K2 IN OUT VB0
1
(JMP)
33
SM86.5
XMT
EN
VB180 TBL
0 POR
MOV_B
EN
K2 IN OUT VB0

```
 LBL:  1

 LBL:  0

36 ──────────────────────────────────────────────────────( RET )

//
// Подпрограмма 5 -- ОШИБКА – СОСТ.СВЯЗИ 4
//
// Функция: Если ошибка произошла на последнем получаемом сообщении
// то первая найденная ошибка передается.
// После передачи ошибки затем
//          следующее состояние-XMT.
//
// Внимание: VB1 означает только то, что состояние связи-ошибка

SBR   5
LDB=  VB1,1                  // если ( подсостояние ошибки-ожидание ) AND
A     T32                    //   (истек 2 символьный)
MOVB  2,VB1                  //   Следующее подсостояние-передача ошибки
LDB=  VB1,2                  // если ( подсотояние ошибки-XMT )
NOT                          // {
JMP   0                      //     в случае (ОШИБКА) {
LD    SM86.7                 //     запрещенная компнда :
XMT   VB100,0                //           передача ошибки
MOVB  2,VB0                  //           следующее состояни = передача
JMP   1                      //
LD    SM86.2                 //     Завершение передачи сообщения(таймер) :
XMT   VB260,0                //           ошибка передачи
MOVB  2,VB0                  //           следующее состояние = передача
JMP   1                      //
LD    SM86.1                 // Завершение передачи сообщения (счетчик символов) :
XMT   VB340,0                //           ошибка передачи
MOVB  2,VB0                  //           следующее состояние = передача
JMP   1                      //
LD    SM86.0                 // Завершение передачи сообщения (Четность) :
XMT   VB420,0                //           ошибка передачи
MOVB  2,VB0                  //           следующее состояние = передача
JMP   1                      //

LD    M0.0                   //     Недопустимая команда:
XMT   VB500,0                //          ошибка передачи
MOVB  2,VB0                  //           следующее состояние = передача
JMP   1                      //
LD    SM86.5                 //     Получен последний символ :
XMT   VB180,0                //           ошибка передачи
MOVB  2,VB0                  //           следующее состояние = предача
LBL   1                      //
LBL   0                      //
RET                          //


 INT:  0
    M0.7                                                    RCV
```




```
//
// ПРЕРЫВАНИЕ 0 -- Порт 0 передача завершена

INT   0
LD    M0.7                    // если (Нет ошибок)
NOT                           // {
RCV   VB640,0                 //    установка получения следующей команды
RETI                          // }
```



```
 LBL:  1
     SM0.0                                                    T32      K1
45  ─┤    ├──────────────────────────────────────────────────────────( R  )
                                                        │
                                                        │          TON──T32
                                                        └──────────┤IN     │
                                                                K5─┤PT     │
                                                                   └───────┘
46  ──────────────────────────────────────────────────────────────────( RETI  )
```

//

```
// ПРЕРЫВАНИЕ 1 -- Порт 0 звершено получение сообщения
//
INT   1
LDB=  SMB86,16#20          // если (статус==получен последний символ) { Нет ошибок
NOT                        // {
JMP   0                    //
NOT                        //
MOVB  1,VB0                //     следующее состояние-получение
MOVB  0,MB0                //     индикация отсутствия ошибок
JMP   1                    // }
LBL   0                    // иначе-установка бита ошибки
LD    SM0.0                // {
MOVB  4,VB0                //     следующее сотояние-ошибка
MOVB  1,VB1                //     следующее подсостояние ошибки-ожидание
MOVB  16#80,MB0            //     индикация произошедшей ошибки
LBL   1                    // }
LD    SM0.0                //
R     T32,1                // сброс 2 символьного таймера
TON   T32,5                // установка 2 символьного таймера на 5 мсек.
RETI
```

Блок данных: DB1

```
//
// XMT Формат сообщения для данного применения:
//                    [длина<LFCR>текст_сообщения]
//
VB100 51
VW101 16#0A0D
VB103 'Receive message terminated by user disable command'
//
VB180 24
VW181 16#0A0D
VB183 'End character received'
//
VB260 42
VW261 16#0A0D
VB263 'Receive message terminated: Timer Expired'
//
VB340 62
VW341 16#0A0D
VB343 'Receive message terminated: Maximum character count achieved'
//
VB420 52
VW421 16#0A0D
VB423 'Receive message terminated because of parity error'
//
VB500 17
VW501 16#0A0D
VB503 'Invalid Command'
//
VB580 32
VW581 16#0A0D
VB583 'Enter Command (ON|OFF + <CR>):'
//
```

При преобразовании из IEC STL в S7-Micro/DOS STL:

- Добавлять ‘K’ перед всеми нешестнадцатиричными числовыми константами (т.е. 4 ⇒ K4)
- Заменять ‘16#’ на ‘KH’ для всех шестнадцатиричных констант (т.е. 16#FF ⇒ KHFF)
- Запятые обозначают разделение полей. Используйте стрелки или клавишу TAB, чтобы переключаться между полями.
- Чтобы преобразовывать S7-Micro/DOS программу STL в форму LAD, каждая сеть должна начинаться со слова 'СЕТЬ' и номера. Каждая сеть в Application Tip программы обозначена номером на ступенчатой диаграмме. Используйте команду INSNW под меню EDIT, чтобы ввести новую сеть Команды.MEND, RET, RETI, LBL, SBR, и INT каждая получает свою собственную сеть.

- Линия - комментариев, обозначенная '//' невозможна в S7-Micro/DOS, но Сеть - комментариев возможна.

Показанные блоки данных были созданы в Micro / Win; в Miicro / DOS использование редактора V-памяти для ввода блока данных (Обратитесь к руководству пользователя Micro / DOS). Обратите внимание что в Micro / Win:

- 16#.. обозначает шестнадцатиричные значения
- ‘*text*‘ представляет строковые значения

Общие замечания

SIMATIC S7-200 Советы по применению обеспечиваются для того, чтобы дать пользователям S7-200 некоторое представление относительно того, как, некоторые задачи из представления методики программирования, могут быть решены с этим контроллером. Эти инструкции не предполагают отражать все детали или разновидности оборудования, и при этом они не предусматривают любое возможное непредвиденное обстоятельство. Использование S7-200 Советов по применению свободное.
Siemens оставляет право делать изменения в спецификациях, показанных здесь или делать усовершенствования в любое время без примечания или обязательства. Это не освобождает пользователя от ответственности, при использовании звуковых методов в прикладной программе, установке, операции, и сопровождении приобретенного оборудования. Если возникает конфликт между общей информацией, содержащейся в этой публикации, содержанием рисунков и дополнительным материалом, или обоими, последние должны иметь приоритет.
Siemens не несет ответственности, по любой допустимой причине, в повреждениях или персональном ущербе, последующем из использования советов по применению.
Все права защищены. Любая форма дублирования или распространения, включая выдержки, разрешается только с конкретным разрешением SIEMENS.

SIMATIC S7-200 Примеры

Группа	Тема
6	Использование PID-КОМАНДЫ

CPU , необходимые для этого примера					
CPU 210 ☐	CPU 212 ☐	CPU 214 ☐	CPU 215 ☑	CPU 216 ☑	OTHER ☐

Краткий обзор

Следующая программа S7-200 - короткий пример того, как использовать PID команду.
PID команда будет объясняться на примере резервуара с водой. Цель - поддерживать постоянное давление воды а также сохранять резервуар, чтобы не освободился.

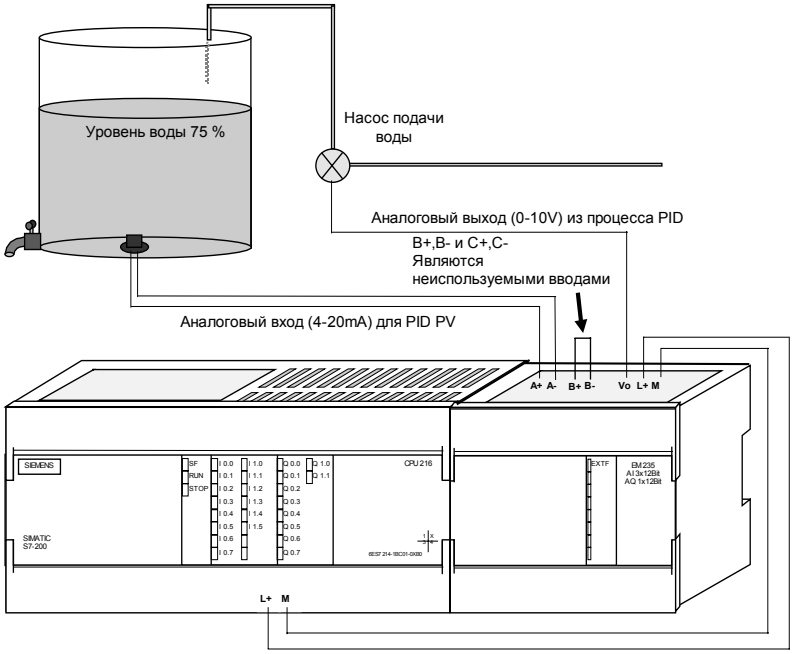


Рисунок 53.1

Аппаратные Требования

1 S7-200 PLC
1 Модуль расширения EM 235 Аналоговая комбинация AI 3/ AQ 1x 12 Bits

Общее описание

В этом примере, резервуар с водой используется, чтобы поддержать постоянное давление воды. Вода - постоянно берётся из резервуара в изменяющемся режиме. Используется насос переменной скорости, чтобы добавлять воду в резервуар в режиме, который поддержит адекватное давление воды а также сохранять резервуар, чтобы он не освободился.
Уставка для этой системы -установленный уровень воды, который является эквивалентным резервуару, наполненному на 75 %. Переменная процесса имеет формат с плавающей точкой, который обеспечивает эквивалентное считывание того, насколько полон резервуар и который может изменяться от 0 % (пустой) до 100 % (полностью полный). Вывод - значение скорости насоса, которое позволяет насосу работать со скоростью от 0 % до 100 % максимального значения.
Уставка предопределена и будет введена непосредственно в таблицу цикла. Переменная процесса будет обеспечена как аналоговое значение от 4 до 20 ma в формате с плавающей точкой. Вывод цикла будет записан в аналоговый вывод (от 0 до 10 V), который используется, чтобы управлять скоростью насоса. Промежуток обоих, аналогового входа и аналогового выхода - 32,000.
ВНИМАНИЕ: аналоговое значение (AIW0) может быть усреднено подпрограммой усреднения,чтобы фильтровать ввод прежде, чем значение будет использовано как PV. Обратитесь к примеру номер 54 для более детального пояснения.
В этом примере будет использоваться только пропорциональное и интегральное управление. Усиление цикла и константы времени были определены из разработки вычислений и могут быть откорректированы для достижения оптимального управления.
Расчетные значения констант времени:
KC - 0,25 TS - 0,1 секунд и TI - 30 минут.
Скорость резервуара будет управляться вручную, пока резервуар с водой наполнен на 75 %, затем клапан будет открыт, для того чтобы позволить воде выливаться из резервуара. В то же самое время, насос будет переведен в автоматический или ручной режим управления. Цифровой ввод будет использоваться,для того чтобы переключать управление от ручного к автоматическому. Этот ввод описан ниже:
I0. 0 - Ручное / автоматическое управление: 0 - ручное, 1 - автоматическое
В ручном режиме быстродействие насоса будет описано оператором VD108 как значение вещественного числа от 0.0 до 1.0.
Следующая информация описывает программу и её работу, определение переменных, подпрограмм и подпрограмм прерывания, используемых программой.

Подпрограммы:

SBR0 Подпрограмма инициализации
SBR1 Подпрограмма проверки наличия ошибок аналогового модуля

Подпрограммы прерывания:

INT 0 100 ms установленное прерывание, которое вызывает выполнение PID

Описание переменных:

I0.0 Ручное/ Автоматическое управление

Q0.7 Ошибка в аналоговом модуле

Таблица цикла - длиной 36 байтов, и параметры в таблице V-памяти - стандартные двойные слова (VD), таблица цикла имеет следующий формат:

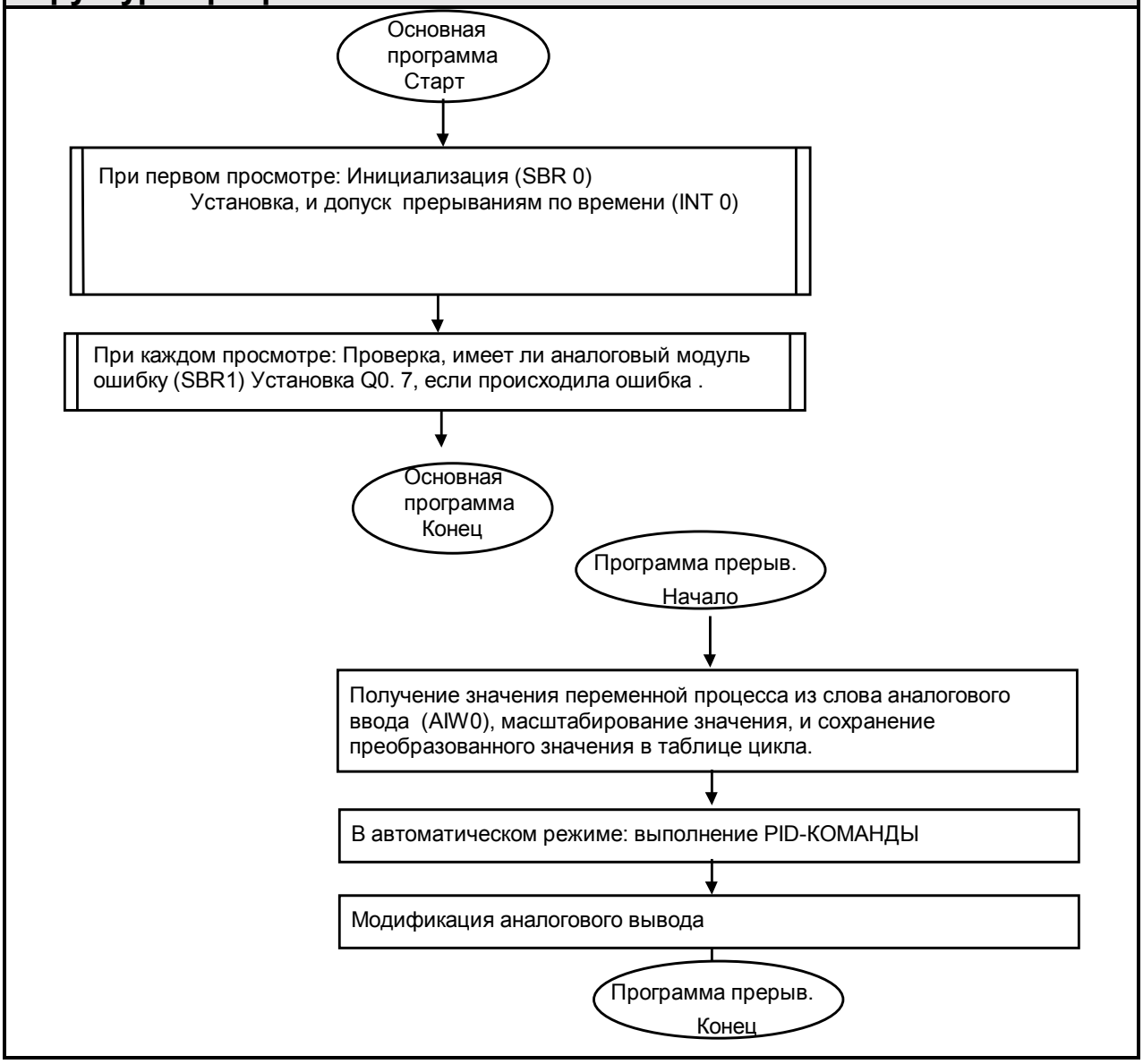
Смещение	Вычисление	Формат	Тип данных	Описание	Определяется пользователем
0	Переменная процесса	Двойное слово - реальное	In	Содержит переменную процесса, которая должна масштабироваться между 0.0 и 1.0	NO
4	Уставка	Двойное слово - реальное	I_N	Содержит уставку, которая должна масштабироваться между 0.0 и 1.0	YES
8	Вывод	Двойное слово - реальное	In/Out	Содержит расчетный вывод, Масштабированный между 0.0 и 1.0	NO
12	Усиление	Двойное слово - реальное	In	Содержит усиление, которое является пропорциональной константой. Может быть положительный или отрицательный номер.	YES
16	Типовое время	Двойное слово - реальное	In	Содержит типовое время, в секундах. Должен быть положительный номер.	YES
20	Интегральное время или сброс	Двойное слово - реальное	In	Содержит интегральное время или сброс, в минутах. Должен быть положительный номер.	YES
24	Производное время или режим	Двойное слово - реальное	In	Содержит производное время или режим, в минутах. Должен быть положительный номер	YES
28	Уклон	Двойное слово - реальное	In/Out	Содержит уклон или интегральное значение суммы Между 0.0 и 1.0	YES
32	Предыдущая переменная процесса	Двойное слово - реальное	In/Out	Содержит предыдущее значение переменной процесса, сохраненной из последнего выполнения PID команды.	NO

(Обратите внимание: базисный адрес программы примера - VD100.)

VD100 Переменная процесса (PV) , из аналогового ввода AWI0
VD104 точка отсчета (SP)
VD108 вывод (M), сщдержит расчитанный выход
VD112 Усиление (Kc), пропорциональная константа
VD116 Типовое время (Ts)
VD120 Интегральное время (Ti)
VD124 Производное время (Td) или режим
VD128 Уклон (MX) или интегральная сумма
VD132 Предыдущая переменная процесса (PVN-1)

Дополнительная информация относительно PID-команды содержится в руководстве по системе S7-200. Основы Управления с PID-регулятором объясняются в примере 32. Информация относительно аналогового модуля обеспечивается в руководстве по системе S7-200 и в примере 34.

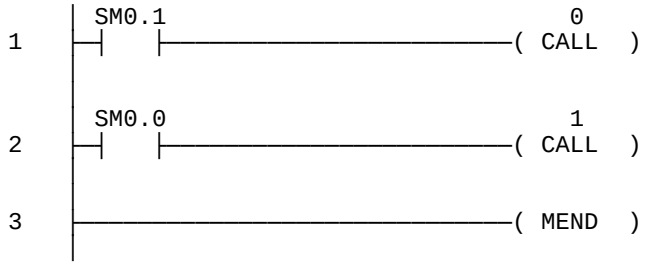
Структура программы



LAD (S7-MicroDOS)	STL (IEC)

Описание программы и основная программа

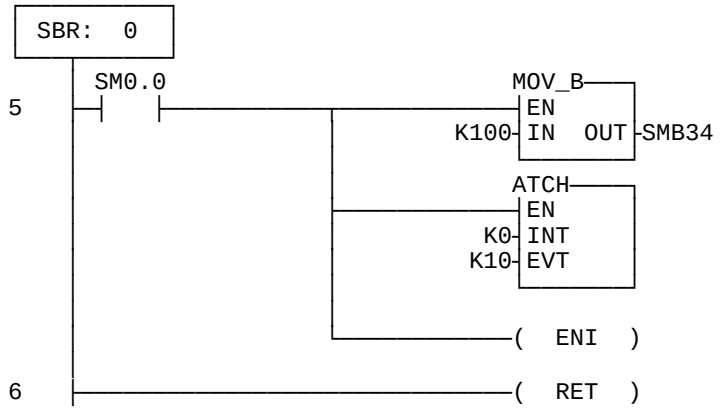
// PID Пример программы




```
LD      SM0.1   // При первом просмотре
CALL    0       // Вызов подпрограммы
                // инициализации

LD      SM0.0   // При каждом просмотре
CALL    1       // Проверка наличия ошибок
                //аналогового модуля

MEND            // Конец основной программы
```

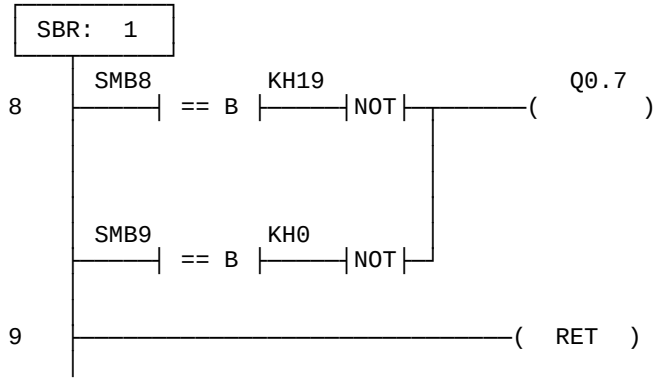
Подпрограммы

// SBR0: Инициализация (установка и допуск прерывания по времени)



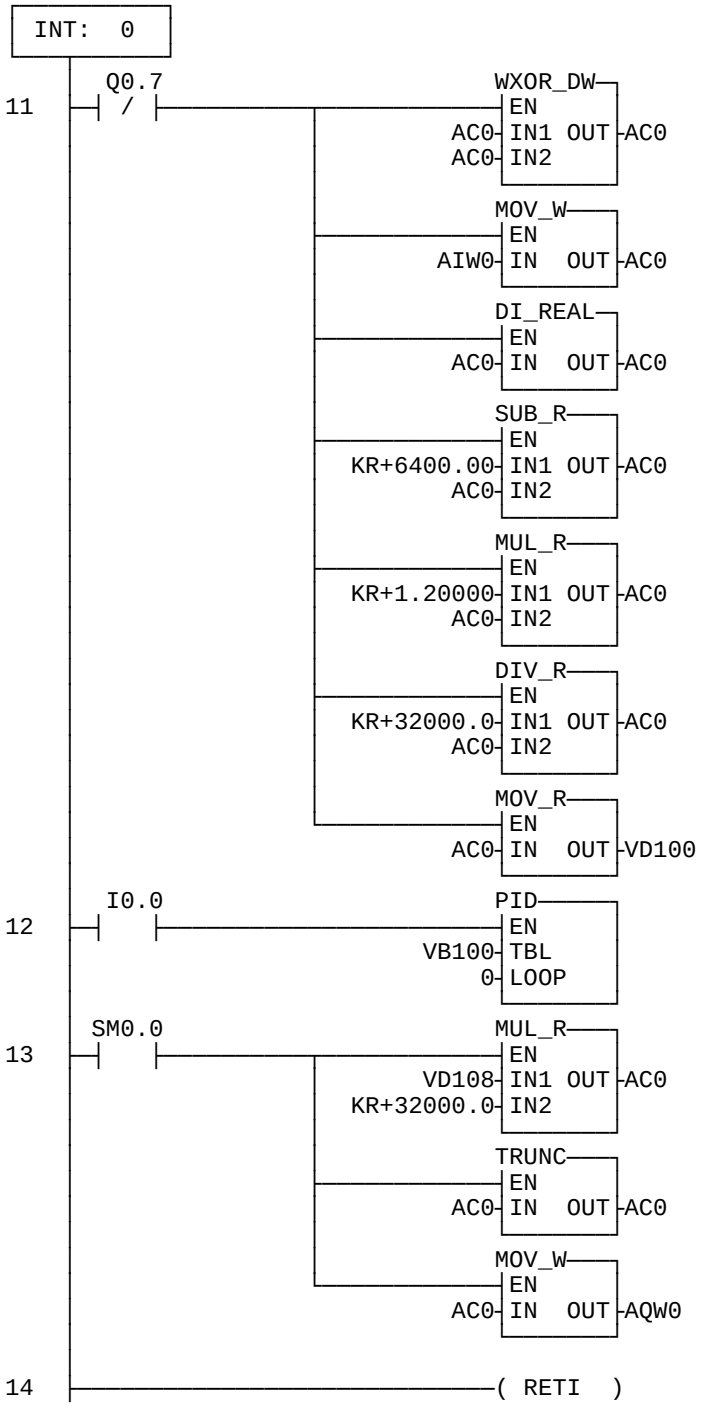

```
SBR  0

LD SM0.0
MOVB  100, SMB34
      // временной интервал (100ms)
      // для прерывания по времени

ATCH  0, 10
      //Установка прерывания для
      // вызова  выполнения PID

ENI   // Допуск прерываний

RET
```

// SBR1: Проверка ошибки в аналоговом модуле. Если ошибка происходит, установка Q0. 7.




```
SBR   1

LDB=  SMB8, 16#19     // Проверка,
NOT   // подключения аналогового
      // модуля

LDB=  SMB9, 16#0      // Проверка,
NOT   // наличия ошибки в модуле
OLD

=     Q0.7            // Ошибка чтения модуля
RET
```

// INT 0: Подпрограмма прерывания - PID- Подпрограмма (установленное прерывание для PID-ВЫПОЛНЕНИЯ).




```
INT   0

LDN   Q0.7      // МодульО.К.?
XORD  AC0, AC0
        // Очистка аккумулятора

MOVW  AIW0, AC0
        // сохранение аналогового
        //значения в аккумуляторе

DTR   AC0, AC0
        // Перевод 32-битового
        //значнения в вещественное

-R    6400.0, AC0
        // Коррекция смещения от 4 до 20
mA

*R    1.20, AC0
        // Масштаб для смещения 20 %

/R    32000.0, AC0        // Нормализация
        //PV значения в аккумуляторе

MOVR  AC0, VD100
        // Сохранение аккумулятора в
        //таблице цикла

LD    I0.0        // В авто режиме,
PID   VB100, 0    // вызов выполнения PID

// Аналоговый выход
LD    SM0.0
MOVR  VD108, AC0  // выход цикла в
        //акумулятор
*R    32000.00, AC0  // масштабирование
        // значения в аккумуляторе

TRUNC AC0, AC0        // Перевод
//вещественного значения в 32-битное
//целое

MOVW  AC0, AQW0
        // запись значения в аналоговый
        //выход

RETI
```

Блок данных DB1 (V Память):

```
// Инициализация значений
VD104    0.75    // точка отсчета = 0.75 = 75% наполнения
VD112    0.25    // Усиление цикла = 0.25
VD116    0.10    // Типовое время = 0.1 секунды
VD120    30.0    // Интегральное время = 30 минут
VD124    0.0     // никаких действий
```

Примечания преобразования

Для перевода из IEC STL в S7-Micro/DOS STL:

- Добавить ‘K’ перед всеми не шестнадцатиричными константами (например 4 ⇒ K4)
- Заменить ‘16#’ на ‘KH’ для всех шестнадцатиричных констант (например16#FF ⇒ KHFF)
- Запятые обозначают разделение полей. Используйтее стрелку или клавишу TAB для переходов между полями.
- Чтобы преобразовывать S7-Micro/DOS программу STL в форму LAD, каждая сеть должна начинаться со слова 'СЕТЬ' и номера. Каждая сеть в Application Tip программы обозначена номером на диаграмме . Используйте команду INSNW под меню РЕДАКТИРОВАНИЯ, чтобы ввести новую сеть. MEND, RET, RETI, LBL, SBR, и INT команды получают каждая свою собственную сеть.
- Линия - комментариев, обозначенная '//' не возмона в S7-Micro/DOS, но Сеть - комментариев возможна

Общие примечания

SIMATIC S7-200 Примеры применения предназначены для того, чтобы дать пользователям S7-200 некоторое представление относительно того, как, некоторые задачи программирования могут быть решены этим контроллером. Эти инструкции не учитывают все детали или разнообразие в оборудовании, и при этом они не предусматривают какое - либо непредвиденное обстоятельство. Использование S7-200 Примеров применения - свободное.

Siemens оставляет за собой право делать изменения в спецификациях, показанных здесь или делать усовершенствования в любое время без оповещения или обязательств. Это не освобождает пользователя от ответственности, за использование методов на практике, установке, обработке, и сопровождении приобретенного оборудования. Если конфликт возникает между общей информацией, содержащейся в этой публикации, содержанием рисунков или дополнительным материалом, или обоих, более поздние должны иметь приоритет.

Siemens не ответственен, по любой допустимой причине, в повреждениях или персональном ущербе, последующим из использования примеров применения.

Все права зарезервированы. Любая форма дублирования или распространения, включая выдержки, разрешается только с конкретным разрешением SIEMENS

SIMATIC S7-200 Примеры

Группа	Тема
6	**Демонстрационная версия, использующая PID-КОМАНДУ**

CPU необходимые для этого примера					
CPU 210 ☐	CPU 212 ☐	CPU 214 ☐	CPU 215 ☑	CPU 216 ☑	Другие ☐

Обзор

Следующая программа S7-200 - пример того, как использовать PID команду.
Пример включает интерфейс оператора для запуска TD200, подпрограмму моделирования процесса, которая обеспечивает большую задержку, и аналоговое значение вывода и цифровой вывод для управления циклом режима работы, и верхнего и нижнего пределов тревоги значения переменной процесса.

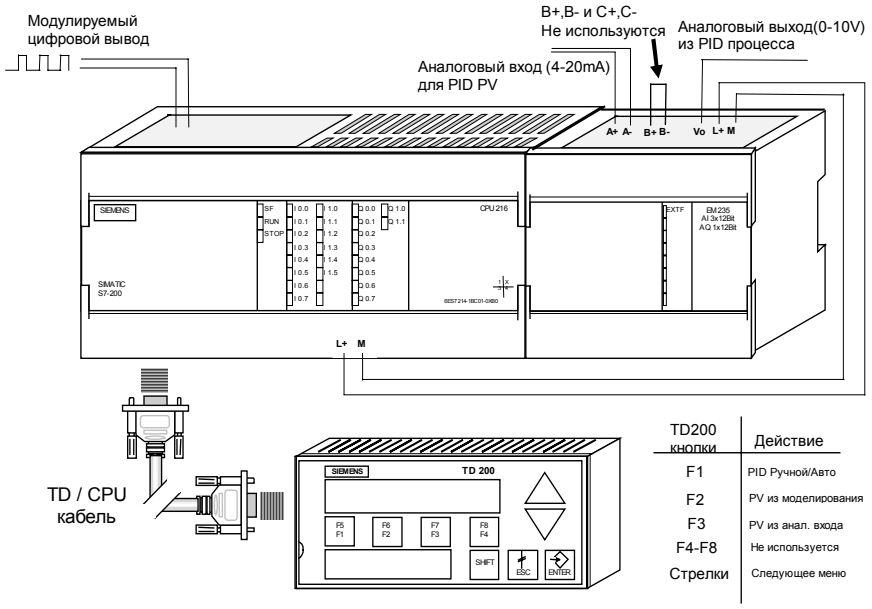


Рисунок 54.1

Аппаратные требования

1 S7-200 PLC
1 Расширенный модуль EM 235 Аналоговая комбинация AI 3/ AQ 1x 12 Бит
1 TD200s (версия 1.1 или следующие)
1 TD / CPU Кабель (включающий TD 200)

Общее описание

PID ввод цикла (Переменная процесса) может исходить из 2 различных параметров, выбираемых следующими функциональными клавишами TD200

F2 - Выбор опции ввода PV 1: Внутреннее моделирование процесса (от 4 до 20 milliamp)

F3 - Выбор опции ввода PV 2: AIW0 (от 4 до 20 milliamp)

Аналоговое значение (AIW0) усредняется прежде, чем оно используются как PV. Чтобы отфильтровывать шум, подпрограмма усреднения вычисляет действующее среднее значение (более чем 64 выборки) аналогового ввода прежде, чем оно используется как PV.
Подпрограмма моделирования процесса генерирует значение, которое соответствует сигналу от 4 до 20 milliamp, произведимоому аналоговым входным модулем. Все эти входные параметры корректируются подпрограммой прерывания по времени, которая выполняет цикл один раз каждые 100 миллисекунд, чтобы нормализовать PV.
Верхние и нижние пределы тревоги проверяются внутри подпрограммы прерывания по времени, которая вызывает выполнение PID команды. Верхнее условие тревоги отображается на Q0. 0, а нижнее условие тревоги отображается на Q0.1.
Вывод цикла записывается в AQW0, чтобы обеспечить аналоговое значение вывода. Кроме того вывод цикла масштабируется к целочисленному значению между 0 и 100, чтобы обеспечить управление циклом режима работы в приращениях 1 %. Период управления циклом режима работы - 1 секунда.
Автоматический / ручной режим PID команды управляется функциональной клавишей F1 на TD200. Каждое нажатие этой клавиша переключает режим. Режим сохраняется в M0. 0 и этот бит используется, чтобы разрешать или запрещать выполнение PID команды. PID команда обеспечивает бескофликтную передачу управления от ручного к автоматическому при помощи записи значения вывода в таблицу цикла прежде, чем режим переключается.

Следующая информация описывает программу и ее работу, определение переменных, подпрограмм и подпрограмм прерывания, используемых программой.

Подпрограммы:

SBR50	Вычисление действующего среднего значения AWI0 (более чем 64 выборки)
SBR 58	Подпрограмма проверки диапазона/ правильности
SBR 59	Подпрограмма переключения Автоматического / ручного режима
SBR 60	TD200 Подпрограмма
SBR 61	Подпрограмма цифрового вывода
SBR 62	Подпрограмма моделирования процесса (Дополнительная)
SBR 63	Подпрограмма инициализации

Подпрограммы прерывания:

INT 127	100 время прерывания, которое вызывает PID выполнение

Описание переменных:

Q0.0	Выход тревоги по верхнему уровню
Q0.1	Выход тревоги по нижнему уровню
Q0.2	PV ввод опции 1: Внутреннее моделирование процесса - Выбран
Q0.3	PV ввод опции 2: AIW0 - Выбран
Q0.4	Цифровой вывод (цикл управления режимом работы)
Q0.7	Ошибка в аналоговом модуле
M0.0	PID индикатор автоматического режима (0 - Авто; 1 - Ручной)
M0.1	1 секунда сброса таймера
M0.2 - M0.7	зарезервированные
M1.0	TD200 Функциональная клавиша F1 (Переключает Авто / ручной режим PID)
M1.1	TD200 Функциональная клавиша F2 - Выбор опции ввода PV 3: Внутреннее моделирование процесса
M1.2	TD200 Функциональная клавиша F3 - Выбор опции ввода PV 2: AIW0
M1.3	TD200 Функциональная клавиша F4 – не используется
M1.4	TD200 Функциональная клавиша Shift F1 - Не используется
M1.5	TD200 Функциональная клавиша Shift F2 - Не используется
M1.6	TD200 Функциональная клавиша Shift F3 - Не используется
M1.7	TD200 Функциональная клавиша Shift F4 - Не используется
VB0 - VB11	TD200 параметр блока
VB12	байт возможного сообщения
V12.7	0 бит возможного сообщения
V12.6	1 бит возможного сообщения
V12.5	2 бит возможного сообщения
V12.4 - V12.0	не используется
VB13	Стартовый байт 0 сообщения - A_PV_LA__SP____OUT_ 0.xxx _0.xxx 0.xxx
VB13	Байт индикации Авто/Ручного режима сообщения
VW18	Два байта, содержащие текст для PV в сигнализации (изменяемый программой)
VD35	Переменная #1 - PV
V40.2	Переменная #2 - бит сообщения редактирования

V40.4 Переменная #2 служебный бит разрешения редактирования (0 - нельзя редактировать; 1 - можно редактировать)
VD42 Переменная #2 - SP

V46.2 Переменная #3 - бит сообщения редактирования
V46.4 Переменная #3 - бит разрешения редактирования (0 - нельзя редактировать; 1 - редактирование позволено)
VD48 Переменная #3 - Вывод цикла
VB53 Стартовый байт сообщения 1 - __Kc_____Ti____Td__
xxx.xx__xx.xx_xx.xx

V73.2 Переменная #1 - бит сообщения редактирования
V73.4 Переменная #1 - бит разрешения редактирования (0 - нельзя редактировать; 1 - редактирование позволено)
VD75 Переменная #1 - Kc

V80.2 Переменная #2 - бит сообщения редактирования
V80.4 Переменная #2 - бит разрешения редактирования (0 - нельзя редактировать; 1 - редактирование позволено)

VD82 Переменная #2 - Ti

V86.2 Переменная #3 – бит сообщения редактирования
V86.4 Переменная #3 - бит разрешения редактирования (0 - нельзя редактировать; 1 - редактирование позволено)

VD88 Переменная #3 - Td

VB93 Стартовый байт Сообщения 2 - Hi_Alarm____Lo_Alarm
0.xxx_____0.xxx___

V113.2 Переменная #1 - бит сообщения редактирования
V113.4 Переменная #1 - бит разрешения редактирования (0 - нельзя редактировать; 1 - редактирование позволено)
VD115 Переменная #1 - Высокое ограничение сигнала

V125.2 Переменная #2 - бит сообщения редактирования
V125.4 Переменная #2 - бит разрешения редактирования (0 - нельзя редактировать; 1 - редактирование позволено)

VD127 Переменная #2 - Нижний предел тревоги

VW 206 MSB производит выборку значения
VW208 Типовое значение
VW210 Среднее Значение

VD206 рабочая область
VD212 Действующая сумма
VD216 рабочая область

VW4024	PV как значение от 6400 до 32000 (4 до 20 ma ввод) Из моделирования
VD4026	Резервированный
VW4030	Цифровое значение вывода (под циклическим контролем)
VW4032	Цифровое значение вывода (под циклическим контролем) 1 кадр в секунду
VD4036	Резервированный
VD4040	Резервированный
VD4044	Максимальная входная температура в °C
VD4048	Нижний предел тревоги
VD4052	Верхний предел тревоги
VD4056	Переменная задержки используемая в моделировании
VB4060	Таблица цикла
VD4060	PV
VD4064	SP
VD4068	ВЫХОД
VD4072	Усиление
VD4076	Типовое Время
VD4080	Суммарное Время
VD4084	Производное Время
VD4088	Уклон
VD4092	Предыдущий PV

TD200 Сообщения:

TD200 обеспечивает следующим три вида сообщений:

Сообщение № 1

Сообщение № 2

Интегральное время (Сброс)

Усиление

Производное время (Режим)

		K	c						T	i					T	d		
		0	.	0	0				0	.	0	0			0	.	0	0

Усиление(редактир.)

Интегральное время (редактир.)

Производное время(Редактир.)

Сообщение № 3

Верхний предел

Нижний предел

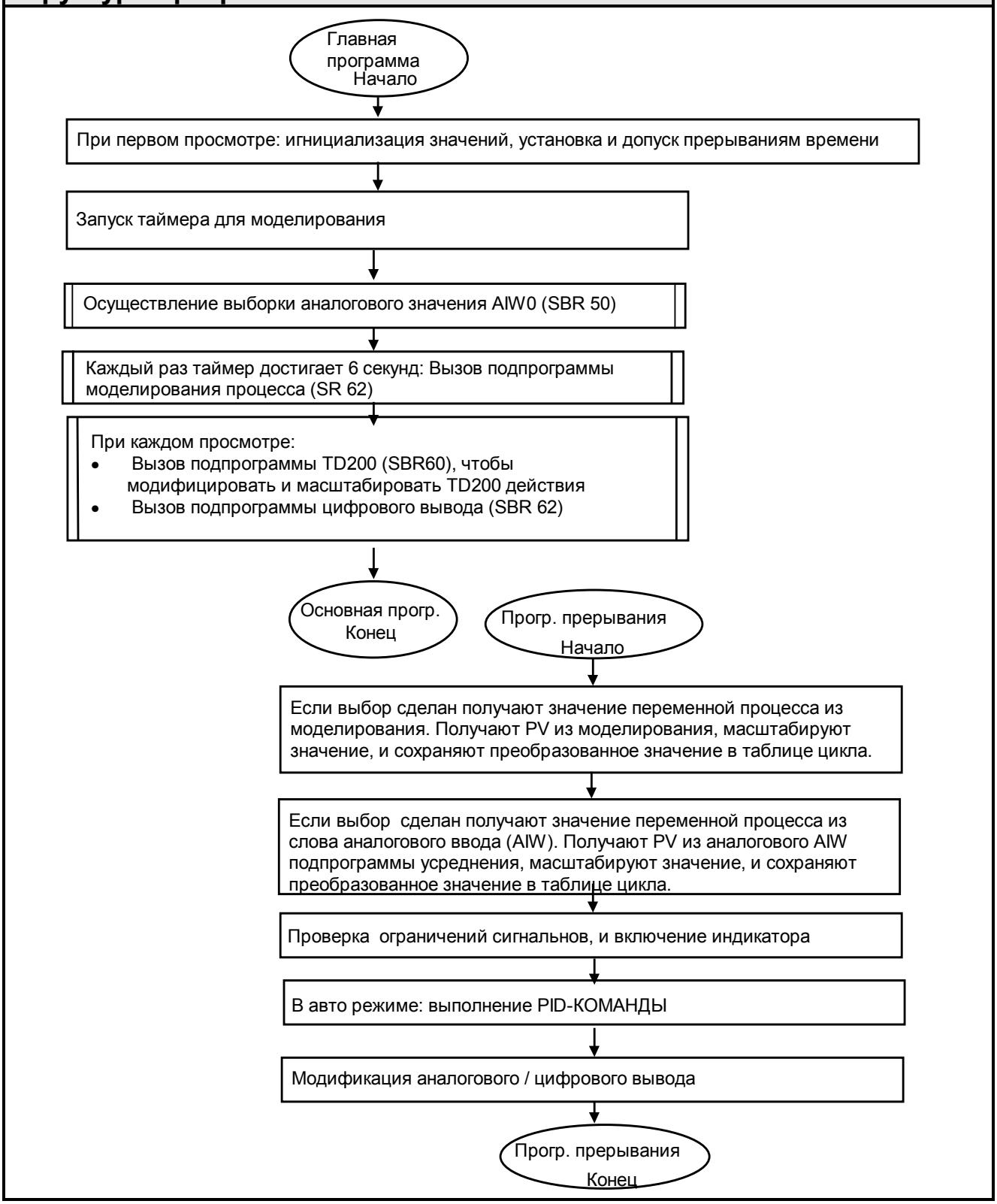
H	i		A	l	a	r	m					L	o		A	l	a	r	m
		0	.	0	0									0	.	0	0		

Значение ВП (Редактируемое)

Значение НП (Редактируемое)

Дополнительная информация относительно PID Команды обеспечивается в S7-200 руководстве Системы. Основы работы с PID объясняются в примере 32.
Информация относительно аналогового модуля обеспечивается в S7-200 руководстве Системы и в примере 34.

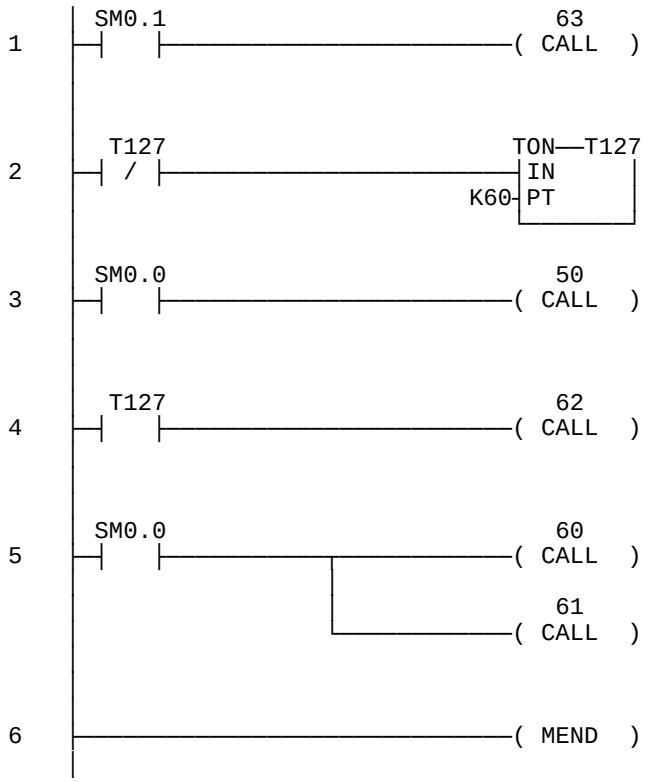
Структура программы



LAD (S7-MicroDOS)	STL (IEC)

Описание программы и главная программа

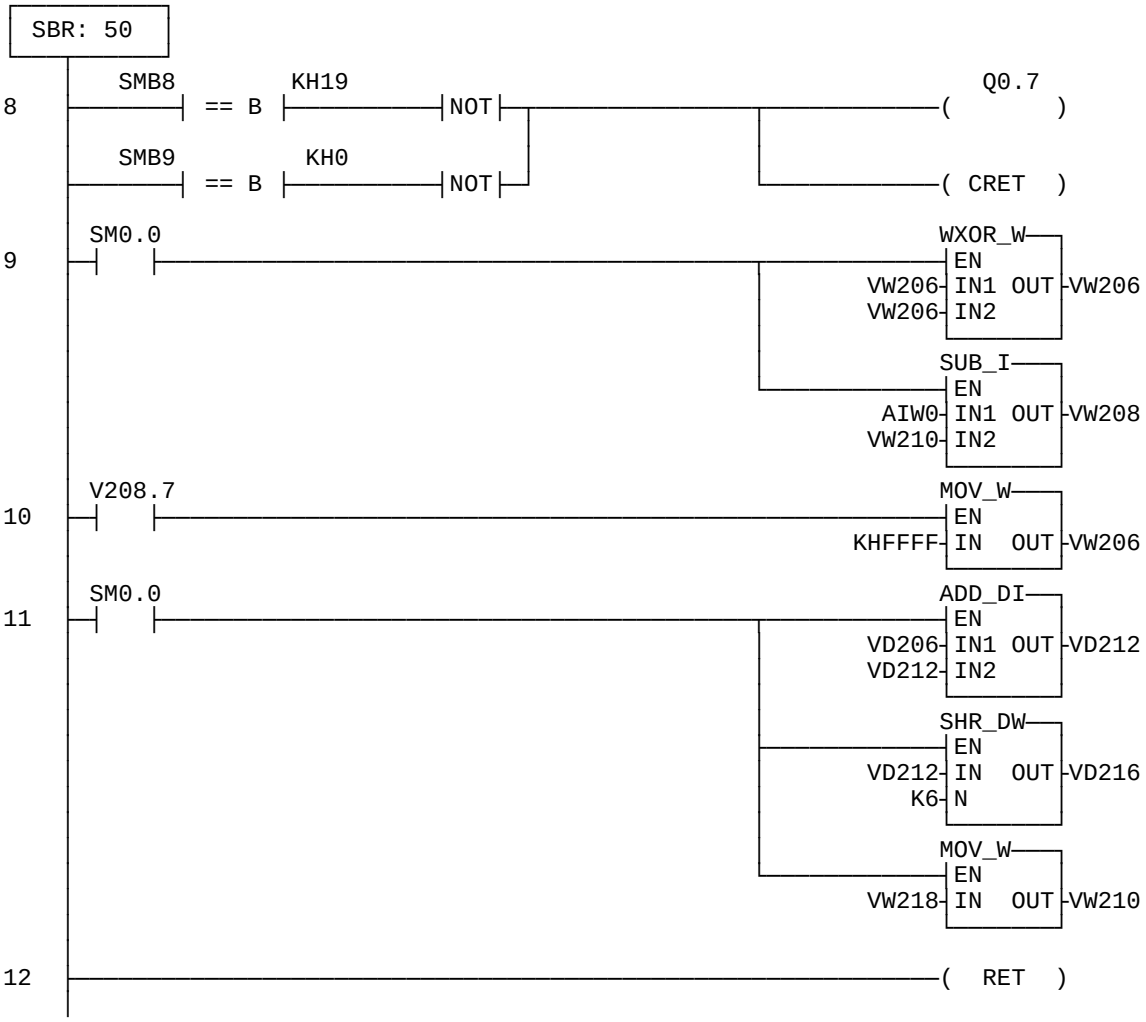
```
//
// PID Демонстрационная программа
//
```




```
LD      SM0.1   // При первом проходе
CALL    63      // Вызов подпрограммы
                // инициализации

LDN     T127    // Запуск  6 сек таймера для
                // моделирования
TON     T127, 60

LD      SM0.0   // При каждом проходе
CALL    50      // Вызов подпрограммы
                // усреднения

LD      T127    // При достижении таймером
                // 6 секунд,
CALL    62      // Вызов подпрограммы
                //моделирования процесса

LD      SM0.0   // Каждый проход
CALL    60      // Вызов
                // подпрограммыTD200
CALL    61      // Вызовподпрограммы
                // цифрового выхода

MEND            // Завершение главной
        // программы
```

Подпрограммы

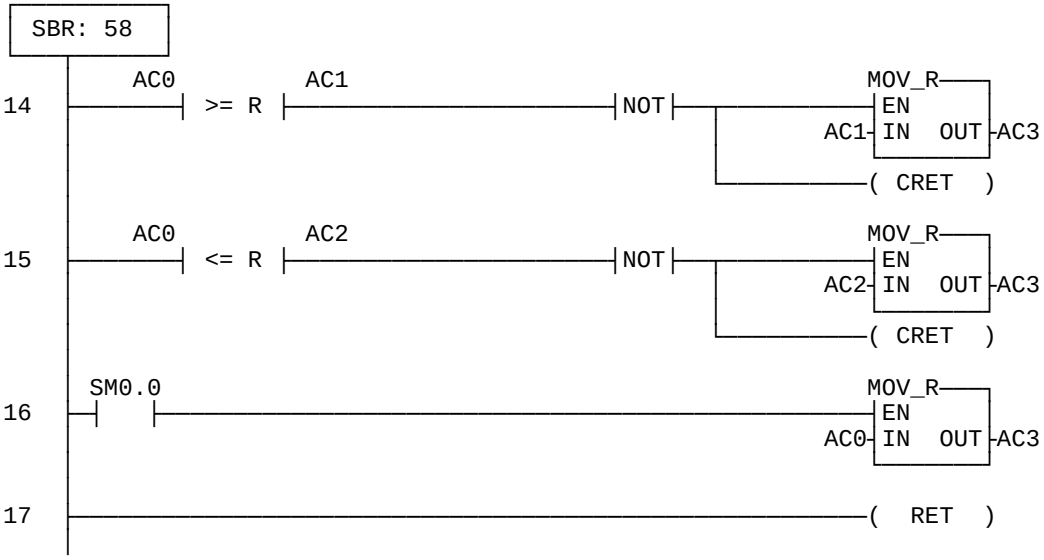
// SBR 50: Выполнение вычисления среднего значения AWI0 (более чем 64 выборки)
// Команда сдвига используется для разделения действующей суммы необходимым числом
//выборок
// (как степень 2). Увеличивать количество выборок с числом сдвигов. Например, сдвиг на 7
// позиций 2^7 = 128 выборок.




```
SBR     50
LDB= SMB8, 16#19            // Проверка включен ли
NOT                         // аналоговый модуль
LDB= SMB9, 16#0             // Проверка наличия ошибки
NOT                         // аналогового модуля
OLD
=       Q0.7                // Ошибка чтения аналогового модуля
CRET                        // "Заморозка" средненго аналогового значения

LD SM0.0                    // При каждом проходе
XORW VW206, VW206           // Стирание старшего слова значения в виде двойного слова
MOVW AIW0, VW208            // Получение типового значения
```

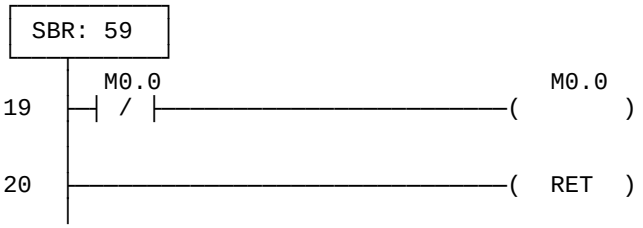
```
-I VW210, VW208          // Получение разницы между типовым и
                         // средним значением

LD V208.7                // Если отрицательная,
MOVW 16#FFFF, VW206      // Знак распространяется на значение в виде двойного слова

LD SM0.0                 // При каждом проходе
+D VD206, VD212          // Изменение действующей суммы
MOVD VD212, VD216        // Копирование рабочей области
SRD VD216, 6             // разделение действующей суммы необходимыми
выборками (2^6)
MOVW VW218, VW210        // сохранение среднего значения

RET
```

// Подпрограмма 58: TD200 Подпрограмма проверки диапазона (дополнительная)



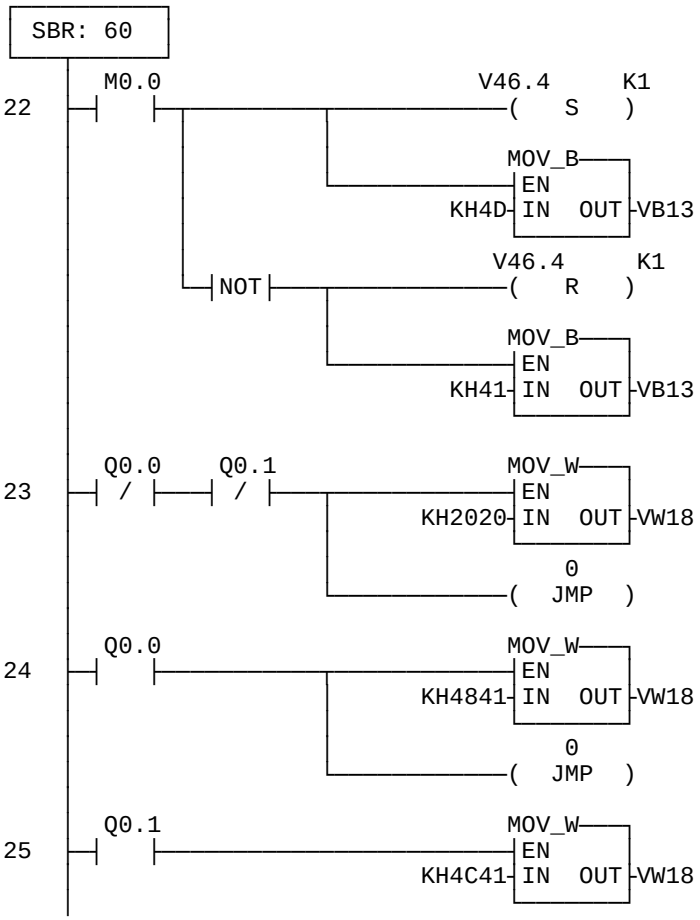

```
SBR     58               // Подпрограмма проверки диапазона / правильности

LDR>=   AC0, AC1         // Если новое значение меньше,чем нижний предел
NOT
MOVR    AC1, AC3         // Использовать нижний предел как новое значение
CRET
LDR<=   AC0, AC2         // Если новое значение больше, чем верхний предел
NOT
MOVR    AC2, AC3         // Использовать верхний предел как новое значение
CRET
LD      SM0.0            // В любом другом случае,
MOVR    AC0, AC3         // Использовать новое значение
RET
```

// Подпрограмма 59: Переключение Авто / Ручной режим

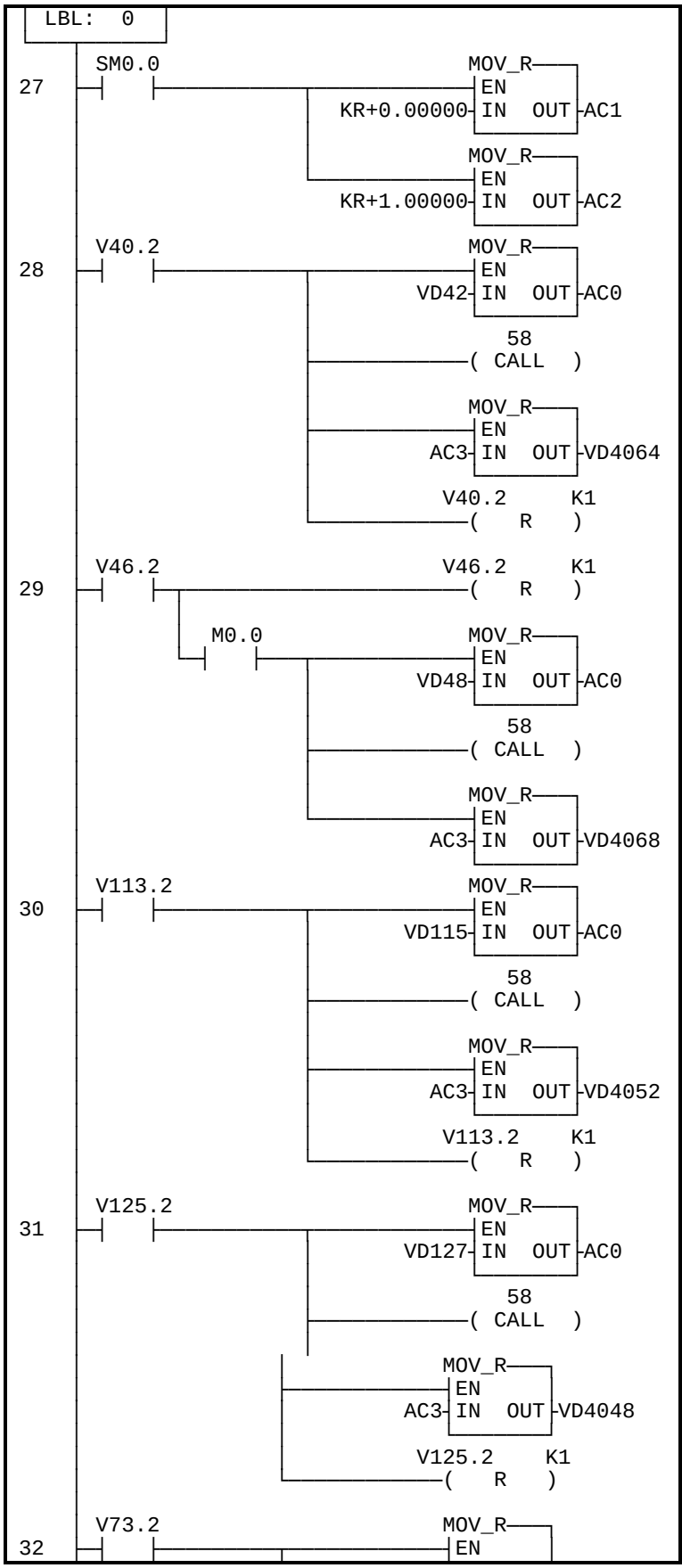



```
SBR     59

LDN     M0.0    // Управляющий бит
=       M0.0    // переключения режима

RET
```

// Подпрограмма 60: Подпрограмма передачи данных TD200 (опционно)




```
SBR     60      // Подпрограмма TD200

LD M0.0//В ручном режиме,
S V46.4, 1      // Позволить
        //редактировать цикл вывода
MOVB    16#4D, VB13
        // Индикация ручного режима

NOT             // иначе,
R V46.4, 1      // Зпрет редактирования
                // цикла вывода
MOVB 16#41, VB13
        // Индикация авто режима

LDN Q0.0        // Если нет верхней
                // тревоги
AN Q0.1// и нет нижней тревоги,
MOVW 16#2020, VW18   // Изменение
        //сообщения
        // 0 не показывать  условия
        //тревоги
JMP 0   // Пропустить проверки верхней
        //и нижней тревоги

LD Q0.0         // если верхняя тревога,
MOVW 16#4841, VW18
                // индикация условияНА

JMP 0           // пропустить проверку //
нижней тревоги

LD Q0.1         // если нижняя тревога,
MOVW 16#4C41, VW18
                // индикация условий LA
```



LBL 0

LD SM0.0 // Каждый проход
MOVR 0.0, AC1 // Установка нижнего //предела

MOVR 1.0, AC2 // установка верхнего //предела

LD V40.2 // если уставка //изменялась,
MOVR VD42, AC0 // получить новое //значение в °C

CALL 58 // Вызов подпрограммы //диапазона

MOVR AC3, VD4064 // изменение //таблицы цикла со значением //проверенного диапазона

R V40.2, 1 // очистка бита сообщения //редактирования

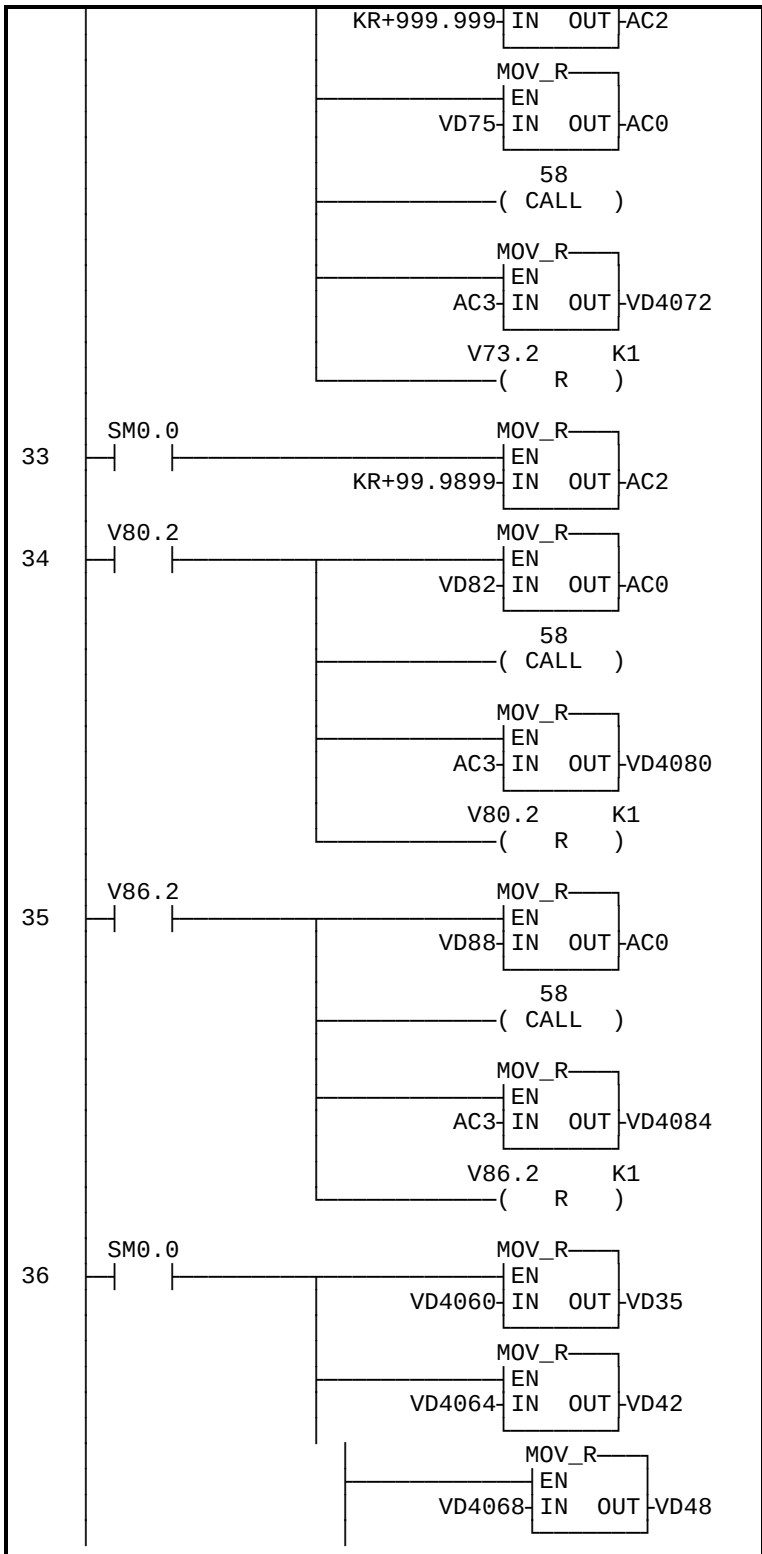
LD V46.2 // если вывод изменялся,
R V46.2, 1 // очистка бита сообщения //редактирования

A M0.0 // и если цикл в ручном //режиме
MOVR VD48, AC0 // получение //нового значения

CALL 58 // вызов подпрограммы //диапазона
MOVR AC3, VD4068 // изменение таблицы цикла со //значением проверенного //диапазона
LD V113.2 // если высокая тревога //была изменена,
MOVR VD115, AC0 // получить новое //значение

CALL 58 // вызов проверки //диапазона

MOVR AC3, VD4052 // изменить //предел тревоги со значением //проверенного диапазона
R V113.2, 1 // очистка бита сообщения



//редактирования

LD V125.2 // если низкая тревога
//была изменена,
MOVR VD127, AC0 // получить новое
//значение

CALL 58 // вызов проверки
//диапазона
MOVR AC3, VD4048 // изменить
//предел тревоги со значением
//проверенного диапазона
R V125.2, 1 // очистка бита сообщения
//редактирования

LD V73.2 // если усиление было
//изменено,
MOVR 999.9900, AC2 // установка
//верхнего предела

MOVR VD75, AC0 // получение
//нового значения

CALL 58 // вызов проверки
//диапазона

MOVR AC3, VD4072 // изменение
//таблицы цикла со значением
//проверенного диапазона

R V73.2, 1 // очистка бита сообщения
//редактирования

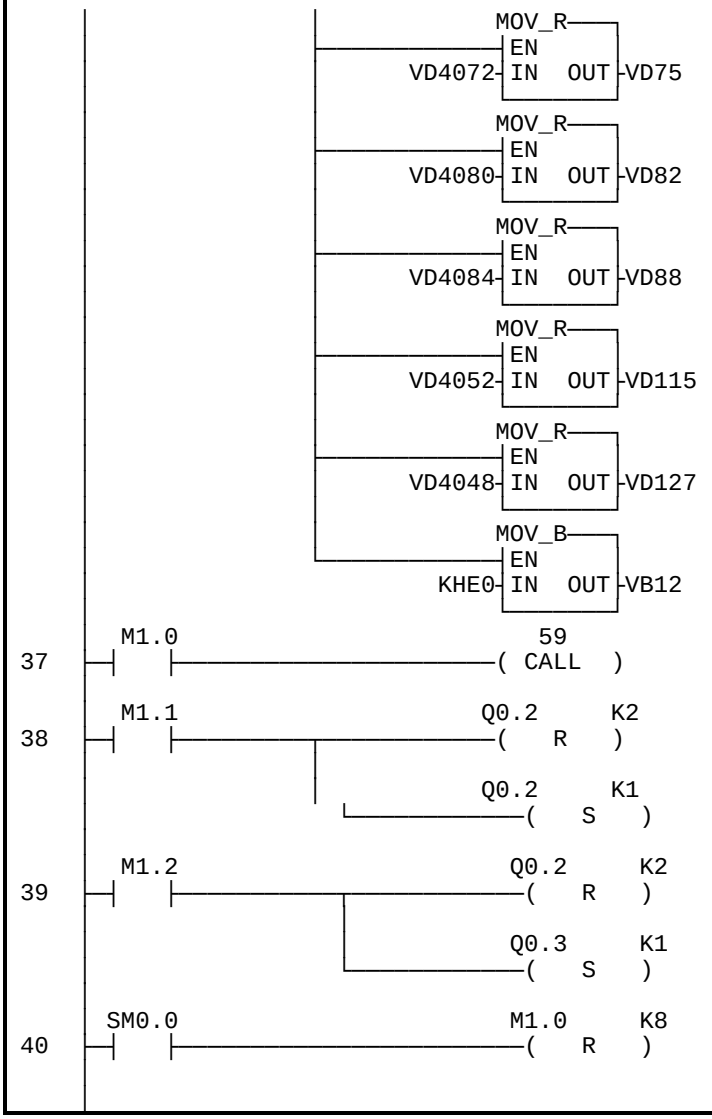
LD SM0.0 // каждый проход
MOVR 99.99, AC2 // установка
//верхнего предела
LD V80.2 // если суммарное время
//(Ti) было изменено,
MOVR VD82, AC0 // получение
//нового значения

CALL 58 // вызов проверки
//диапазона

MOVR AC3, VD4080 // изменение
//таблицы цикла со значением
//проверенного диапазона
R V80.2, 1 // очистка бита сообщения
//редактирования
LD V86.2 // если производное
//время (Td) было изменено,
MOVR VD88, AC0 // Получение
//нового значения

CALL 58 // вызов проверки
//диапазона

```
MOVR AC3, VD4084      // изменение
          //таблицы цикла со значением
          //проверенного диапазона

R V86.2, 1     // очистка бита сообщения
          //редактирования

LD SM0.0          // изменение TD200 со
          //значением измененного
          //диапазона
MOVR VD4060, VD35      // PV

MOVR VD4064, VD42      // SP

MOVR VD4068, VD48      // выход
```




```
MOVR VD4072, VD75      // усиление

MOVR VD4080, VD82      // суммарное
          //время

MOVR VD4084, VD88      // производное
          //время

MOVR VD4052, VD115     // верхний предел
          //тревоги

MOVR VD4048, VD127     // нижний предел
          //тревоги

MOVB 16#E0, VB12
          // разрешение всем трем TD200
          //сообщениям

LD M1.0    // если нажата клавиша          //F1 ,
CALL 59    // переключение авто/ручной
          //режим

LD M1.1    // если нажата клавиша F2 ,
R Q0.2, 2  // очистка всех входных PV
          //опций и
S Q0.2, 1  // выбор входной PV опции 1

LD M1.2    // если нажата клавиша          //F3 ,
R Q0.2, 2  // очистка всех входных PV
          //опций и
S Q0.3, 1  // выбор входной PV опции 2
```

```
41 ├──────────────────────────────( RET )
```

```
LD SM0.0        // каждый проход
R M1.0, 8       // сброс всех битов
                //функциональных клавиш

RET
```

// Подпрограмма 61: Подпрограмма цифрового выхода (дополнительная)

```
SBR: 61
43 ─┤ / ├──────────────────────────────────────────┤TON──T33
                                                    IN
                                              K100─PT

44 VW4032 ──┤ == W ├──┤NOT├──VW4032 ──┤ >= W ├──────────( Q0.4 )

45 ─┤ T33 ├──────────────────────────────────────( M0.1 )
                                                  MOV_W
                                                  EN
                                         VW4030─IN  OUT─VW4032

46 ──────────────────────────────────────────────( RET )
```

```
SBR 61                          // подпрограмма цифрового выхода
LDN M0.1                        // старт таймера на 1 секунду
TON T33, +100
LDW=   VW4032, T33              // если цифровой выход боьше чем
NOT
AW>=   VW4032, T33              // текущий счетчик,
= Q0.4                          // Включите цифровой выход
LD T33                          // когда таймер достигнет 1 секунду
= M0.1                          //  сброс таймера
MOVW VW4030, VW4032             //  и сохранение текущего значения цифрового выхода
RET
```

// Подпрограмма 62: Подпрограмма моделирования процесса (дополнительная-необходима только для демонстрации)

```
SBR: 62
48 ─┤ SM0.0 ├──┬──────────────────────────MUL_R
                │                          EN
                │                 VD4056─IN1 OUT─AC0
                │              KR+9.00000─IN2
                │
                │                          ADD_R
                │──────────────────────────EN
                │                 VD4068─IN1 OUT─AC0
                │                    AC0─IN2
                │
                │                          DIV_R
                │──────────────────────────EN
                │              KR+10.0000─IN1 OUT─AC0
                │                    AC0─IN2
```

```
SBR 62     // подпрограмма
           //моделирования процесса

LD SM0.0        // Каждый проход
MOVR VD4056, AC0
           // получение переменной
           //задержки
*R 9.0, AC0
           // умножение переменной
           //задержки на 9

+R VD4068, AC0
           //прибавление нового значения
           //выхода
```

```
                                  MOV_R
                                  EN
                          AC0-IN    OUT-VD4056
                                  MUL_R
                                  EN
                   KR+32000.0-IN1   OUT-AC0
                          AC0-IN2
                                  DIV_R
                                  EN
                   KR+1.20000-IN1   OUT-AC0
                          AC0-IN2
                                  ADD_R
                                  EN
                   KR+6400.00-IN1   OUT-AC0
                          AC0-IN2
      AC0      KR+32000.0         MOV_R
49 ---------| >= R |-------------EN
                          KR+32000.0-IN    OUT-AC0
```

/R 10.0, AC0
// деление на 10

MOVR AC0, VD4056
// сохранение нового значения
//переменной задержки

***R 32000.0, AC0 // масштабирование**
//значения
//в аккумуляторе

/R 1.20, AC0 // масштабирование 20%
//для диапазона от
// 4 до 20 ma

+R 6400.000, AC0
// смещение для диапазона от 4
//до 20 ma

```
50 --| SM0.0 |-------------------TRUNC
                                EN
                        AC0-IN    OUT-AC0
                                MOV_W
                                EN
                        AC0-IN    OUT-VW4024

51 ------------------------------( RET )
```

LDR>= AC0, 32000.00
// если значение в AC0 //превышает 32000
MOVR 32000.00, AC0
// ограничить значение до 32000

LD SM0.0 // каждый проход
TRUNC AC0, AC0
// преобразование
//вещественного значения
// в целое 32 битовое

MOVW AC0, VW4024 // сохранение
//этого значения в качестве
//новой PV

RET

// Подпрограмма 63: Подпрограмма инициализации

```
SBR: 63
53 --| SM0.0 |-------------------FILL_N
                                EN
                          K0-IN    OUT-VW4024
                         K36-N
                                MOV_R
                                EN
                   KR+0.20000-IN   OUT-VD4064
```

SBR 63

LD SM0.0
FILL 0, VW4024, 36 // очистка
//рабочей области памяти
// и таблицы цикла

MOVR 0.20, VD4064
// инициализация уставки цикла

```
                    MOV_R
                  EN
KR+1.00000 IN   OUT VD4072

                    MOV_R
                  EN
KR+0.10000 IN   OUT VD4076

                    MOV_R
                  EN
KR+1.50000 IN   OUT VD4080

                    MOV_R
                  EN
KR+0.00000 IN   OUT VD4084

                    MOV_R
                  EN
KR+275.000 IN   OUT VD4044

                    MOV_R
                  EN
KR+0.10000 IN   OUT VD4048

                    MOV_R
                  EN
KR+0.90000 IN   OUT VD4052

                    MOV_W
                  EN
     K6400 IN   OUT VW4024

                    MOV_B
                  EN
      K100 IN   OUT SMB34

                    ATCH
                  EN
     K127 INT
      K10 EVT

                  (  ENI  )

    SM0.0                Q0.2
54 |  |              (      )

                   Q0.3     K1
                  (   R   )

                  (  RET  )
55
```

MOVR 1.0, VD4072
// инициализация усиления
//цикла

MOVR 0.10, VD4076
// инициализация типового
//времени

MOVR 1.50, VD4080
// инициализация суммарного
//времени

MOVR 0.0, VD4084 // инициализация
//производного времени – не
//производного действия

MOVR 275.0, VD4044 // инициализация
//максимальной температуры в °C

MOVR 0.10, VD4048 // инициализация
//10% нижнего предела сигнала

MOVR 0.90, VD4052 // инициализация
//90% верхнего предела сигнала

MOVW +6400, VW4024 // инициализация
//переменной процесса от 4 ma для
// подпрограммы моделирования

MOVB 100, SMB34 // установка 100 ms
// для прерывания по времени

ATCH 127, 10 // установка прерывания по
//времени для вызова
//выполнения PID

ENI // возможность прерываний

LD SM0.0 // каждый проход
= Q0.2 // выбор входа из опции 1:
// внутреннее моделирование

R Q0.3, 1 // очистка остальных опций
//ввода

RET

// Подпрограмма прерывания 157: PID - подпрограмма (прерывание по воемени для вызова PID)

```
INT:127
```

INT 127

```
57  Q0.2
 ─┤   ├──┬──────────────────────────────────────────────────────────
          │                                  WXOR_DW
          │                                 ┌EN     ┐
          │                            AC0─┤IN1 OUT├─AC0
          │                            AC0─┤IN2    ┘
          │                                  MOV_W
          │──────────────────────────────────┌EN     ┐
          │                         VW4024─┤IN  OUT├─AC0
          │                                  DI_REAL
          │──────────────────────────────────┌EN     ┐
          │                            AC0─┤IN  OUT├─VD4060
          │                                  SUB_R
          │──────────────────────────────────┌EN     ┐
          │                   KR+6400.00─┤IN1 OUT├─VD4060
          │                       VD4060─┤IN2    ┘
          │                                  MUL_R
          │──────────────────────────────────┌EN     ┐
          │                   KR+1.20000─┤IN1 OUT├─VD4060
          │                       VD4060─┤IN2    ┘
          │                                  DIV_R
          │──────────────────────────────────┌EN     ┐
          │                   KR+32000.0─┤IN1 OUT├─VD4060
          │                       VD4060─┤IN2    ┘

51  Q0.3
 ─┤   ├──┬──────────────────────────────────────────────────────────
          │                                  WXOR_DW
          │                                 ┌EN     ┐
          │                            AC0─┤IN1 OUT├─AC0
          │                            AC0─┤IN2    ┘
          │                                  MOV_W
          │──────────────────────────────────┌EN     ┐
          │                          VW210─┤IN  OUT├─AC0
          │                                  DI_REAL
          │──────────────────────────────────┌EN     ┐
          │                            AC0─┤IN  OUT├─VD4060
          │                                  SUB_R
          │──────────────────────────────────┌EN     ┐
          │                   KR+6400.00─┤IN1 OUT├─VD4060
          │                       VD4060─┤IN2    ┘
          │                                  MUL_R
          │──────────────────────────────────┌EN     ┐
          │                   KR+1.20000─┤IN1 OUT├─VD4060
          │                       VD4060─┤IN2    ┘
          │                                  DIV_R
          │──────────────────────────────────┌EN     ┐
          │                   KR+32000.0─┤IN1 OUT├─VD4060
          │                       VD4060─┤IN2    ┘

52  VD4060      VD4048          Q0.1
 ──────┤ <= R ├──────────────(        )
```

// PID масштабирование входов

// OPTION 1:
// ввод PV из внутреннего моделирования
LD Q0.2 // Если выбрано затем
//получают значение PV из
//моделирования
XORD AC0, AC0 // очистка AC0

MOVW VW4024, AC0
// получение необработанного
//значения PV из подпрограммы
//моделирования

DTR AC0, VD4060
// преобразование в вещественный
//номер

-R 6400.0, VD4060
// коррекция смещения от 4 до 20
//ma

***R 1.20, VD4060 // масштабирование для**
//20 смещения

/R 32000.0, VD4060 // нормализация
//значения PV в таблице цикла
// опция 2:
// ввод PV из входного аналогового
//модуля
LD Q0.3 // если выбрана то
//получение значения PV из AIW
XORD AC0, AC0 // очистка AC0

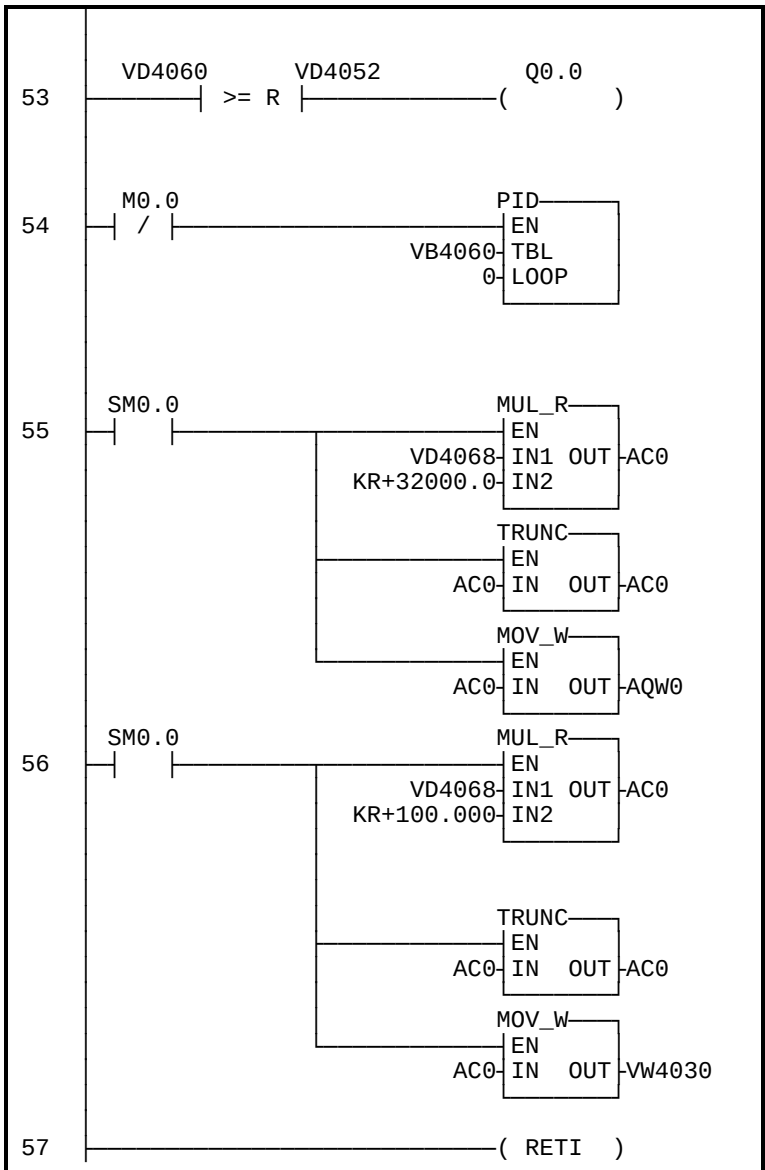
MOVW AIW0, AC0
// получение среднего значения PV

DTR AC0, VD4060
// преобразование в вещественный
//номер

-R 6400.0, VD4060
// корректировка смещения от 4 до
//20 ma

***R 1.20, VD4060 // масштабирование для**
//20% смещения

/R 32000.0, VD4060 // нормализация
//значения PV в таблице цикла




```
          // PID проверка сигнала
LDR<= VD4060, VD4048
          // если достигнут нижний предел
          //сигнала,
= Q0.1    // включить индикатор нижнего
          //сигнала
LDR>= VD4060, VD4052
          // если достигнут верхний предел
          //сигнала,
= Q0.0    // включить индикатор верхнего
          //сигнала

          // выполнение PID –команд в
          //автоматическом режиме

LDN M0.0        // в авто режиме
PID VB4060, 0
          // выполнение PID-команд

          // аналоговый выход PID

LD SM0.0        // обеспечивается
          //аналоговый выход
MOVR VD4068, AC0      // перенос выхода
          //цикла в аккумулятор
*R 32000.0, AC0       // масштабирование
          //значения в аккумуляторе

TRUNC AC0, AC0        // преобразование
          //вещественного номера в целое 32
          //битное

MOVW AC0, AQW0        // запись целого 16
          //битного значения в аналоговый
          //выход

// цифровой выход PID

LD SM0.0        // обеспечение цифрового
          //выхода
          // (управление режимом работы)
MOVR VD4068, AC0      // перенос выхода
          //цикла в аккумулятор
*R 100.0, AC0    // масштабирование
          //значения от 0 до 100



TRUNC AC0, AC0        // преобразование
          //вещественного числа в целое 32
          //битное

MOVW AC0, VW4030
          // сохранение значения цифрового
          //выхода


RETI
```

Блок данных DB1 (V Память):

```
// начало TD200_BLOCK 0
// (Комментарии внутри этого блока не должны быть редактированы или удалены)
VB0     'TD'            // TD 200 идентификатор
VB2     16#11           // Установка языка- Английский, установка модификации - каждая секунда
VB3     16#01           // Установка дисплея в 40 символьный режим; клавиша вверх v3. 2; клавиша вниз v3.
3VB4    3               // Установка числа сообщений
VB5     1               // Установка бит сообщения Функциональных клавиш в M1. 0 - M1. 7
VW6     13              // Установка стартового адреса для сообщений в VW13
VW8     12              // Установка стартового адреса для разрешающего бита сообщения в VW12
// СООБЩЕНИЕ 1  Бит разрешения сообщения V12.7
VB13    'M PV LA  SP    OUT  '
VB33    16#00           // Нет редактирования; Нет подтверждения; Нет пароля;
VB34    16#52           // Реальное Двойное Слово; 2 цифры справа от десятичной точки;
VD35    16#0000                 // Вложенное значение данных: переместить данные для отображения сюда.
VB39    ' '
VB40    16#00           // Нет редактирования; Нет подтверждения; Нет пароля;
VB41    16#52           // Реальное Двойное Слово; 2 цифры справа от десятичной точки;
VD42    16#0000                 // Вложенное значение данных: переместить данные для отображения сюда.
VB46    16#00           // Нет редактирования; Нет подтверждения; Нет пароля;
VB47    16#52           // Реальное Двойное Слово; 2 цифры справа от десятичной точки;
// СООБЩЕНИЕ 2 Бит разрешения сообщения V12.6
VB53    '  Kc    Ti    Td   '
VB73    16#00           // Нет редактирования; Нет подтверждения; Нет пароля;
VB74    16#52           // Реальное Двойное Слово; 2 цифры справа от десятичной точки;
VD75    16#0000                 // Вложенное значение данных: переместить данные для отображения сюда.
VB79    ' '
VB80    16#00           // Нет редактирования; Нет подтверждения; Нет пароля;

VB81    16#52           // Real Double Word;  2 Digits to the right of the decimal;
VD82    16#0000                 // Вложенное значение данных: переместить данные для отображения сюда.VB86
        16#00                   // Нет редактирования; Нет подтверждения; Нет пароля;
VB87    16#52           // Реальное Двойное Слово; 2 цифры справа от десятичной точки;
VD88    16#0000                 // Вложенное значение данных: переместить данные для отображения сюда.
VB92    ' '
// СООБЩЕНИЕ 3 Бит разрешения сообщения V12.5
VB93    'Hi Alarm    Lo Alarm'
VB113   16#00           // Нет редактирования; Нет подтверждения; Нет пароля;
VB114   16#52           // Реальное Двойное Слово; 2 цифры справа от десятичной точки;
VD115   16#0000                 // Вложенное значение данных: переместить данные для отображения сюда.
VB119   '     '
VB125   16#00           // Нет редактирования; Нет подтверждения; Нет пароля;
VB126   16#52           // Реальное Двойное Слово; 2 цифры справа от десятичной точки;
VD127   16#0000                 // Вложенное значение данных: переместить данные для отображения сюда.
VB131   ' '
// КОНЕЦ TD200_BLOCK 0
```

Примечания преобразования

Для перевода из IEC STL в S7-Micro/DOS STL:

- Добавить ‘K’ перед всеми не шестнадцатиричными константами (например 4 ⇒ K4)
- Заменить ‘16#’ на ‘KH’ для всех шестнадцатиричных констант (например16#FF ⇒ KHFF)
- Запятые обозначают разделение полей. Используйтее стрелку или клавишу TAB для переходов между полями.
- Чтобы преобразовывать S7-Micro/DOS программу STL в форму LAD, каждая сеть должна начинаться со слова 'СЕТЬ' и номера. Каждая сеть в Application Tip программы обозначена номером на

диаграмме . Используйте команду INSNW под меню РЕДАКТИРОВАНИЯ, чтобы ввести новую сеть. MEND, RET, RETI, LBL, SBR, и INT команды получают каждая свою собственную сеть.

- Линия - комментариев, обозначенная '//' не возмона в S7-Micro/DOS, но комментарии к стеи возможны

Показанный блок данных, был создан в Micro/Win; в Micro/DOS использован редактор V-памяти для ввода блока данных (Обратитесь к руководству пользователя Micro/DOS). Обратите внимание что в Micro/Win:

- 16 *.. обозначает шестнадцатеричное значение
- 'текст' представляет строковое значение

Общие примечания

SIMATIC S7-200 Примеры применения предназначены для того, чтобы дать пользователям S7-200 некоторое представление относительно того, как, некоторые задачи программирования могут быть решены этим контроллером. Эти инструкции не учитывают все детали или разнообразие в оборудовании, и при этом они не предусматривают какое - либо непредвиденное обстоятельство. Использование S7-200 Примеров применения - свободное.
Siemens оставляет за собой право делать изменения в спецификациях, показанных здесь или делать усовершенствования в любое время без оповещения или обязательств. Это не освобождает пользователя от ответственности, за использование методов на прктике, установке, обработке, и сопровождении приобретенного оборудования. Если конфликт возникает между общей информацией, содержащейся в этой публикации, содержанием рисунков или дополнительным материалом, или обоих, более поздние должны иметь приоритет.
Siemens не ответственен, по любой допустимой причине, в повреждениях или персональном ущербе, последующим из использования примеров применения.
Все права зарезервированы. Любая форма дублирования или распространения, включая выдержки, разрешается только с конкретным разрешением SIEMENS

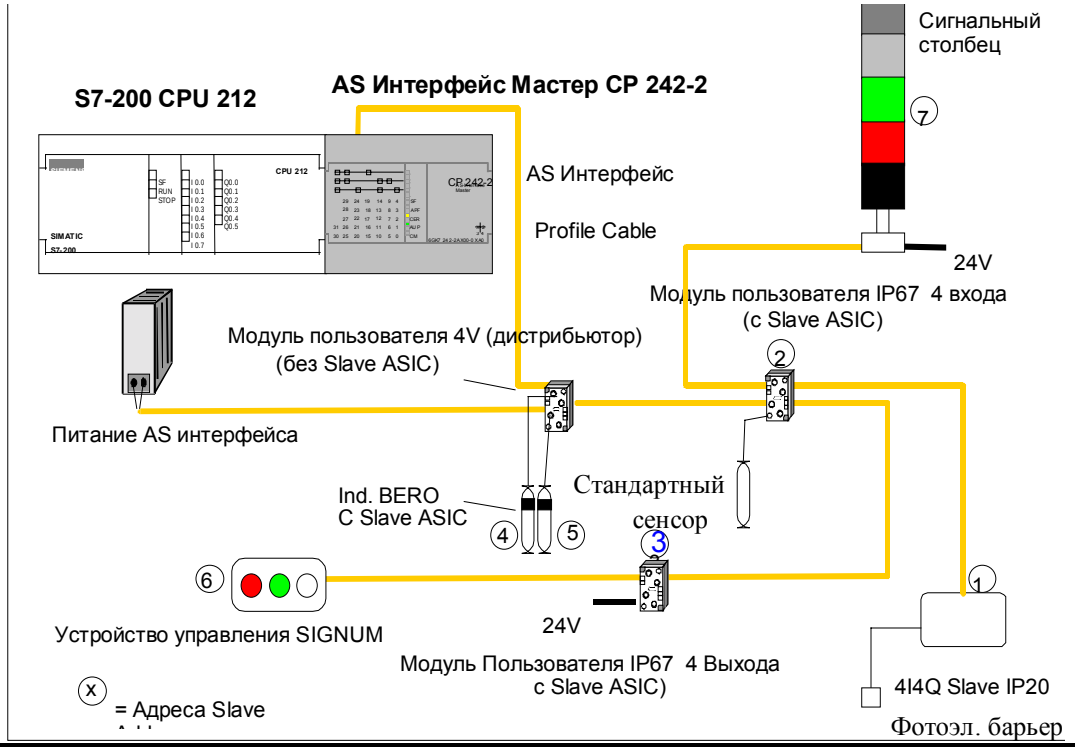
SIMATIC S7-200 Примеры

Группа	Тема
3	**Интеграция данных Входов - выходов AS интерфейса в S7-200**

Требуемые для данного примера CPU					
CPU 210 ☐	CPU 212 ☑	CPU 214 ☑	CPU 215 ☑	CPU 216 ☑	ДРУГИЕ ☐

Обзор

- **Функции типовой системы**
- · красная клавиша переключает красный индикатор сигнального столбца вкл\выкл
- · зеленая клавиша переключает зеленый индикатор столбца сигналов вкл\выкл
- · Активный Датчик Интерфейса AS 1 преключает прожектор столбца сигнала вкл\выкл
- · Активный Датчик Интерфейса AS 2 переключает сенсорный элемент столбца сигнала
- · Стандартный датчик на theIP65 модуле пользователя переключает индикатор в устройстве управления вкл\выкл
- · Фотоэлектрический барьер на IP 20 4I4Q slave переключателе всех четырех выводов IP65 4A slave



Компоненты оборудования

Список компонентов AS интерфейса, используемые в Типовой Системе

Внутри области обозначенной типовой системы, используются следующие компоненты AS интерфейса:
Для минимальной конфигурации, необходимы по крайней мере компоненты перечисленные от Позиции 1 до Позиции 4. Эти компоненты могут также быть соединены одним круговым кабелем.
Модули интерфейса также как устройство управления SIGNUM связаны через AS интерфейсный профильный кабель.
Устройство адресации используется для установки slave адресов. Slave адреса, перечисленные ниже используются для типовой системы. В состоянии ' как установлено ', Адрес 0 установлен для slaves. Для работы, он должен быть установлен в значение между 1 и 31.

Поз.	Порядковый номер	Название	Адреса Slave
1.	6ES7212-1BA01-0XB0	SIMATIC S7-200 CPU 212 с выходными реле	
2.	6GK7242-2AX00-0XA0	CP 242-2 AS Интерфейс Мастер для S7-200	
3.	3RX9300-0AA00	AS-Интерфейс Питание с разделением данных, IP 20	
4.	3RG9002-0DB00	AS-Интерфейс Slave Модуль 4I/4Q с Screw Terminals	1
5.	3RG9001-0AA00	AS-Интерфейс Slave Модуль пользователя IP67 4 Входа	2
6.	3RG9010-0AA00	Модуль интерфейса FK (для Поз. 5)	
7.	3RG9001-0CB00	AS-Интерфейс Slave Модуль пользователя IP67 4 Выхода	3
8.	3RG9030-0AA00	Модуль интерфейса FK-E (для Поз. 7)	
9.	3RG9001-0AL00	Модуль пользоваетля 4V	
10.	3RG9010-0AA00	Модуль интерфейса FK (для Поз. 9)	
11.	3RG4613-3WS00	Индуктивный BERO 3RG4, Тип M18 с соединением M12	4 и 5
12.	3RX1651	Соединительный кабель M12 (для Поз. 11)	
13.	3SB3803-4DA	Устройство управления SIGNUM 3SB3 (2клавиши;1индикатор)	6
14.	8WD4308-0DA	Столбец сигналов: Foot with Tube	
15.	8WD4308-0AA	Столбец сигналов: элемент подключения	
16.	8WD4328-0BB	Столбец сигналов: Элемент адаптера AS интерфейса со внешним дополнительным напряжением	7
17.	8WD4320-0FA	Столбец сигналов: гудок 24V	
18.	8WD4320-0CE	Столбец сигналов: Чистый элемент прожектора 24V	
19.	8WD4320-1BB	Столбец сигналов: Мигающий красный световой элемент 24V	
20.	8WD4300-1AC	Столбец сигналов: Устойчивый зеленый световой элемент 24V	
21.	3RX9010-0AA00	Профильный кабель AS интерфейса, желтый, каучуковый (100м рулон)	
22.	3RX9400-0AA00	Адресное устройство AS интерфейса с 230V загрузочным адаптером (для установки адресов slave)	

Структура программы

Функции, упомянутые выше, выполняются программой ниже. Для функционирования программы принято следующее:

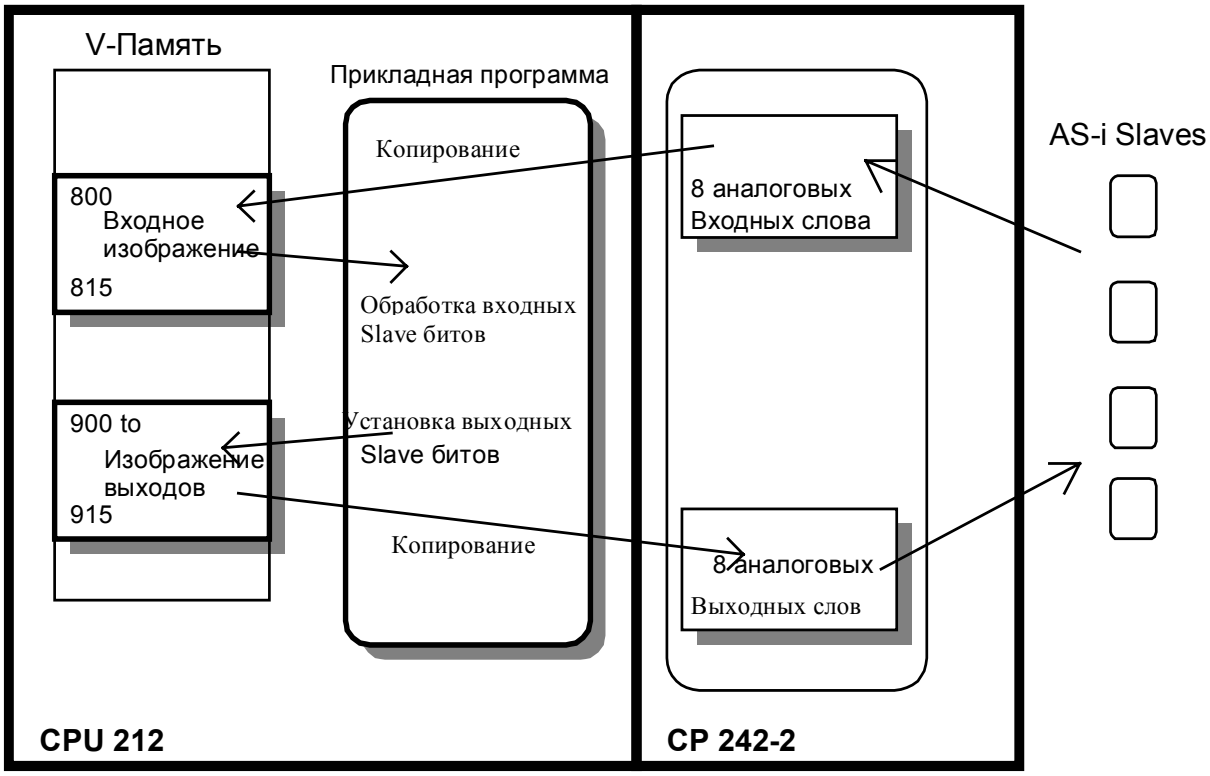
· AS интерфейс мастер CP 242-2 - единственный модуль, связанный с CPU 212.
· входные данные, вспомогательных AS интерфейсов копируются из аналоговых входных слов AIW0 в AIW14 в V-памяти от 800 до 815. Начиная с 800, изображение входных данных, AS интерфейсов slave затем размещаются в V-памяти.
· выходные данные для AS интерфейса мастера регистрируется в V-памяти от 900 до 915. Начиная с 900, изображение выходных данных AS интерфейса slave затем размещаются в V-памяти.
· после того, как все slave биты вывода будут установлены в V-памяти, начинающейся с 900, 16 байтов, начинающиеся с V-памяти 900 копируются в аналоговые слова AQW0 в AQW14 CP 242-2.

Описание программы

Доступ к битам Ввода - вывода AS интерфейса

Для функционирования программы, принято следующее:

- AS интерфейс мастер CP 242-2 - единственный модуль, связанный с CPU 212.
- Входные данные AS интерфейсов slave копируются в V-память от 800 до 815 из аналоговых входных слов AIW0 в AIW14. Изображение входных данных AS интерфейсов slave - затем в V-память, начинающейся с 800.
- Выходные данные для AS интерфейса мастера регистрируются в V-памяти от 900 до 915. Изображение выходных данных AS интерфейсов slave - затем в V-память, начинающейся с 900.
- После того, как все slave биты вывода установлены в V-памяти, начинающейся с 900, 16 байтов копируются в аналоговые значения AQW0 к AQW14 CP 242-2 начиная с V-памяти 900.



Дальнейшие детали обеспечиваются в руководстве CP 242-2.

Список Адресов и назначение разрядов

Адреса для AS интерфейсов slave устанавливаются устройством адресации.

Позиция slaves или slave битов в области аналоговых значений также как в области V-памяти (вводит - или выводит изображения slaves) определена простым способом посредством таблицы назначений, обеспеченной в руководстве CP 242-2.

Для slaves, используемых в представленной типовой системе, следующие результаты назначения:

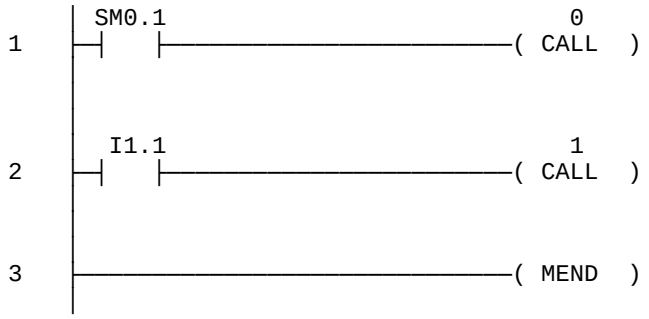
		Входы		Выходы	
Slave адреса	**Slave**	**Позиция в ан. Области**	**Позиция в V-памяти начиная с 800**	**Позиция в области AQ**	**Позиция в V-памяти начиная с 900**
1.	4I/4Q	AIW0	800 бит 0-3	AQW0	900 бит 0-3
2.	4I	AIW0	801 бит 4-7	-	-
3.	4Q	-	-	AQW0	901 бит 0-3
4.	2I	AIW2	802 бит 4-5	-	-
5.	2I	AIW2	802 бит 0-1	-	-
6.	4I/4Q	AIW2	803 бит 4-7	AQW2	903 бит 4-7
7.	4Q	-	-	AQW2	903 бит 0-3

Данные относительно позиции вспомогательных битов AS интерфейсов в V-памяти необходимы для производства программы.

LAD (S7-MicroDOS)	STL (IEC)

Главная программа

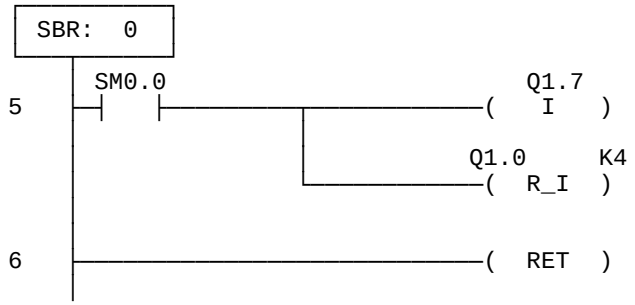
```
//
// Типовая система с CP 242-2
//
// Конфигурация: CPU 212 и CP 242-2 с AS интерфейсом
//
```




```
// Запуск процесса
LD SM0.1        // При первом проходе
CALL 0          // Выполнение UP 0

// Программа управления
LD I1.1         // если CP_READY
CALL 1          // Затем обработка AS-i I/O

MEND
```

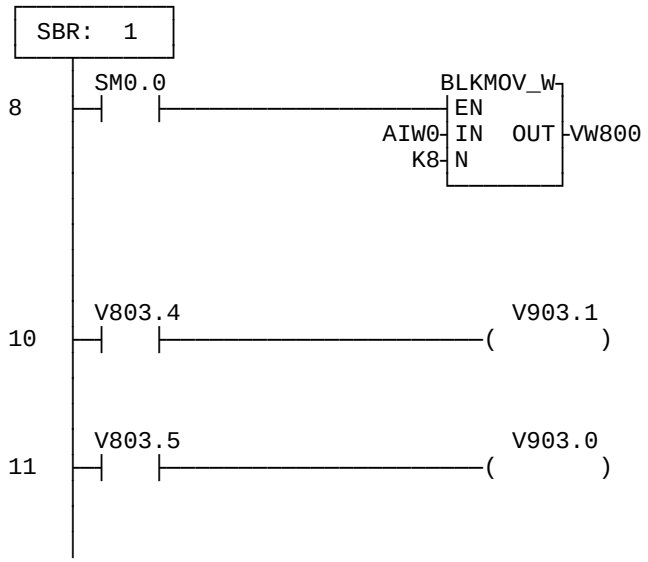
Подпрограммы

```
//
// Подпрограмма 0: "Первый проход"
//
// Важно: Только если  PLC_RUN = 1 возможен обмен данными с вспомогательными ASi!
//
```

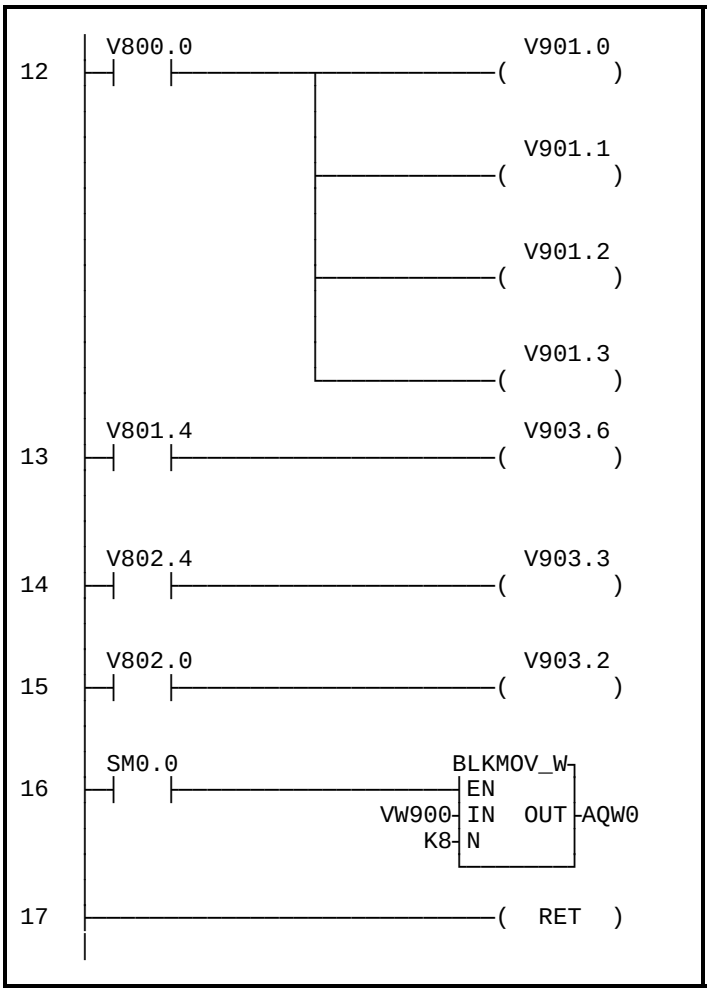



```
SBR 0            // PLC_RUN = 1

LD SM0.0         // В каждом проходе
=I Q1.7          // PLC_RUN = 1

RI Q1.0, 4       // Выбор Банка 0

RET
```

```
//
// Подпрограмма 1 "Обработка AS-i  I/O"
//
// В начале, подпрограмма циклически копирует данные ввода ASi из CP в V-память
// (начиная с Байта 800), и в конце передает ASi выходные данные из V-памяти в CP
// (начиная с Байта 900).
```




```
SBR 1   // "обработка AS-i  I/O"

// Установка" входного изображения"
LD SM0.0         // в каждом проходе
BMW AIW0, VW800, 8
          // выборка входов AS-i в V-память

// Ниже, несколько доступов к битам ASi
// используемых для примера:
LD V803.4        // если нажата красная
                 // кнопка
                 // (на  Slave 6)
= V903.1         // включить красный свет
                 // (на Slave 7)

LD V803.5        // если нажата зеленая
                 // кнопка
                 // (на Slave 6)
= V903.0         // включить зеленый свет
                 // (на Slave 7)
```




```
LD V800.0        // Фотобарьер Slave1
                 // Вход1
= V901.0         // Slave 3 Выход 1

= V901.1         // Slave 3 Выход 2

= V901.2         // Slave 3 Выход 3

= V901.3         // Slave 3 Выход 4

LD V801.4        // Slave 2 Бит 0
                 // Вход сенсора
= V903.6         // Свет в Slave 6  бит 2

LD V802.4        // если сенсор 4  включен,
= V903.3         // включен гудок

LD V802.0        // если сенсор 5 включен,
= V903.2         / включена вспышка

// Прочитано " Изображение выхода "
LD SM0.0         // В каждом проходе
BMW VW900, AQW0, 8
                 //Писать из V-памяти в выходы AS-i

RET
```

Примечания преобразования

При преобразовании из IEC STL в S7-Micro/DOS STL:

- Добавлять ‘K’ перед всеми нешестнадцатиричными числовыми константами (т.е. 4 ⇒ K4)
- Заменять ‘16#’ на ‘KH’ для всех шестнадцатиричных констант (т.е. 16#FF ⇒ KHFF)
- Запятые обозначают разделение полей. Используйте стрелки или клавишу TAB, чтобы переключаться между полями.
- Чтобы преобразовывать S7-Micro/DOS программу STL в форму LAD, каждая сеть должна начинаться со слова 'СЕТЬ' и номера. Каждая сеть в Application Tip программы обозначена номером на ступенчатой диаграмме. Используйте команду INSNW под меню EDIT, чтобы ввести новую сеть Команды.MEND, RET, RETI, LBL, SBR, и INT каждая получает свою собственную сеть.
- Линия - комментариев, обозначенная '//' невозможна в S7-Micro/DOS, но Сеть - комментариев возможна.

Показанные блоки данных были созданы в Micro / Win; в Miicro / DOS использование редактора V-памяти для

ввода блока данных (Обратитесь к руководству пользователя Micro / DOS). Обратите внимание что в Micro / Win:

- 16#.. обозначает шестнадцатиричные значения
- '*text*' представляет строковые значения

Общие замечания

SIMATIC S7-200 Советы по применению обеспечиваются для того, чтобы дать пользователям S7-200 некоторое представление относительно того, как, некоторые задачи из представления методики программирования, могут быть решены с этим контроллером. Эти инструкции не предполагают отражать все детали или разновидности оборудования, и при этом они не предусматривают любое возможное непредвиденное обстоятельство. Использование S7-200 Советов по применению свободное.
Siemens оставляет право делать изменения в спецификациях, показанных здесь или делать усовершенствования в любое время без примечания или обязательства. Это не освобождает пользователя от ответственности, при использовании звуковых методов в прикладной программе, установке, операции, и сопровождении приобретенного оборудования. Если возникает конфликт между общей информацией, содержащейся в этой публикации, содержанием рисунков и дополнительным материалом, или обоими, последние должны иметь приоритет.
Siemens не несет ответственности, по любой допустимой причине, в повреждениях или персональном ущербе, последующем из использования советов по применению.
Все права защищены. Любая форма дублирования или распространения, включая выдержки, разрешается только с конкретным разрешением SIEMENS.

SIMATIC S7-200 Примеры

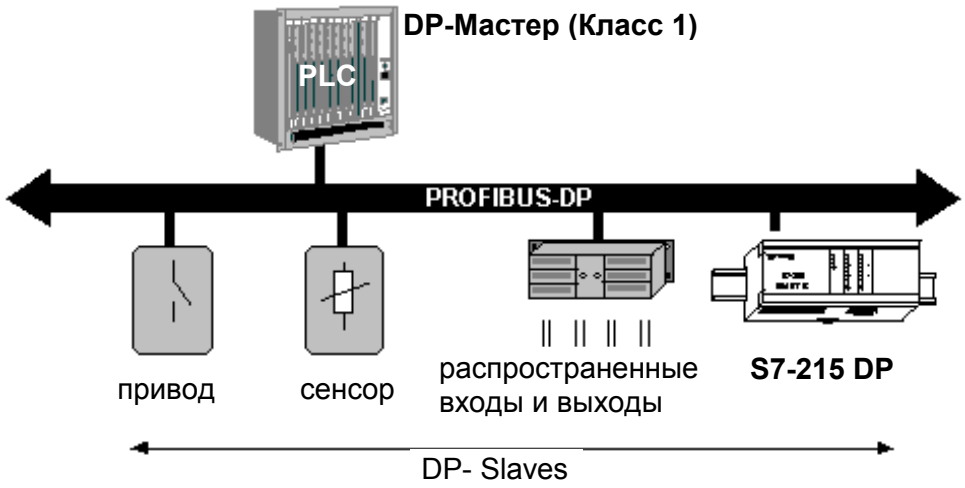
Группа	Тема
3	Общие примечания относительно Profibus и CPU215

Требуемые для этого примера CPU					
CPU 210 ☐	CPU 212 ☐	CPU 214 ☐	CPU 215 ☑	CPU 216 ☐	ДРУГИЕ ☐

Обзор

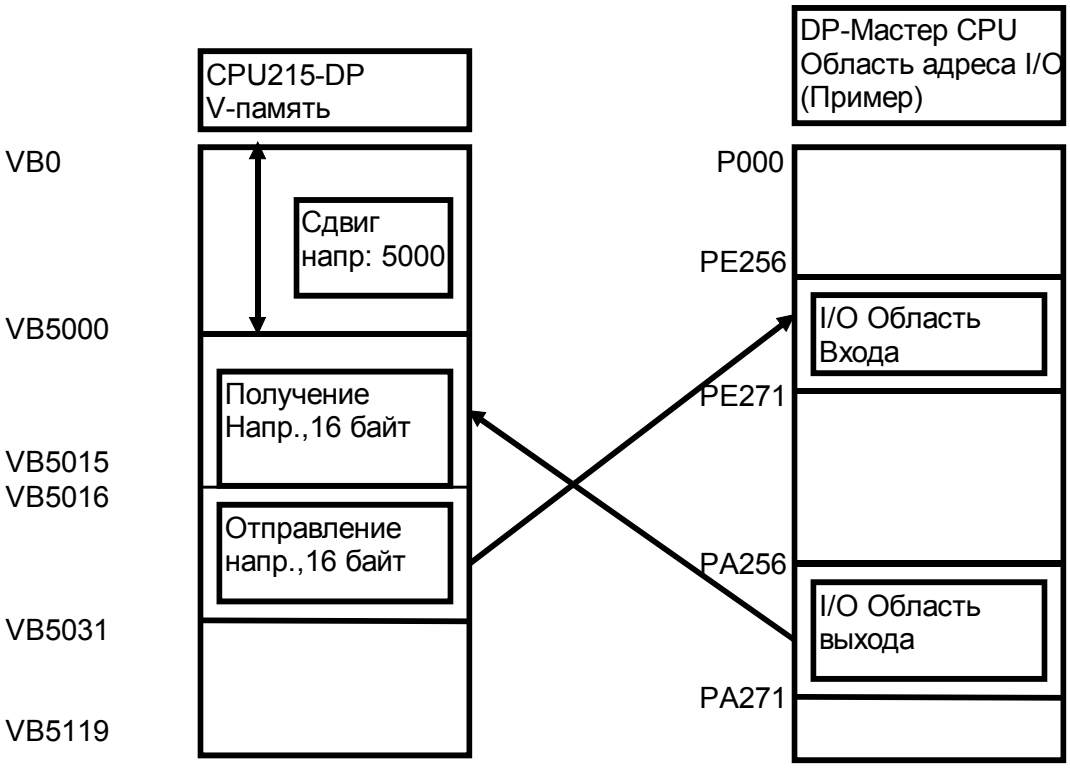
CPU215-DP может быть связан с Profibus DP, и в этом случае используется исключительно как подчиненное устройство. Чтобы гарантировать безупречную связь между DP мастером и CPU215-DP, должны быть сделаны некоторые настройки параметров участвующих устройств.

На CPU215-DP должен быть установлен только адрес DP. Это возможно с программным обеспечением программирования STEP7-MICRO/DOS начиная с Версии 1.3, или с STEP7-MICRO/WIN, начиная с Версии 2.0. Мастер автоматически корректирует скорость в болдах; она не может быть установлена.



Модель памяти DP-мастера и DP-слейва

Цель Profibus подключения - обмен данными. Данные из PLC перемещаются к другому PLC. Инициатор этой передачи данных обозначаются как 'Мастер', а другая станция связи так называемая 'Слейв'. Принцип прост: мастер читает, данные от почтового ящика отправления слейва, и сохраняет это в области ввода Ввода - вывода. Кроме того мастер передает данные в почтовый ящик получения слейва. Так как CPU215 может только быть слейвом, почтовые ящики отправления и получения, находятся в переменной памяти CPU (V-память). Рисунок ниже показывает модель памяти.



Вы можете конфигурировать в мастере где начало почтового ящика получения - устанавливая смещение. Если смещение - 0, это означает, что почтовый ящик получения начинается в переменном байте VB0. Если смещение - 5000, начало почтового ящика получения в VB5000. Кроме того, размер почтового ящика получения может быть определен в байтах, когда конфигурируется мастер. Если размер - 16 байтов, и смещение - 5000, почтовый ящик получения начинается в VB5000 и заканчивается в VB5015. Размер почтового ящика отправления - является непосредственно связанным с почтовым ящиком получения -и может также быть конфигурирован; то есть если например размер - 16 байтов, почтовый ящик отправления простирается от VB5016 до VB5031.

После того, как устанавливается подключение между мастером и слейвом, эта информация может быть прочитана в информации меню 'CPU'.

Установка DP адреса CPU 215

A.) Со STEP7-Micro/DOS V1.3

1.) Начните (F4) и войти в меню 'Utilities' (плоса инструментов => F6). Режим работы должен быть переключен в TERM.

2.) В меню 'Communication Parameters', установить DP адрес CPU215 (AS утилита № 10) .

3.) Сохранить адрес в CPU с помощью клавиши F3 Запись. Затем, выйти из меню 'Communication Parameters' и 'Utilities' с помощью клавиши ESC. Активировать spacer bar и клавишу 'ASSSPEI' (F8) для сохранения изменений в CPU.

B.) Со STEP7-Micro/WIN V2.0

1.) Открыть новый проект и выбрать CPU типа CPU215.

2.) В пункте меню 'CPU', выбрать подменю 'Configure' и щелкнуть на 'DP-Port'. Установить требуемый адрес CPU в области от 0 до 126 в соответствующем поле.
Примечание: Поле 'Baudrate' неприменимо для интерфейса DP , поскольку мастер всегда устанавливает скорость в бодах.

3.) В меню 'Project', подменю 'Load from PG', загружает конфигурацию CPU в CPU. Это установит адрес DP в CPU.

Конфигурирование DP Мастера

Со STEP7:

Со STEP7, S7-CPU (например, CPU315-DP) конфигурируется как DP мастер. Требуются следующие шаги для Версии V2.1 STEP7. Для поздних версий могут быть изменения.

1.) Добавить файл типа 215-DP в случае STEP 7 (опущен начиная с версии 3.0):

a.) Копировать файл типа ‘W92150bx.200’ на диске 2 из директории A:\Step7-V2.1 в директорию C:\Step7_v2\s7data.

b.) Открыть SIMATIC Manager => Open Projekt => правой кнопкой мыши: добавить новый объект, например, SIMATIC 300 Station и затем отметить левой кнопкой мыши => правой кнопкой мыши: Open Object (HWConfig) => Menu‘Extras’ => Update DP Type File => Выход из STEP7 и рестарт.

2.) Конфигурирование аппаратуры S7 как обычно (требуется для пункта1b)

3.) Конфигурирование DP Мастера: Select Master Station и Open Object => Open Subdirectories в Slot Address 2 (CPU) для 2.1 DP-Master => правой кнопкой мыши => Master System -> Open => Set L2 Address => Button 'New' (SINEC L2 network)

4.) Назначение параметров для215 DP Slave: Open Catalog (Button) => Profibus -> Standard Slave -> SIMATIC -> CPU215-DP ->Установка параметров адреса L2 , как для Micro/DOS (требуется для пункта A) и выбор поля I/O в каталоге в поддиректории CPU215.

5.) Установка сдвига I/O в переменной памяти (V-память) CPU215: Open Master System => Mark Slave => Select I/O Field => правая кнопка мыши => Object attributes => Assign Parameters=> Установка сдвига с помощью клавиши 'Change Value'.
(После изменений, включить и выключить CPU215 для передачи) и загрузить проект в мастер CPU.

<u>С COM-PROFIBUS:</u>

Для других устройств мастер (например, S5), программное обеспечение COM Profibus используется для конфигурирования. Для Версии 2.x, файл типа CPU215 еще должен загружаться, как в случае STEP7. Начиная с Версии 3.0, этот файл типа встраивается.

1.) Открыть проект и выбрать DP мастер

2.) В таблице слейв, щелкнуть на пункте 'SIMATIC' => перенести влево и оставить с помощью клавиши мыши=> Select Slave Adress => Select Station Type 'CPU215-2DP' .

3.) Конфигурировать меню: Определить размер поля I/O (например, 16 входных байт/ 16 выходных байт) для CPU215 и назначение области адреса I/O мастера.

4.) Параметры меню: определить сдвиг I/O в переменной памяти CPU215.

5.) Сохранение проекта, экспорт данных и загрузка в мастер CPU.

Приложение: Общие примечания к Profibus DP

Обзор:

0.0 Общие
1.0 Основные функции PROFIBUS-DP
1.1 Основные характеристики
1.2 Конфигурация системы и типы устройств
1.3 Поведение системы
1.4 Циклиечская передача данных между DPM1 и DP слейв
1.5 Циклическая передача данных между DPM1 и устройствами конфигурации
1.6 Режимы Sync и Freeze
1.7 Механизмы защиты
2.0 Расширенные функции DP
2.1 Расширенное соединение данных между DPM1 и DP Slaves
2.1.1 Функции нециклических чтени/записи с DDLM_Read и DDLM_Write
2.1.2 Подтверждение тревоги
2.2 Расширенная передача данных между DPM2 и Slaves
3.0 Файлы базы данных устройств (GSD) позволяют открытую конфигурацию
3.1 Библиотека GSD на www
3.2 Идентификационный номер
3.3 PROFIBUS-DP профили

0.0 Общие
PROFIBUS-DP разработан для быстродействующей связи данных на уровне устройства. Здесь, центральные контроллеры (например, PLC/PC) связываются с их распределенными полевыми устройствами (Ввод - вывод, диски, клапаны, и т.д.) через быстродействующую последовательную связь. Большинство связей данных с этими распределенными устройствами выполнено циклическим способом. Функции, требуемые для этой связи определены базисными функциями PROFIBUS-DP в соответствии с 50 170. В дополнение к выполнению этих циклических функций, нециклические функции связи требуются для интеллектуальных устройств поля, чтобы разрешить конфигурацию, диагностику и сигнальную обработку. Эти новые нециклические функции определены в PROFIBUS руководстве Номер 2.042 и описаны более детально в главе 2.

1.0 Основные функции PROFIBUS-DP
Центральный контроллер (мастер) циклически читает входную информацию от слейвов и циклически пишет выходную информацию слейвам.Время цикла шины должно быть меньше чем время цикла программы центрального PLC, которое для многих приложений является приблизительно 10 мсек. В дополнение к циклической передаче данных пользователя, PROFIBUS-DP обеспечивает мощные функции для диагностики и конфигурирования. Связь данных проверяется, функцией контроля, и на главной и подчиненной сторонах.

1.1 Основные характеристики
Высокая производительность данных один - но не единственный критерии для успешного использования системы шины. Простая установка и сервис, хорошие диагностические возможности и свободная от ошибок доказанная технология передачи также важна пользователю. PROFIBUS-DP представляет оптимальную комбинацию этих характеристик.
Скорость
PROFIBUS-DP требует только приблизительно 1 мсек в 12 Мбит/сек для передачи 512 битов входных данных и 512 битов распределенных выходных данных более чем 32 станций. Рисунок 1 показывает типичное PROFIBUS-DP время передачи в зависимости от числа станций и скорости передачи. Значительное увеличение быстродействия по сравнению с результатами PROFIBUS-FMS главным образом из-за использования SRD обслуживания (обслуживание данных отправления и получения) уровня 2 для передачи входных и выходных данных в одиночном цикле сообщения.
Функции диагностики
Обширные диагностические функции PROFIBUS-DP допускают быстрое обнаружение неисправностей. Диагностические сообщения - передаются над шиной и собираются в мастере. Эти сообщения разделяются на три уровня:

- **· Связанная со станцией диагностика**

Эти сообщения касаются общего операционного состояния целого устройства (то есть, Overtemperature или низкое напряжение).

- **· Связанная с модулем диагностика**

Эти сообщения указывают, что неисправность присутствует в специфическом диапазоне Ввода - вывода (то есть, в 8-разрядном модуле вывода) станции.

- **· Связанная с каналом диагностика**

Эти сообщения указывают на ошибку в индивидуальном бите ввода -вывода (то есть, короткая схема на
Выходе 7).

1.2 Конфигурация системы и типы устройств
PROFIBUS-DP позволяет системы Моно-мастер или Мульти-мастер. Это обеспечивает высокую степень гибкости в течение конфигурирования системы. До 126 устройств (мастеров или слейвов) могут быть связаны с одной шиной. Описание конфигурации системы состоит из числа станций, назначений между адресом станции и адресом Ввода - вывода, формата данных Ввода - вывода, формата диагностических сообщений и используемых параметров шины. Каждая PROFIBUS-DP система может содержать три различных типа устройств:

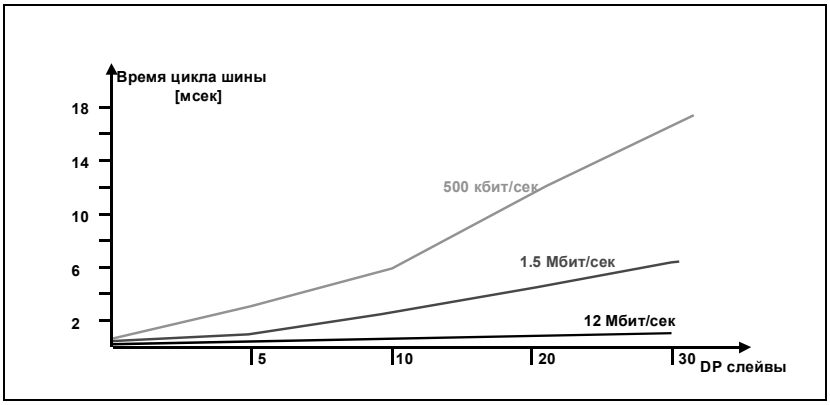
DP Мастер Класс 1 (DPM1)
Класс -1 DP мастер - центральный контроллер, который обменивается информацией с децентрализованными станциями (то есть, DP слейвами) внутри определенного цикла сообщения. Типичные главные устройства включают программируемые контроллеры (PLC) и PC или VME системы.

DP Матсер Класс 2 (DPM2)
Класс 2 DP мастеры - программаторы, устройства конфигурации или панели оператора. Они используются в течение ввода в действие для конфигурирования DP системы или для управления и текущего контроля .

DP слейвы

DP слейв - периферийное устройство (устройства Ввода - вывода, диски, HMI, клапаны, и т.д.) который собирает входную информацию и посылает информацию вывода на контроллер. Имеются также устройства, которые обеспечивают только ввод или только вывод информации.

Количество информации вывода-вывода зависит от типа устройства. Разрешается максимум 246 байтов входной информации и 246 байтов выходной информации.



Условие теста: Каждый слейв имеет 2 байта входных данных и 2 байта выходных данных. Минимальный временной интервал слейва
200 микросекунд. TSDI = 37 bit times и TSDR = 11 bit times.

Рисунок 1: Время цикла шины PROFIBUS-DP одно-мастерной системы

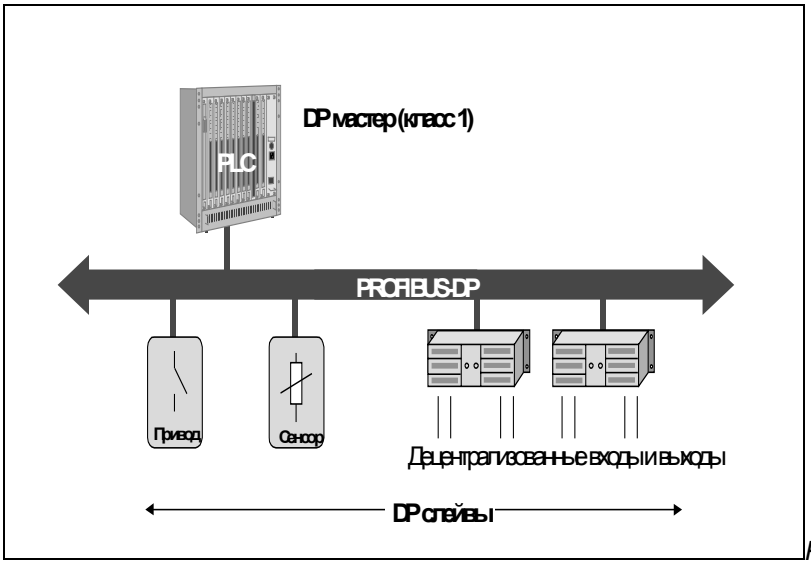


Рисунок 2: PROFIBUS-DP система моно-мастер

В одно-мастерных системах активен только один мастер в сети во время рабочей фазы шинной системы. Рисунок 8 показывает конфигурацию одно-мастерной системы. The programmable controller is the central control component. The distributed DP slaves are linked via the bus to the PLC. Mono-master systems attain the shortest bus cycle time.

В конфигурациях мульти-мастер несколько мастеров связаны с шиной. Эти мастера являются или независимыми подсистемами каждая состоящая из одного мастера DPM1 и назначенных DP слейвов, или дополнительной конфигурации и диагностических устройств (рисунок 3) .Ввод, и изображения вывода DP слейвов могут читаться всеми DP мастерами. Однако, только один мастер DP (то есть, DPM1, назначенный в течение конфигуриривания) имеет доступ для записи выводов. Системы мульти-мастер производят более длинное время цикла шины чем системы моно-мастер.

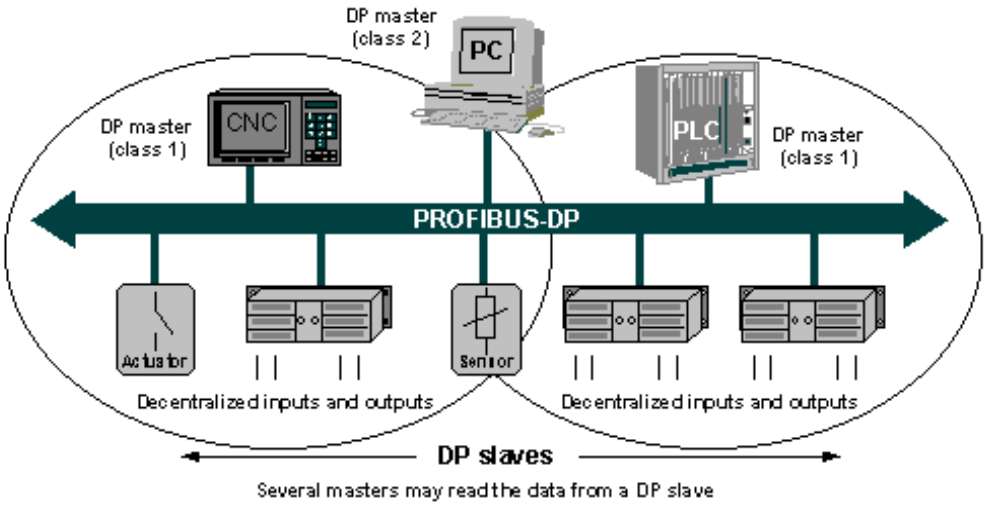


Рисунок 3: PROFIBUS-DP система мульти-мастер

1.3 Поведение системы

PROFIBUS-DP спецификация включает детализированное описание поведения системы, для того, чтобы гарантировать экснепостоянство устройства. Поведение системы определяется прежде всего действующим состоянием DPM1.
DPM1 может управляться или в местном масштабе или через шину устройством конфигурации. Имеются три основных состояния:

- **Stop**
  В этом случае нет никакой передачи данных между DPM1 и DP слэйвами.
- **Clear**
  В этом состоянии, DPM1 читает входную информацию DP слейвов и задерживает выходы в отказоустойчивом состоянии.
- **Operate**
  В этом состоянии, DPM1 находится в фазе передачи данных. В циклической связи данных, вводы DP слейвов читаются, и выводятся, информация записывается в DP слейвы.

DPM1 циклически посылает локальное состояние всем DP слейвам, назначенным к нему командой Multicast в изменяемом интервале времени.

Реакция системы на ошибку в течение фазы передачи данных DPM1 (то есть, сбой DP слейва) определяется параметром конфигурации "авто-очистка". Если этот параметр установлен в истину, DPM1 включает выходы всех назначенных DP слейвов в отказоустойчивое состояние, как только DP слейв больше не готов для передачи данных пользователя. DPM1 затем изменяется на состояние Clear. Если этот параметр- ложь, DPM1 остается в состоянии Operate даже, когда происходит неисправность, и реакцию системы определяет пользователь.

1.4 Циклическая Передача данных между DPM1 и DP слейвами
Передача данных между DPM1 и DP слейвами назначеннных к нему, выполняется автоматически DPM1 в определенном, возвратном порядке. При конфигурировании системы шины, пользователь определяет назначение DP слейвов к DPM1 и какие DP слейвы должны быть включены или исключены из циклической передачи данных пользователя.
Передача данных между DPM1 и DP рабами разделяется на три фазы:
параметризация, конфигурирование ипередача
В течение фаз параметризации и конфигурации, каждый DP-слейв сравнивает реальную конфигурацию и ожидаемую конфигурацию, которая была конфигурирована в DPM1. Только, если реальная конфигурация соответствует желательной, DP-слейв будет включен в фазу передачи данных пользователя. Следовательно, тип устройства, формат и длина информации также как число вводов и выводов должны соответствовать фактической конфигурации. Эти тесты обеспечивают пользователя надежной защитой от ошибок параметризации. В дополнение к передаче данных пользователя, которая выполняется автоматически DPM1, новые данные парамеризации, могут быть посланы DP слейвам при запросе пользователя.

1.5 Циклическая Передача данных между DPM1 и Устройствами Конфигурации
В дополнение к функциям мастер-слейв, доступны функции связи мастер-мастер. Эти функции дают возможность устройствам конфигурации и диагностики конфигурировать систему над шиной.

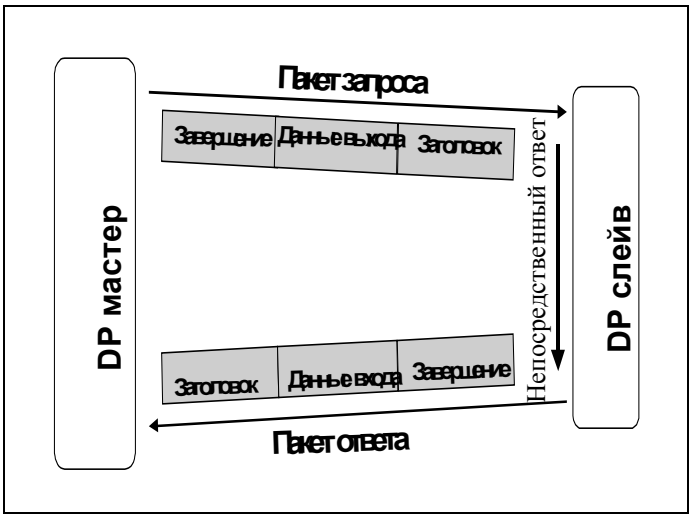


Рисунок 4: Передача данных пользователя с PROFIBUS-DP

В дополнение к передачи и функциям загрузки, функции мастер-мастер позволяют динамическое предоставление или отключение передачи данных пользователя между DPM1 и индивидуальными DP слейвами. Действующее состояние DPM1 может также быть изменено.

1.6 Режимы Sync и Freeze

В дополнение к связанной со станцией передаче данных пользователя, которая выполняется автоматически DPM1, мастер может посылать команды управления одиночному слейву, группе слейвов или всем слейвам одновременно. Эти команды управления передаются как команды Multicast. Они разрешают использование режимов sync и freeze для управляемой событием синхронизации DP слейвов.
Слейвы начинают **sync режим**, когда они получают sync команду от их мастера. Выводы всех адресованных слейвов закрепляются в их текущих состояниях. В течение последующих передач данных пользователя, выходные данные сохраняются в слейвах, но состояние вывода остаётся неизменяемым. Сохраненные выходные данные не посылаются выводам до тех пор, пока не получена следующая sync команда. Sync режим связан с командой unsync.

Точно так же команда управления freeze заставляет адресованных слейвов принимать режим freeze. В этом режиме действия, состояния вводов закрепляются с текущими значениями. Входные данные не модифицируются до тех пор, пока мастер не посылает следующую команду freeze. Режим Freeze связан с командой unfreeze.

Функция	Значение	DPM1	DPM2
Get_Master_Diag	Чтение диагностических данных DPM1 или резюме диагноза DP слейвов.	M	O
Download/Upload Gruppe (Start_Seq, Down-/Upload, End_Seq)	Загрузка или передача всех данных конфигурации DPM1 и связанных DP слейвов.	O	O
Act_Para_Brct	Активиация параметров шины одновременно для всех DPM1 адресованных устройств.	O	O
Act_Param	Активация параметров или изменение действующего состояния DPM1 адресованного устройства.	O	O
M: Обязательно, O: дополнительно			

Таблица 1: Краткий обзор функций мастер-мастер с PROFIBUS-DP

1.7 Механизмы защиты

Защита и надежность делает необходимым обеспечить PROFIBUS-DP эффективными функциями защиты против ошибок параметризации или сбоя оборудования передачи. Текущий временной контроль предусмотрен для мастера DP и DP слейвов. Интервал контролирования определяется в процессе конфигурирования.

В DP мастере

DPM1 контролирует передачу данных слейвов с Data_Control_Timer. Отдельный таймер управления используется для каждого слейва. Этот таймер истекает, когда правильная передача данных не происходит внутри интервала контролируемого. Пользователь информируется, когда это случается. Если допущена автоматическая реакция на ошибку

(Auto_Clear = Истина), DPM1 выходит из состояния Operate, переключает выводы всех назначенных слейвов в отказоустойчивое состояние и изменяется на состояние Clear.

В DP слейве
Слейвы используют управляющий сторож, для обнаружения сбоев мастера или линии передачи. Если никакая связь данных с мастером не происходит внутри интервала управления сторожа, слейв автоматически включает выводы в отказоустойчивое состояние.

Кроме того, для вводов и выводов DP слейвов, действующих в системах мульти-мастер, требуется защита доступа. Это гарантирует, что прямой доступ может выполняться только уполномоченным мастером. Для всех других мастеров, слейвы предлагают изображение их вводов и выводов, которые могут читаться любым мастером, даже без прав доступа.

2.0 Расширенные функции DP
Расширенные функции DP позволяют нециклические функции чтения и записи и прерывают подтверждение, которое нужно выполнить параллельный циклической передаче данных. Кроме того, нециклический доступ к параметрам и значениям измерений слейвов могут выполняться для нескольких станций диагностики и управления оператора (мастер класса -2, DPM2). С этими расширенными функциями PROFIBUS-DP встречает требования сложных устройств, которые часто должны быть параметризованы в течение работы. Типичные примеры включают полевые устройства, используемые в автоматизации процесса, интеллектуальном управлении оператора и устройств текущего контроля, таких как конвертеры частоты. В сравнении с циклическими значениями измерений, эти параметры редко изменяются. По этой причине, передача выполняется с более низким приоритетом параллельно с быстродействующей циклической передачей данных пользователя.
Эти расширенные функции дополнительные. Они совместимы с базисными функциями PROFIBUS-DP. Существующие устройства, которые не хотят или должны использовать новые функции, могут продолжать использоваться, так как расширенные функции - только добавления к уже существующим базисным функциям. ASIC доступный на рынке может продолжать использоваться. DP расширения обычно выполняются программными модификациями.

DP расширения определены в техническом руководстве PROFIBUS Номер 2.082.

2.1 Расширенная Связь Данных между DPM1 и DP слейвами
Нециклические функции связи между классом -1 DP мастером (DPM1) и DP слейвами выполнены через дополнительный пункт сервисного доступа 51. В сервисной последовательности, DPM1 устанавливает подключение со слейвом, который обозначен как MSAC_C1. Учреждение подключения близко связано с подключением для циклической передачи данных между DPM1 и слейвами. После того, как подключение было установлено успешно, DPM1 может выполнять циклическую передачу данных через MSCY_C1 подключение и нециклическую передачу данных через MSAC_C1 подключение.

2.1.1 Нециклические Функции чтения/записи с DDLM_READ и DDLM_WRITE
Эти функции используются, для доступа чтения или записи к любому нужному блоку данных в слейве. Используется SRD обслуживание уровня 2. После передачи запроса DDLM_READ/WRITE, мастер опрашивает слейв с SRD телеграммами до тех пор, пока не прибывает соответствующий DDLM_READ/WRITE ответ.
Адресация блоков данных принимает, что DP слейвы модульные в физическом проекте или могут быть структурированы внутри логических функциональных модулей (модули). Эта модель также используется в базисных функциях DP для циклической передачи данных, где каждый модуль имеет постоянное число входных и-или выходных байтов, которые передаются в фиксированной позиции в телеграмме данных пользователя. Адресация основана на идентификаторах (то есть, вводе или выводе, типе данных, и т.д.) Все эти идентификаторы слейва вместе составляют конфигурацию слейва, которая также проверяется DPM1 в течение запуска.

Эта модель также используется как основание новых нециклических услуг. Все блоки данных, допускаемые для чтения или доступные для записи также рассматриваются как принадлежность к модулям. Эти блоки могут быть адресованы номером слота и индексом. Номер слота адресует модуль, а индекс адресует блоки данных, принадлежащие к модулю. Каждый блок данных может содержать до 256 байтов.

Когда включаются модульные устройства, номер слота назначается модулям. Начиная с 1, модули нумеруются последовательно в увеличивающемся порядке. Слот номер 0 предусмотрен непосредственно для устройства.
Компактные устройства обрабатываются как один модуль из действительных модулей. Адресация с номером слота и индексом также используется здесь.

С использованием спецификации длины в запросе чтения или записи, могут читаться или записываться части блока данных. Если доступ к блоку данных был успешен, DP слейв отвечает положительным ответом чтения или записи. Если не успешный, DP слейв дает отрицательный ответ, в котором точно классифицирована проблема.

2.1.2 Подтверждение тревоги
Базисные функции PROFIBUS-DP разрешают DP слейвам спонтанно передавать события мастеру с диагностическим сообщением. Когда диагностические значения очень динамические, необходимо корректировать частоту передачи с быстродействием PLC. Новая функция DDLM_ALARM_ACK обеспечивает это управление потоком данных. Это используется, чтобы явно подтвердить тревоги, полученные от DP слейва.

2.2 Расширенная Передача данных между DPM2 и слейвами

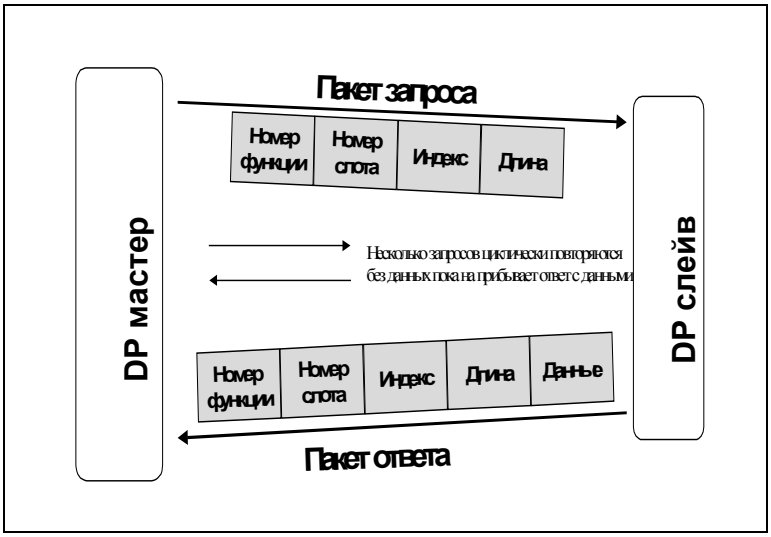


Рисунок 5: Выполнение обслуживания чтения

DP расширения позволяют одному или большему числу устройств диагностики или управления оператора (DPM2) выполнять услуги нециклического чтения или записи на любой желательный блок данных DP слейва. Связь выполняется ориентируемой на подключение. Подключение называется MSAC_C2. Новое обслуживание DDLM_INITIATE используется, чтобы установить подключение прежде, чем начинается передача данных пользователя. Успешное учреждение подключения подтверждается слейвом с положительным ответом (DDLM_INITIATE. res).
Подключение может теперь использоваться для передачи данных пользователя с DDLM_READ и DDLM_WRITE услугами. Между передачами данных пользователя разрешаются паузы любой длины. Если требуется, мастер автоматически вставляет контролирующие телеграммы (Idle_PDUs) в течение этих пауз. Это обеспечивает MSAC_C2 подключение с управляемым временем автоматическим текущим контролем подключений. Контролирующий интервал определяется с DDLM_INITIATE обслуживанием, когда подключение установливается. Если монитор подключения обнаруживает неисправность, подключение автоматически разъединяется, и на стороне мастера и на стороне слейва. Подключение может затем быть установлено снова или использоваться другим партнером. Пункты сервисного доступа от 40 до 48 на слейве и пункт сервисного доступа 50 на DPM2, зарезервированы для MSAC_C2 подключения.

3.0 Файлы Базы данных Устройств (GSD) Разрешают Открытую Конфигурацию
PROFIBUS устройства имеют различные характеристики эффективности. Свойства отличаются по отношению к доступным функциональным возможностям (то есть, число сигналов Ввода - вывода и диагностических сообщений) или возможных параметров шины типа скорости в бодах и текущего времени контроля. Эти параметры изменяются индивидуально для каждого типа устройства и производителя. Эти параметры обычно зарегистрированы в техническом руководстве. Чтобы достичь простой установки PROFIBUS, характерные свойства определяются в электронном листе данных, иногда называемом файлом базы данных устройства или GSD файлом. Стандартизированные GSD данные разворачивают открытую связь до уровня управления оператора. Использование инструментальных средств конфигурации, основанных на GSD файлах

делает интеграцию устройств от различных продавцов в систему шины, простой и дружественной.
Файлы Базы данных Устройств обеспечивают ясное и всестороннее описание характеристик типов устройств в точно определенном формате. GSD файлы подготавливаются индивидуально продавцом для каждого типа устройств и делаются доступными пользователю в форме листа базы данных устройства и файла базы данных устройства. Точно определенный формат файла разрешает системе конфигурации просто читать в файле базы данных устройства любого PROFIBUS-DP устройства и автоматически использовать эту информацию при конфигурировании системы шины. Инжениринг проекта экономит отнимающую много времени работу по определению этой информации из технических руководств. Во время фазы конфигурации, система конфигурации автоматически выполняет проверку входных ошибок и непротиворечивость введенных данных по отношению ко всей системе..

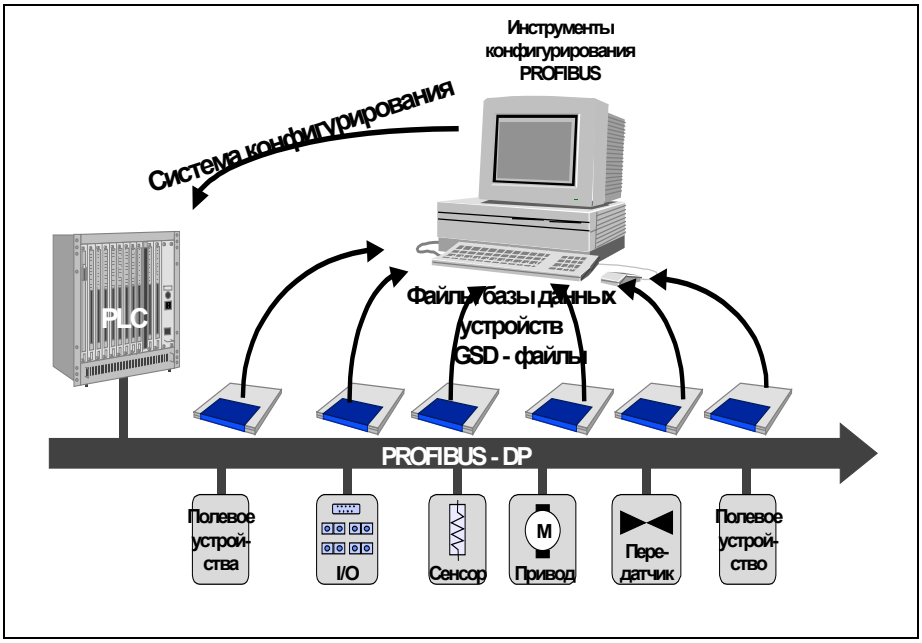


Рисунок 6: Базы данных устройств позволяют открытое конфигурирование

Файл базы данных устройства разделен на три части:

- **Общие спецификации**
  Этот раздел содержит имена продавца и устройства, версии аппаратного и программного обеспечения, поддержанние скорости в бодах, возможные интервалы времени для текущего контроля и назначение сигнала на соединителе разъема шины.
- **Спецификации относящиеся к DP мастеру**
  Этот раздел содержит все параметры, которые только относятся к устройствам DP мастера (то есть, максимальное число слейвов, которые могут быть подсоединены или возможности загрузки-разгрузки). Этот раздел не существует для устройств-слейвов.
- **Спецификации относящиеся с DP слейвами**
  Этот раздел содержит все спецификации, связанные со слейвами (то есть, номер и тип каналов Ввода - вывода, спецификация диагностических тестов и информации относительно непротиворечивости данных Ввода - вывода).

В индивидуальных разделах параметры отделяются ключевыми словами. Различие сделано между обязательными параметрами (то есть, Vendor_Name) и дополнительными параметрами (то есть, Sync_Mode_supported). Определение групп параметров делается выбором из возможных параметров. Кроме того, могут быть подсоединены файлы битового массива с символами устройств . Для гибкости разработан формат GSD. Он содержит оба списка (то есть, скорость в бодах поддержанная устройством) и область для описания модулей, доступных в модульном устройстве. Диагностическим сообщениям может также быть назначен простой текст.
PROFIBUS Организация Пользователя предлагает удобный в работе редактор (доступный под порядковым номером 5.003) для подготовки GSD файлов, которые будут соответствовать стандарту.

3.1 GSD Библиотека на www
GSD файлы всех PROFIBUS-DP устройств, которые проверены на согласование PROFIBUS стандарту, доступны в GSD библиотеке на WWW сервере PROFIBUS Организации Пользователей для свободной загрузки. Адрес**: http: // www.profibus.com**.

3.2 Номер индекса
Каждый тип DP слейва и каждый тип класса -1 DP мастера должен иметь идентификационный номер. Мастера требуют этот номер для возможности идентифицировать типы соединенных устройств, без создания значительнго протокола. Мастер сравнивает идентификационный номер связанных DP устройств с идентификационным номером, определенным инструментом конфигурирования в данных конфигурации. Передача данных пользователя не начинается, пока правильные типы устройств с правильными адресами станции не будут связаны на шине. Это обеспечивает высокую степень защиты против ошибок конфигурирования.
Продавцы должны просить идентификационный номер в PROFIBUS Организации Пользователей для каждого типа DP слейва и для каждого типа класса -1 DP мастера. PROFIBUS Организация Пользователей осуществляет администрирование идентификационных номеров. Заявления - анкеты доступны из всех региональных офисов.

3.4 PROFIBUS-DP конфигурации
PROFIBUS-DP протокол определяет, как данные пользователя должны передаваться между станциями по шине. Данные Пользователя не оцениваются PROFIBUS-DP протоколом передачи. Значение определяется в конфигурациях. Кроме того, конфигурации определяют, как PROFIBUS-DP должен использоваться в области приложения. При использовании конфигураций, операторы завода и конечные пользователи имеют преимущество вожможности заменять устройства от различных продавцов. Конфигурации также значительно уменьшают издержки разработки для пользователя, так как значения связанных с приложениями параметров точно определено. Использование конфигураций позволяет заменять индивидуальные компоненты от различных продавцов без оператора завода, отмечающего различие. До настоящего времени были определены следующие конфигурации PROFIBUS-DP и доступны, используя номер документа в первоисточниках:

NC/RC конфигурация (3.052)
Эта конфигурация описывает, как манипулятор и роботы сборки управляются через PROFIBUS-DP. Основанный на детализированных последовательных диаграммах,

движение и программа управления роботами описывается с точки зрения средств автоматизации более высокого уровня.

Конфигурация кодировщика (3.062)
Эта конфигурация описывает подключение к PROFIBUS-DP вращательных кодировщиков, угловых кодировщиков и линейных кодировщиков с разрешающей способностью мульти-поворота или моно-поворота. Два класса устройств определяют основные и дополнительные функции типа масштабирования, обработки прерывания и расширенной диагностики.

Конфигурация привода переменного быстродействия (3.071)
Ведущие изготовители приводной технологии совместно определили конфигурацию PROFIDRIVE. Конфигурация определяет, как приводы должны быть параметризованы и как должны быть переданы уставки и фактические значения. Это дает возможность заменять приводы от различных продавцов. Конфигурация содержит необходимые спецификации для управления быстродействием и позиционированием. Это определяет основные функции привода при оставлении достаточной свободы для специфических для приложения расширений и дальнейших достижений. Конфигурация описывает отображение функций приложения для DP или FMS.

Конфигурация операторного управления и текущего контроля процесса (HMI)
Эта конфигурация простого управления оператора и текущего контроля процесса устройств (HMI) определяет подсоединение этих устройств через PROFIBUS-DP к компонентам автоматизации более высокого уровня. Конфигурация использует расширенные функции связи PROFIBUS-DP. Конфигурация была в процессе подготовки, когда эта брошюра готовилась к печати.
*) Источник: “PROFIBUS – Техническое описание” of the PROFIBUS Nutzerorganisation e.V.
PROFIBUS Организация Пользователей и присоединившиеся к ней организации во всех главных индустриальных странах представляют интересы всех заинтересованных сторон.
Для дополнительной информацией обращаться :
ROFIBUS Nutzerorganisation, HAID-und-NEU-Str. 7, D-76131 Karlsruhe
Tel.: ++49 721 / 9658 590 Fax: ++49 721 / 9658 589 или в Интернете
http://www.profibus.com

Общие замечания

SIMATIC S7-200 Советы по применению обеспечиваются для того, чтобы дать пользователям S7-200 некоторое представление относительно того, как, некоторые задачи из представления методики программирования, могут быть решены с этим контроллером. Эти инструкции не предполагают отражать все детали или разновидности оборудования, и при этом они не предусматривают любое возможное непредвиденное обстоятельство. Использование S7-200 Советов по применению свободное.
Siemens оставляет право делать изменения в спецификациях, показанных здесь или делать усовершенствования в любое время без примечания или обязательства. Это не освобождает пользователя от ответственности, при использовании звуковых методов в прикладной программе, установке, операции, и сопровождении приобретенного оборудования. Если возникает конфликт между общей информацией, содержащейся в этой публикации, содержанием рисунков и дополнительным материалом, или обоими, последние должны иметь приоритет.
Siemens не несет ответственности, по любой допустимой причине, в повреждениях или персональном ущербе, последующем из использования советов по применению.

Все права защищены. Любая форма дублирования или распространения, включая выдержки, разрешается только с конкретным разрешением SIEMENS.

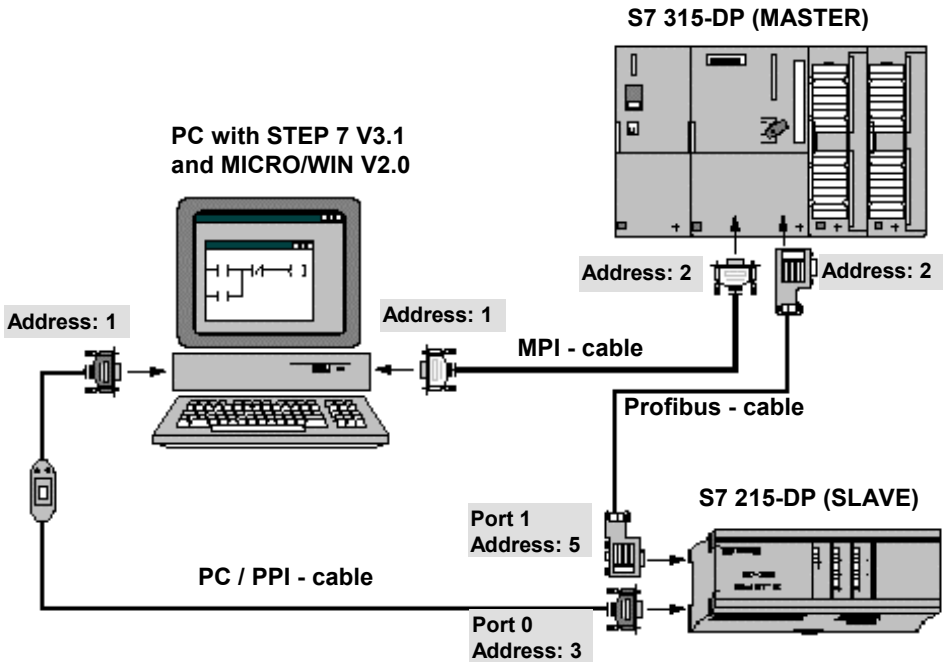
SIMATIC S7-200 Советы

Група	Тема
3	**Соединение S7-300 (мастера) с S7-200 (слейвом) с помощью Profibus**

Требуемые для этого совета CPU					
CPU 210 ☐	CPU 212 ☐	CPU 214 ☐	CPU 215 ☑	CPU 216 ☐	ДРУГИЕ ☐

Обзор

В этом примере, S7-215 DP связан с помощью интерфейса с S7-315 DP посредством Profibus.
S7-315 DP - мастер, и S7-215 DP используется исключительно как устройство слейв.
Чтобы гарантировать безупречную связь между хозяином DP и CPU215 DP, на участвующих устройствах должны быть сделаны некоторые настройки параметров. Параметры настройки / инициализации мастера с программным обеспечением программы STEP7 (начиная с Версии 3.1) будут объясняться подробно ниже.
Типовая программа для S7-200 показывает, как Вы можете применять вводы S7-300 к выводам S7-200. Кроме того, это даёт возможность представить вводы S7-200 на модуле вывода S7-300.



Аппаратные требования

1 S7-215-DP
1 S7-315-DP (Состоящий из блока питания PS307-2A, CPU S7-315-2 DP и Цифровой модуль ввода -вывода SM374 соответственно для моделирования)
1 Siemens PC/PPI кабель для программирования S7-200
1 Profibus кабель (должен быть завершен на двух концах кабеля через соответствующий резистор встроенный в соединитель = включён!)
1 MP кабель для программирования S7-300
1 PC с MPI платой (И программное обеспечение STEP 7 Версии 3.1 Также как MICRO /WIN Версия 2.0)

Установка DP слейв-адресов (S7-215-215)

A.) Со STEP7-Micro/DOS V1.3

1.) Идите интерактивно (F4) и измените в меню ' Утилиты (панель управления= > F6). Режим работы должен быть TERM.

2.) В меню ' Параметры Связи, установите DP адрес CPU215 (AS номер утилиты 10).

3.) Сохраните адрес в CPU клавишей F3, запись. Затем, выйдите из меню ' Параметры Связи и ' "Утилиты" с помощью ESC. Активизируйте панель управления и клавишу 'ASSSPEI' (F8), чтобы сохранить изменения в CPU.

B.) Со STEP7-Micro/WIN V2.0

1.) Откройте новый проект и выберите тип CPU -CPU215.

2.) В пункте меню 'CPU', выберите под-меню 'Выбор конфигурации', и нажмите на 'DP-ПОРТ'.
Установите желаемый адрес CPU в области от 0 до 126 в соответствующем поле.
Обратить внимание: поле 'Baudrate' неприменимо к интерфейсу DP, поскольку мастер всегда устанавливает Baudrate.

3.) В меню 'Проект', под-меню ' Загрузка из PG ', загрузите конфигурацию CPU в CPU. Это установит адрес DP в CPU.

Configuring the DP Master (S7-315 DP)

Со STEP7 (Версия 3.1):

S7-CPU315 DP конфигурируется как DP мастер с STEP7. Следующие шаги относятся к Версии v3. 1 STEP7. Могут иметься отклонения относительно предыдущих или более поздних версий.

Шаги для инициализации S7-315 DP:

Запуск Step 7 S7/M7 (Используемая версия 3.1)

1. Установка нового проекта:

Ввод имени проекта : например, 'DP_Connect'
Подтверждение ввода нажатием кнопки 'OK'

2. Выбор компонентов:

a) Добавление станции
Меню: 'Insert - Station - Simatic S7-300 Station'
⇨ Символ для станции Simatic S7-300 вставляется в текущую директорию и отображается.

b) Ввод соединения между S7-300 и S7-200
Меню: 'Insert - Subnet - Profibus'
⇨ Символ для Pofibus Subnet вставляется в текущую директорию и отображается.

3. Конфигурирование аппаратного обеспечения S7-300:

A) Запуск программы аппаратного конфигурирования

- Двойной щелчок (левая кнопка мыши) на символе 'SIMATIC 300 Station(1)'

⇨ Директория станции SIMATIC 300 должна быть открыта! Должен быть отображен почтовый ящик с названием 'Hardware'.

- Жвойной щелчок (левая кнопка мыши) на символе для 'Hardware'

⇨ Начинается программа аппаратного конфигурирования под названием 'HWKONFIG' . В добавление, окно с каталогом аппаратуры должно быть открыто

B) Конфигурирование аппаратуры S7-300 с помощью 'HWKONFIG'

Аппаратное обеспечение S7-300 (используемое оборудование: CPU , модули ввода-вывода) определяется с помощью 'HWKONFIG' .

B1.0 Вставка СТОЙКИ:

- Открыть катталог SIMATIC 300 , выбирать знак "ПЛЮС" при входе в 'SIMATIC300' левой кнопкой мыши

⇨ Войти в открытый 'SIMATIC300' ; отобразятся различные под-элементы

- Под пунктом ' СТОЙКА 300 ', выберите вход 'РЕЛЬС', дважды нажимая на нём левой кнопкой мыши

⇨ В форме таблицы, выбранная S7-300 СТОЙКА вставляется в Окно для аппаратной конфигурации.

B2.0 Вставка компонентов СТОЙКИ

- С помощью мыши, выберите окно ' Аппаратная Конфигурация ' (где размещена таблица Для СТОЙКИ). Если окно или таблица слишком маленькое, Вы можете сделать его больше / меньше, устанавливая его размеры мышью, и корректировать его как желательно.
- Чтобы вставлять компонент в стойку, нажмите на первую линию таблицы СТОЙКИ.
  - ⇨ Линия таблицы имеет синий фон, если Вы её выбрали.
- Теперь, один за другим, аппаратные компоненты вставляются в слоты (Линии) стойки.

B2.1 Вставка Питания (не абсолютно необходимы для конфигурации!)

- В ' аппаратном каталоге ', выберите элемент 'PS-300', и ниже его питание , которое Вы используете.
  - ⇨ В этом примере, аппаратные средства оборудованы модулем питания 'PS-307-2A'. Вы должны выбрать это левой кнопкой мыши. Модуль 'PS-307' помещен в первую позицию в стойке (Имя питания обозначено на первой линии таблицы).

B2.2 Вставка CPU -315 DP (нуждается в двух слотах!)

- Выберите Слот Два, щелкая на нем мышью.
- В ' аппаратном каталоге ', откройте директорию'CPU-300'
- С пмощью мыши выберите точно CPU 315 DP. С тех пор как там Два разных,порядковый номер установки аппаратных средств и программного обеспечения должна согласовываться.
  - ⇨ <u>Здесь:</u> выберите CPU-315 DP с порядковым номером: 6ES7 315-2AF00-OAB0 (важно: этот номер должен быть тот же самый что и на CPU!)
- Двойной щелчок левой кнопкой мыши на правильном CPU-315 DP; появляется диалог, где Вы должны определить параметры Profibus. Установите требуемый адрес Profibus (например, 2) и другие параметры Profibus (Или используйте значения по умолчанию). После того, как Вы выбрали Profibus подсеть, и все параметры для Profibus были установлены, нажмите кнопку 'OK'.
  - ⇨ CPU, включая Profibus сеть вставлен в таблицу под Питание.

B2.3 Вставка Модулей Ввода -вывода

Выбор модулей ввода -вывода при входе 'SM-300'

- ⇨ В этом случае,должны быть вставлены два цифровых модуля ввода -вывода . Первый модуль конфигурирован как цифровой входной модуль (16 битов) и второй как цифровой выходной модуль (16 битов).

B2.3.1 Выбор Цифрового Входного Модуля

Двойное нажатие левой кнопки мыши на обозначенном модуле:
' SM321 DI16XDC24V (Ввод 16DI 24V, группируя 16) ' и порядковый номер 6ES7-321-1BH01-0AA0

⇨ Модуль будет вставлен в четвертый слот.Начальный адрес входного модуля определен системой как 0.

B2.3.2 Выбор Цифрового Модуля Вывода

Двойное нажатие левой кнопки мыши на обозначенном модуле:
' SM322 D016xDC24 (Вывод 16D0 24V/0,5A, группируя 8) ' и порядковый номер 6ES7 322-1BH01-0AA0

⇨ Модуль будет вставлен в пятый слот.Начальный адрес модуля вывода - определен системой как 4 .

Аппаратная конфигурация для S7-300 - относительно завершена; только различные адреса все еще должны быть установлены правильно.

4. Установка Различных Адресов

a. Установка Адреса MPI для Связи между PC и S7-300:

В окне ' Аппаратная Конфигурация ', S7-315 DP был вставлен во второй линии таблицы. Нажмите на эту вторую линию правой кнопкой мыши. Появится ниспадающий диалог.

После выбора элемента 'Свойства Объекта Свойства ALT + RET ', следует окно диалога с именем ', Свойства CPU 315-2-DP '. Адрес MPI должен быть установлен под этим диалогом.

⇨ В этом случае, адрес MPI установлен в 2.

Параметры настройки и диалог закрываются кнопкой 'OK'.

b. Установка Адреса Profibus для Связи между S7-300 и S7-200:

В окне ' Аппаратная Конфигурация ', S7-315 DP был вставлен в второй и третьей линиях таблицы. Правой кнопкой мыши, нажмите на третью линию (DP мастер). Появляется ниспадающий диалог. После выбора элемента ' Объектные Свойства ALT + RET ', следует окно диалога, под названием ' Свойства DP мастера '. Адрес Profibus должен быть установлен под этим диалогом. В этом случае, адрес Profibus установлен в '2'.

Параметры настройки и диалог закрываются кнопкой 'OK'.

5. Проверка адресов Цифровых Модулей Ввода -вывода

- В окне ' Аппаратная Конфигурация ', выберите линию с модулями Ввода - вывода.
- Нажмите на эту линию правой кнопкой мыши.
  - ⇨ Появится ниспадающий диалог.
    После выбора элемента ' Объектные Свойства ALT + RET ', появляется окно диалога под названием: ' Свойства DI16XDC24V или DO16XDC24V '.
- Под Адресом, можно прочитать установленные адреса модулей ввода -вывода.
  - ⇨ В этом примере, адрес входного модуля установлен в 0, адрес Модуля вывода установлен в 4.
- Параметры настройки и диалог закрываются клавишей 'OK'.

6. Вставка S7-215 DP в Подсеть

- В окне ' Аппаратная Конфигурация ', выберите символ (штриховая линия) для Profibus, щелкая на нем левой кнопкой мыши.
- Откройте каталог Profibus, выбирая знак "ПЛЮС" перед входом ' PROFIBUS.
  - ⇨ Вход ' PROFIBUS открывается; отображаются различные под-элементы
- Под элементом 'SIMATIC', выберите вход ' S7-200 CPU 215 DP ' двойным нажатием на нем (левая кнопка мыши)
  - ⇨ Выбранный S7-215 DP вставлен в окно для аппаратной конфигурации (Как новая таблица, добавлена к Profibus).

7. Определение Параметров Связи для S7-215 DP

- После того, как таблица S7-215 DP была добавлена к Profibus, выбирать первую линию таблицы (слейв).
- В аппаратном каталоге, под S7-200 CPU-215 DP, выбирают тип связи (Байт, Слово или буфер). В нашем случае,требуется слово для передачи слов.
  - ⇨ Выбрана передача "1 Слово из / 1 Слово В".

С этим, связь S7-215 DP с S7-315 DP через Profibus определена. Ввод-вывода и область ввода- вывода определены для Адреса 256.

8. Загрузка Аппаратной Конфигурации в S7 315-DP

После того, как все параметры настройки, упомянутые выше были сделаны, аппаратная конфигурация может быть Загружена в CPU через 'загрузку'. Удостоверитесь, что PLC находится в режиме-' ОСТАНОВ-' ; То есть переключатель режима должен быть на ' ОСТАНОВ '.

Загрузка аппаратной конфигурации может быть активизирована в меню под 'PLC/DOWNLOAD' или через кнопку 'download'.

Адреса связи (Profibus, MPI, PPI)

Например могут быть установлены следующие адреса:

Profibus:	**Мастер**	**Адрес (S7-315 DP) = 2**
	Слейв	**Адрес (S7-215 DP, Port1) = 5**
MPI:	**PC**	**Адрес (MPI-Card) = 1**
	Мастер	**Адрес (S7-215 DP,Port0) = 2**
PPI	**PC**	**Адрес (PC) = 1**
	Слейв	**Адрес (S7-215 DP) = 3**

Типовая Программа в DP Мастере (S7-315 DP)

Со STEP7 (Версия 3.1):

Типовая программаDP мастера расположениа в OB1. Она очень простая; Код непосредственно состоит из следующих линий (только одна сеть):

Сеть **1:**

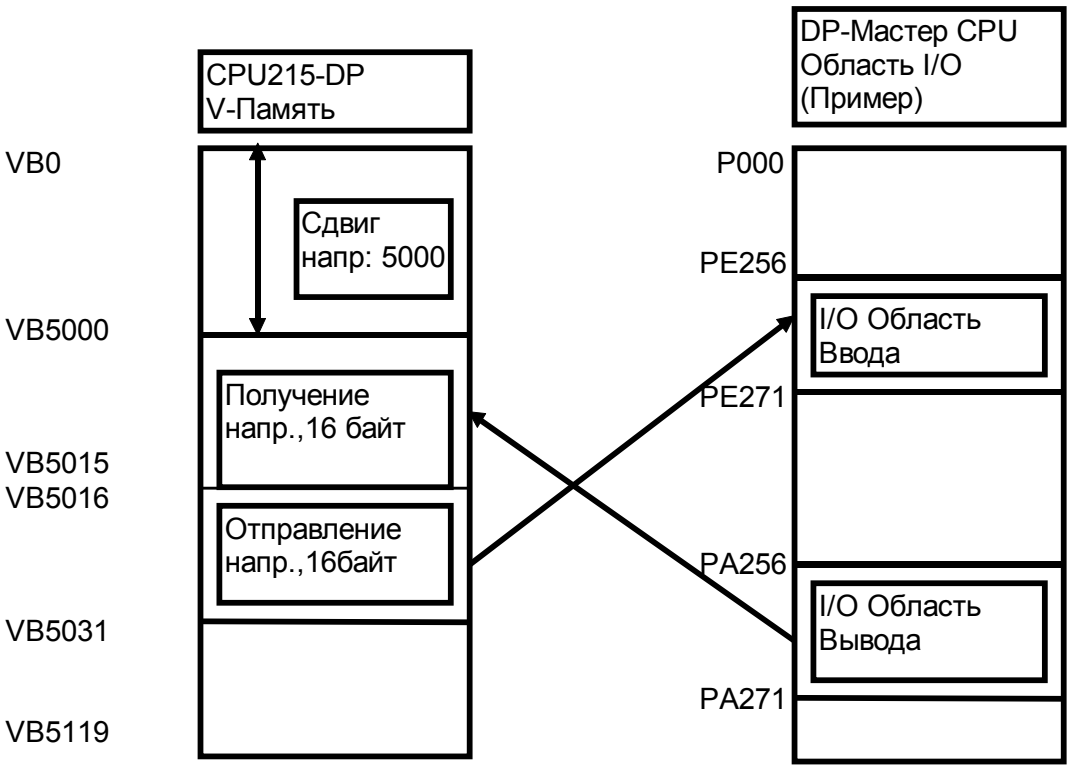
```
// Передача Ввода мастера (S7-315 DP) к слейву (S7-215 DP)
L     PIW   0     // Выбор слова ввода- вывода входного модуля I/O
T     PQW   256   // Передача его в адресную область 256
                  // (ящик отправления мастера или выходная область I/O)
// Передача ввода слейва (S7-215 DP) к мастеру (S7-315 DP)
L     PIW   256   // Выбор слова ввода I/O 256 (вводы слейва)
T     PQW   4     // Передача его на выходы выходного модуля
BE                // Конец блока
```

Мастер инициализирует передачу данных, читая слово ввода Ввода - вывода входного модуля и сохраняя это в области вывода Ввода - вывода слейва (начиная с Адреса PQ256).

Кроме того, мастер читает область ввода Ввода - вывода (начиная с Адреса PI256) где отображены вводы слейва, и сохраняет слово в физических выходах модуля вывода. Если, например, ввод S7-215 DP активизирован, он будет отображен на S7-315 DP на модуле вывода. Наоборот, ввод S7-315 DP отобразится на S7-215 DP.

Модель память DP-Мастера и DP-Слейва

Цель Profibus подключения - обмен данными. Данные из PLC передаются к другому PLC. Инициатор этой передачи данных обозначается как 'Мастер', а другая станция связи так называемая 'Слейв'. Принцип прост: мастер читает, данные от почтового ящика отправления слейва, и сохраняет это в области ввода Ввода - вывода. Кроме того, мастер передает данные почтовому ящику получения слейва. Так как CPU215 может быть только слейвом,почтовые ящики отправления и получения, находятся в переменной памяти CPU (V-память). Рисунок ниже показывает модель памяти.



Вы можете конфигурировать в мастере, где начало почтового ящика получения - устанавливая смещение. Если смещение - 0, это означает, что почтовый ящик получения начинается в переменном байте VB0. Если смещение - 5000, начало почтового ящика получения в VB5000. Кроме того, размер получающегося почтового ящика может быть определен в байтах, когда конфигурируется мастер. Если размер - 16 байтов, и смещение - 5000, начало почтового ящика получения в VB5000 и конец в VB5015. Размер почтового ящика отправления - является непосредственно связан почтовым ящиком получения - может также быть конфигурирован; то есть если например размер - 16 байтов, почтовый ящик отправления простирается от VB5016 до VB5031.

Обмен данными

Как только CPU 215 был связан с мастером, и информация конфигурации была успешно загружена, CPU 215 введет режим обмена данных. В этом режиме, мастер пошлет выходные данные CPU 215, и CPU 215 ответит с входными данными. V адрес памяти, который обеспечен как часть информации конфигурации, определяет стартовое расположение выходных данных, которые посланы мастером. Немедленно после последнего байта выходных данных - стартовое расположение входных данных. Чтобы знать стартовый адрес входных данных, Вы должны знать, сколько слов выходных данных мастера были конфигурированы для отправки.

Стартовый адрес буферов данных и их длина должны быть известны во время создания программы пользователя для CPU 215 . Выходные данные, посланные мастером должны перемещаться программой в CPU 215 из буфера вывода до различных мест использования. Аналогично, входные данные должны перемещаться с различных мест в буфер ввода для передачи мастеру.

Непротиворечивость Данных
К системе где значения данных больше чем один байт нужно обращаться по одному байту, необходимо убедиться в том, что ко всем всем мультм-байтным значениям обращаются как к модулю, таким образом сохраняя непротиворечивость данных. CPU 215 операционной системы гарантирует непротиворечивость слова до тех пор, пока данные выравниваются на границах слова. Программа пользователя может гарантировать непротиворечивость и двойного слова и буфера, просто выполняя все передачи данных к или из входных и выходных буферов в основной программе или в подпрограмме, вызванной из основной программы. Если данные передаются к или из входных и выходных буферов из подпрограммы прерывания, то нужно ожидать противоречивость двойного слова и буфера.

Описание программы

Краткое объяснение типовой программы для DP связи между S7-215 DP и S7-315 DP обеспечивается ниже. Программа размещается в CPU-215 DP, который представляет DP слейв. В этом примере, слейв CPU (CPU -215 DP) читает данные, которые мастер CPU (CPU315 DP) оглашает и пишет значениям физических выходов слейв CPU. Программа пишет значения физических вводов слейв CPU на мастер CPU.

Текущий контроль Стандартного Протокола DP
С использованием специальных байтов памяти от SMB110 до SMB115 (состояние DP стандартных протоколов), может быть проверено состояние DP стандартного протокола.

MSB 7							LSB 0
0	0	0	0	0	0	s	s

Порт 1: стандартный байт состояния протокола DP

ss: DP Стандартный байт состояния
00 = обнаруженная ошибка конфигурирования /параметризации
01 = В настоящее время в режиме обмена данных
10 = В настоящее время в режиме обмена данных
11 = Выход из режима обмена данных

SM111 к SM115 модифицируются, каждый раз, когда CPU принимает информацию конфигурации -параметризации. Эти расположения модифицируются, даже если обнаружена ошибка конфигурации - параметризации. Эти расположения очищаются, каждый раз, при включении CPU.

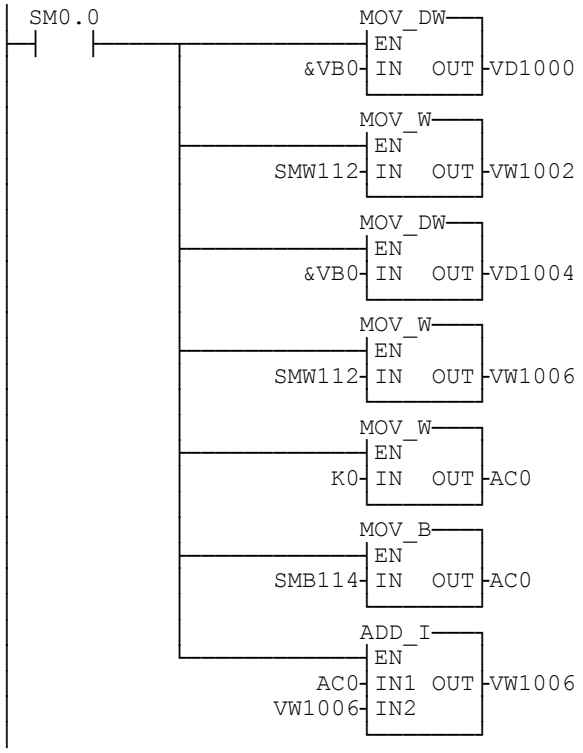
SMB111	Этот байт определяет адрес мастера слейва (от 0 до 126).
SMB112/ SMB113	Эти байты определяют V адрес памяти буфера вывода (Смещение из VB0). SM112 - наиболее значимый байт, а SM113 - наименее значимый байт.
SMB114	Этот байт определяет число байтов для выходных данных.
SMB115	Этот байт определяет количество байтов вводимых данных.

Ladder (S7-MicroDOS)	STL (IEC)

Главная программа

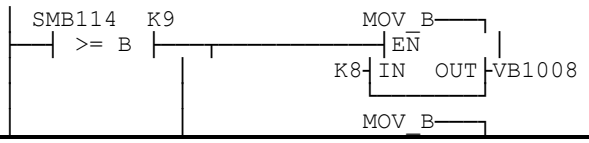
```
//
// Эта программа - универсальная DP слейва программа. Будет получать выводы от мастера и
//писать их в физические выходы PLC. Физические входы от PLC копируются в мастер.
//
// DP данные конфигурации в SM памяти используются,для конфигурирования DP слейва.
// Данные:
//        SM 110  DP статус
//        SM 111  адрес мастера
//        SM 112  Смещение памяти выводов (значение слова)
//        SM 114  число выходных байтов
//        SM 115  число входных байтов
// Мои данные:
//        VD1000 Указатель выходных данных
//        VD1004 Входной указатель данных
```

1

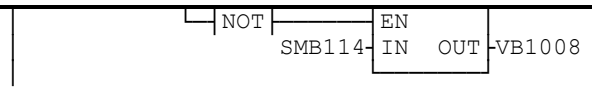
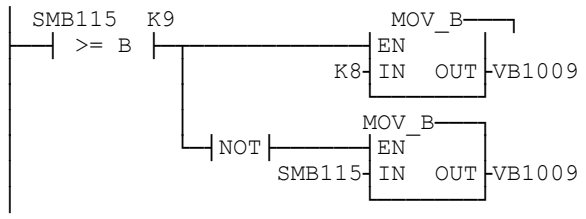
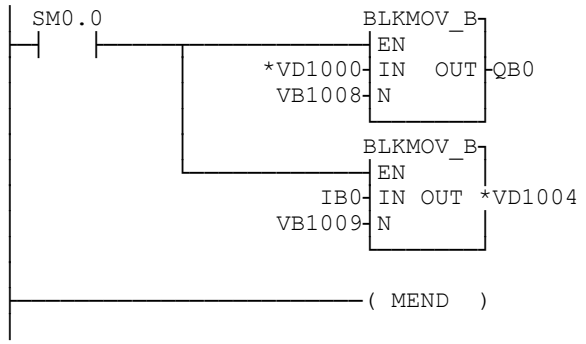



```
LD SM0.0                  // При каждом
                          // просмотре
MOVD &VB0, VD1000         // Создание указателя
                          // вывода

MOVW SMW112, VW1002
                          // Добавьте в смещении
                          // ввода

MOVD &VB0, VD1004
                          // Создайте входной
                          // указатель

MOVW SMW112, VW1006
                          // Добавьте в смещении
                          // вывода

MOVW 0, AC0               // Очистка сумматора

MOVB SMB114, AC0
                          // Получени числа байтов
                          // вывода

+I AC0, VW1006            // указатель
                          // смещения
```

2




```
LDB>= SMB114, 9
        //если число выходных байтов > 8
MOVB 8, VB1008            // счетчик выхода = 8
```

	**NOT // иначе** **MOVB SMB114, VB1008** **// счетчик выхода = числу выходных** **// байт**
3 	**LDB>= SMB115, 9** **// если число входных байт > 8** **MOVB 8, VB1009 // счетчик входа = 8** **NOT // иначе** **MOVB SMB115, VB1009** **// счетчик входа = числу входных** **// байт**
4  5	**LD SM0.0 // при каждом** **// проходе** **BMB *VD1000, QB0, VB1008** **// копирование выходов DP в** **// выходы** **BMB IB0, *VD1004, VB1009** **// копирование входов во входы DP** **MEND**

Примечания преобразования

При преобразовании из IEC STL в S7-Micro/DOS STL:

- Добавлять ‘K’ перед всеми нешестнадцатиричными числовыми константами (т.е. 4 ⇒ K4)
- Заменять ‘16#’ на ‘KH’ для всех шестнадцатиричных констант (т.е. 16#FF ⇒ KHFF)
- Запятые обозначают разделение полей. Используйте стрелки или клавишу TAB, чтобы переключаться между полями.
- Чтобы преобразовывать S7-Micro/DOS программу STL в форму LAD, каждая сеть должна начинаться со слова 'СЕТЬ' и номера. Каждая сеть в Application Tip программы обозначена номером на ступенчатой диаграмме. Используйте команду INSNW под меню EDIT, чтобы ввести новую сеть Команды.MEND, RET, RETI, LBL, SBR, и INT каждая получает свою собственную сеть.

- Линия - комментариев, обозначенная '//' невозможна в S7-Micro/DOS, но Сеть - комментариев возможна.

Показанные блоки данных были созданы в Micro / Win; в Miicro / DOS использование редактора V-памяти для ввода блока данных (Обратитесь к руководству пользователя Micro / DOS). Обратите внимание что в Micro / Win:

- 16#.. обозначает шестнадцатиричные значения

- ‘*text*‘ представляет строковые значения

Общие замечания

SIMATIC S7-200 Советы по применению обеспечиваются для того, чтобы дать пользователям S7-200 некоторое представление относительно того, как, некоторые задачи из представления методики программирования, могут быть решены с этим контроллером. Эти инструкции не предполагают отражать все детали или разновидности оборудования, и при этом они не предусматривают любое возможное непредвиденное обстоятельство. Использование S7-200 Советов по применению свободное.
Siemens оставляет право делать изменения в спецификациях, показанных здесь или делать усовершенствования в любое время без примечания или обязательства. Это не освобождает пользователя от ответственности, при использовании звуковых методов в прикладной программе, установке, операции, и сопровождении приобретенного оборудования. Если возникает конфликт между общей информацией, содержащейся в этой публикации, содержанием рисунков и дополнительным материалом, или обоими, последние должны иметь приоритет.
Siemens не несет ответственности, по любой допустимой причине, в повреждениях или персональном ущербе, последующем из использования советов по применению.
Все права защищены. Любая форма дублирования или распространения, включая выдержки, разрешается только с конкретным разрешением SIEMENS.

SIMATIC S7-200 Tips

Group	Topic
3	**Fault Alarm Recording with S7-200**

CPUs required for this tip					
CPU 210 ☐	CPU 212 ☑	CPU 214 ☑	CPU 215 ☑	CPU 216 ☑	OTHER ☐

Overview

Together with a PC und commercial modems, the Micro-PLC SIMATIC S7-200 is ideally suited for configuring a telemonitoring system. In such a system, it is the task of the S7-200 to record limits and process data of a machine or or plant. If there is an error such as limit underrange or overrange, the S7-200 autonomously dials the phone number of the host computer via a modem. As soon as the connection to the host computer is established, the S7-200 will send a station identification, a fault identification and optionally current process values.

The complete tip is available soon !

Copyright ©1997 by SIEMENS

SIMATIC S7-200 customers have free use of the application tips. These tips are only a general approach to using the S7-200 with various applications. Your specific application may be different. It is your responsibility to use the SIMATIC S7-200 properly in your applications.