Комплекс программный

Фрегат2005

Часть 2.

Руководство программиста

Содержание

1. Программная модель ПЛК CPU-165(166)

1.1. Память

1.1.1. Память программ

В системе в качестве памяти программ используется энергонезависимая память емкостью 8 К слов. Адреса: 0 - 177778

1.1.2. Память данных

ПЛК содержат программно доступные битовые и байтовые области, как указано в Таблице 2.1

В качестве памяти ввода-вывода в системе используется ЗУ емкостью 2 Кбит, а в качестве памяти данных ЗУ емкостью 16 Кбайт.

Данные, записанные в памяти ввода-вывода, не сохраняются при выключении электропитания и при включении выключателя сброса.

Информация, записанная в памяти данных, сохраняется при выключении электропитания, так же как память программы.

Таблица 2.1 Адресное пространство ПЛК

Наименование	Адрес начала	Размер	Примечание
Внешний ввод-вывод	00000	1 Кбит	Битовая адресация.
Внутренний (фиктивный) ввод-вывод	02000	1 Кбит	
Банк данных 0	04000	2 Кбайт	Байтовая адресация.
Банк данных 1	14000	2 Кбайт	
Банк данных 2	24000	2 Кбайт	
Банк данных 3	34000	2 Кбайт	
Банк данных 4	44000	2 Кбайт	
Банк данных 5	54000	2 Кбайт	
Банк данных 6	64000	2 Кбайт	
Банк данных 7	74000	2 Кбайт	

В системе в качестве прикладной команды используется команда функции. Использование команд функции позволяет реализовать на традиционной релейной схеме операции сложения, вычитания, сравнения, передачи данных и др.

1.2. Система команд

Система команд ПЛК использует 16-битную архитектуру совместимую с ПЛК "TOYOPUC", "УПУ-ТП". Длина команды и каждого из параметров - по 2 байта. Адрес ввода-вывода (параметр С, см. примечание 2 к таблице) совмещен с кодом команды.

Таблица 2.2 Система команд ПЛК

Код$_{10}$	**Мнемоника** *1)	**Параметры** *2)	**Длина$_{10}$**	**Комментарий**
0	NOP		2	Нет операций
1	TNA	C	2	AND(И)
2	TFA	C	2	NOT AND(И)
3	TNO	C	2	OR(ИЛИ)
4	TFO	C	2	NOT OR(ИЛИ)
5	TNE	C	2	OR в конце цепи
6	TFE	C	2	NOT OR в конце цепи
7	SKP		2	Пропуск следующей команды
8	SON	C	2	Включение катушки
9	SOF	C	2	Выключение катушки
10	YON	C	2	Включение катушки по условию
11	FON	CFSDR	10	Прикладная команда
12	TIM	CN	4	Таймер
13	CTR	CN	4	Счетчик
15	HLT		2	Стоп
16	JMY	L	2	Условный переход
20	JMN	L	2	Переход по неудовлетворению условия
24	JMP	L	2	Безусловный переход

Примечания:

***1) Колоризация мнемоник:**

Команды	**Комментарий**
TNO,TFO,TNE,TFE	зеленый, вычисления логические OR
TNA,TFA	синий, вычисления логические AND
SON, SOF, YON, FON, TIM, CTR	коричневый, установка битов вывода
SKP, HLT, JMY, JMN, JMP	красный, переходы

***2) Параметры:**

- C - адрес ввода-вывода
- N - число 0-255 параметр таймера, счетчика
- F - номер функциональной команды
- SDR - параметры функциональной команды
- L - метка

1.3. Структура и выполнение программы в ПЛК

Выполнение программы в ПЛК заключается в последовательном выполнении команд, расположенных в памяти команд начиная с адреса 00000. Последовательность выполнения может меняться лишь при выполнении команд перехода.

Программа завершается концевым фрагментом, который включает в себя инструкцию перехода к адресу 00000.

Однократное выполнение программы от адреса 00000 до концевого фрагмента называется сканом.

Алгоритм работы ПЛК состоит из последовательности действий:

1. чтение ввода-вывода;
2. однократное выполнение программы от адреса 00000 до концевого фрагмента (скан);
3. запись ввода-вывода;

Функционально программа состоит из последовательности *цепей*.

Цепь - это последовательность логических команд и ветвлений, завершаемых одной из команд установки битов (катушка, счетчик,таймер, функция).

Цепи условно пронумерованы в порядке по возрастания адресов.

На уровне редактирования программ вводится дополнительное понятие - функциональный *блок*.

Блок состоит из одной или более цепей, выполняющих общую функцию.

Блок может быть включен в программу на этапе сборки проекта. При включении блока все цепи включаются в состав программы.

1.4. Специальные адреса ввода-вывода и памяти

1. Адреса ввода-вывода с 03000_8 по 03477_8 включительно - энергонезависимые.

2. Байт ввода-вывода 03750_8 является аппаратным генератором частоты.

Аппаратный генератор частот:

Имя	Адрес$_8$	Период
АГЧ0_1	3750	0,1с
АГЧ0_2	3751	0,2с
АГЧ0_4	3752	0,4с
АГЧ0_8	3753	0,8с
АГЧ1_6	3754	1,6с
АГЧ3_2	3755	3,2с
АГЧ6_4	3756	6,4с
АГЧ12_8	3757	12,8с

Частоты АГЧ можно использовать для организации выполнения периодических процессов.

3. Бит 03760_8 обеспечивает выполнение функций в каждом скане.

4. Бит 03770_8 - 0 - нормальная работа, 1 - аварийный останов с кодом ошибки 11 (индикатор ошибки мигает 11 раз). Устанавливается пользователем для аварийного останова РКС - программы.

5. Бит 03771_8 и 03772_8 - ошибки в функциональных командах. Устанавливаются функциональными командами и анализируются пользователем.

6. Биты 03773_8 и 03774_8 - выбор дискретности таймера.

03773_8	**03774_8**	**Период**
0	0	0.01с
1	0	0.1с *после старта РКС
0	1	1с
1	1	10с

7. Бит 03775_8 - управление выходом реле готовности: 1- включен, 0 - выключен. Устанавливается пользователем для управления реле готовности.

8. Бит 03776_8 - управление сигналом OUTENABLE: 1- выходы модулей разрешены, 0 - запрещены. Устанавливается пользователем для управления сигналом OUTENABLE.

9. Бит 03777_8 - разряд батареи: 1 - разряжена, 0 - норма. Устанавливается автоматически в зависимости от состояния батареи и анализируется пользователем.

10. Вся память данных и программ - энергонезависима, поэтому техпроцесс можно возобновить после сбоя по питанию.

11. Список абонентов сети “ProfiBus PLUS” выведен в память данных (32 байта начиная с адреса 77740_8 . 00 - абонент отсутствует , FF - присутствует)

12. Поддержка часов реального времени (RTC).
Восемь байт с 77700_8 по 77707_8 включительно зарезервированы под данные часов реального времени, которые представлены в BCD - коде.

Байт	**значение**
77700_8	управление: 0- отсчет времени, не 0- установка времени
77701_8	секунды (0-59)
77702_8	минуты(0-59)
77703_8	часы (0-23)
77704_8	день (1-31)
77705_8	месяц (1-12)
77706_8	год (0-99)
77707_8	день недели (0-6)

Для установки времени выполняется последовательность действий:

1. в байт управления 77700_8 записывается 1;

2. в байты 77701_8-77707_8 - новое время (в BCD - коде);
3. в байт управления 77700_8 - 0.

1.5. Символические имена

В исходных текстах преимущественно используются символические имена соответствующие контактам ввода-вывода и внутренним областям памяти контроллера. Распределение адресов символьных имен происходит автоматически либо вручную, в соответствии с настройками, при сборке проекта.

При написании и отладке программы Фрегат2005 рекомендуется использовать символические имена элементов вместо их адресов, что позволяет сохранять и повторно использовать функциональные блоки в разных проектах.

Автоматическая система распределения адресов настраивается субмодулем «Таблица символов» на различные варианты аппаратной конфигурации.

Программа Фрегат-2005 может быть использована без изменения исходных текстов при изменении аппаратной конфигурации ПЛК.

1.5. NetPlc

Логический контроллер NetPlc - это программная реализация специализированного ПЛК для персонального компьютера.

Основные задачи:

1. обмен данными с сетью ПЛК по заданному расписанию;
2. выполнения программ обработки принятых данных;
3. регистрация данных и событий в базе данных;
4. реакции на аварийные сигналы;
5. взаимодействие с формами визуализации (HMI).

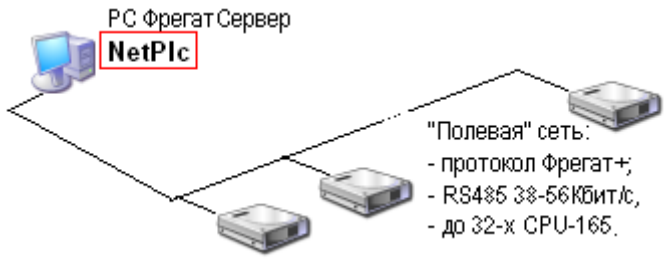


Рис. 1.5.1 Схема сети ПЛК с NetPlc.

Указанные возможности позволяют реализовать на базе PC автоматизированное рабочее место оператора (SCADA) для управления сетью контроллеров Рис. 1.5.1.

NetPlc выполнен на основе архитектуры памяти и логической системы команд CPU-165, что упрощает его использование специалистами знакомыми с CPU-165. Для отладки NetPlc используются те же средства Фрегат2005, что и для CPU-165.

Программа для NetPlc разрабатывается в редакторе Lad. Для выбора типа контроллера добавлена опция «платформа» в настройках проекта.

Логическая группа команд NET PLC , включая таймеры и счетчики, соответствует системе команд CPU-165.

Функциональные команды отличаются от прототипа. В данную группу включены команды обмена с сетью ПЛК, регистрации событий и обработки данных.

2. Поддерживаемые системы программирования

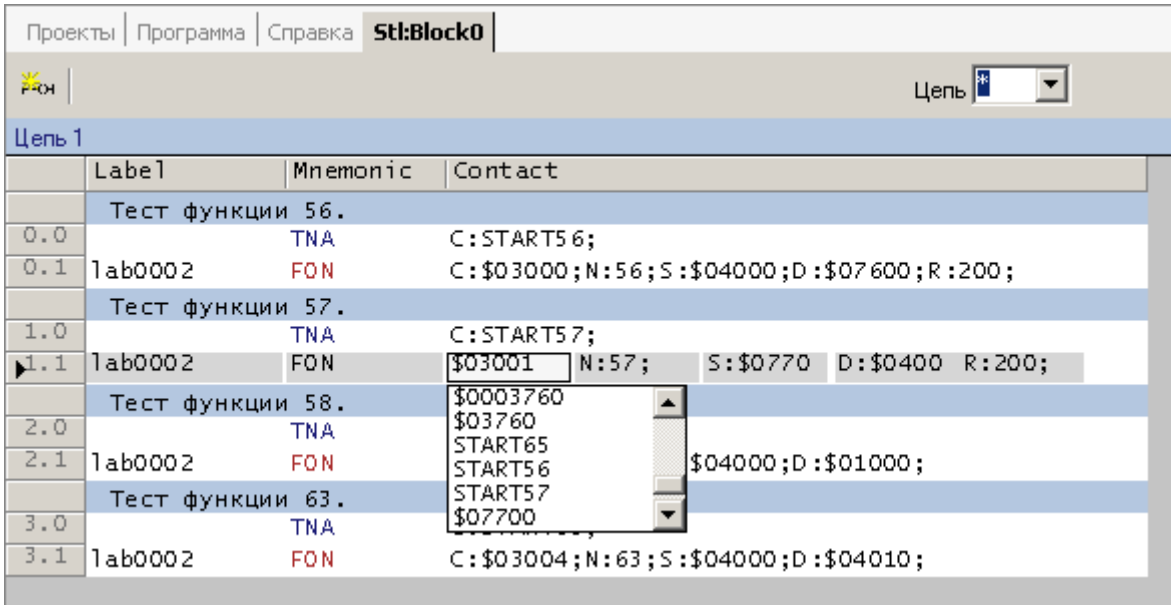
2.1. STL - список инструкций

Термином STL (STatement list) в данном документе обозначается программирование в мнемонических инструкциях ПЛК.

Перечень инструкций приведен в разделе "Программная модель ПЛК"

Каждая инструкция располагается в отдельной строке редактора. Инструкция содержит поля: метка, мнемоника, параметры. Цепь может содержать строку комментария.

Текст мнемоники инструкции окрашивается в соответствии с ее типом.



При вводе параметров ссылающихся на области памяти или ввода-вывода используются символические имена.

Для ввода мнемоники и символических параметров используются выпадающие списки. Численные параметры вводятся в текстовых окнах.

2.2. LAD(РКС) - релейно-контактная схема

Графический язык LAD был предложен для переноса существующих ранее релейно-контактных схем автоматики в ПЛК с минимальными изменениями.

Он позволяет представить объект управления как последовательность цепей.

Программа LAD состоит из одной или более цепей. Цепи вычисляются последовательно, от первой до последней.

Цепь может включать множество (ноль и более) контактов включенных параллельно или последовательно, и одной выходной катушки (таймера, счетчика, функции), см примеры цепей на Рис. 2.2.1, 2.2.2, 2.2.3.

Вычисление логической функции проводится слева направо и сверху вниз.

Левый край цепи всегда принимает значение ИСТИНА (соответствует шине питания в релейной схеме).

Состояние контактов:

- 0 - ЛОЖЬ,
- 1 - ИСТИНА

Способ включения контактов в цепи определяет логическое условие:

- И - последовательное соединение см. Рис. 2.2.1,
- ИЛИ - параллельное соединение см. Рис. 2.2.2,
- смешанное - последовательные и параллельные соединения в одной цепи см. Рис. 2.2.3.

Значения контактов могут быть:

- прямыми - изображаются параллельными линиями (аналог: нормально разомкнутый контакт реле),
- инверсными (НЕ) - параллельные линии с перечеркиванием (аналог: нормально замкнутый контакт реле).

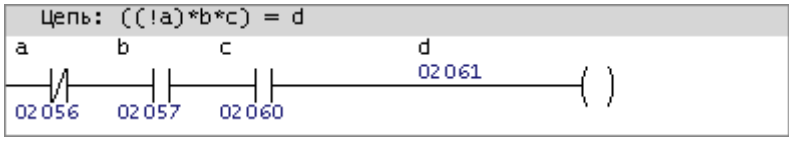


Рис. 2.2.1 Пример цепи с последовательными соединениями контактов.

Логическая функция: (!a)*b*c = d, где:

1. ! инверсия (НЕ),
2. * логическое умножение (И),
3. + логическое сложение (ИЛИ),
4. = присвоение результата.

Процесс выполнения цепи РКС сводится к последовательной проверке логических условий включения контактов и включения или выключения выходной катушки.

В простейшем случае, когда в цепи нет контактов, происходит включение выходной катушки.

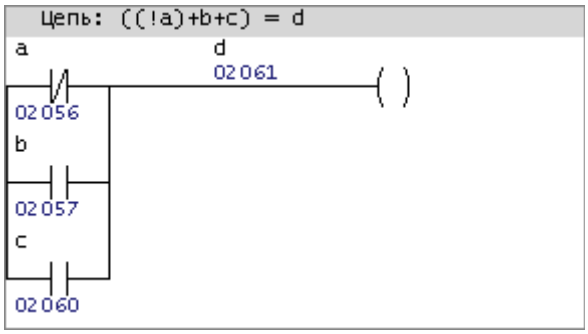


Рис. 2.2.2 LAD пример цепи с параллельными соединениями

Логическая функция: ((!a)+b+c) = d, где:

1. ! инверсия (НЕ),
2. * логическое умножение (И),
3. + логическое сложение (ИЛИ),
4. = присвоение результата.

В качестве контактов принимается любой бит из множества адресов ввода-вывода. При этом:

- контакты внешнего ввода-вывода отражают текущее состояние объектов управления,

- контакты фиктивного ввода-вывода являются результатом анализа предыдущих логических условий.

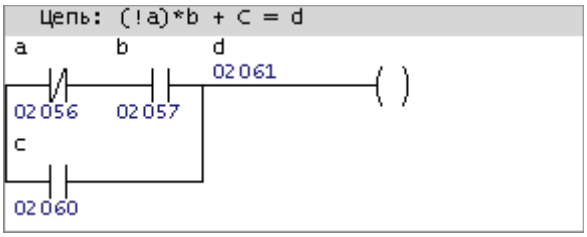


Рис. 2.2.3 LAD пример смешанной цепи

Логическая функция: ((!a)*b+c) = d, где:

1. ! инверсия (НЕ),
2. * логическое умножение (И),
3. + логическое сложение (ИЛИ),
4. = присвоение результата.

2.3. FBD - функциональная схема

Поддержка данной системы программирования будет включена в следующие версии Фрегат2005.

2.4. Программирование HMI/SCADA

Описываемая в данном руководстве подсистема SCADA-HMI находится в стадии разработки и не рекомендуется для самостоятельного применения в ответственных узлах систем управления. По вопросам применения SCADA-HMI обращайтесь к специалистам ООО Автоматика плюс.

2.4.1 Основные понятия

SCADA
- программный комплекс для визуализации и диспетчеризации технологических процессов

HMI
- интерфейс человек-машина, часть системы SCADA. Экранные формы, отображающие результаты работы системы управления и обеспечивающие ввод управляющих сигналов.

Аппаратные значения
- значения измеряемых параметров в представлении ПЛК.

Аппаратные значения могут представлять дискретные сигналы которые представлены в ПЛК одним битом, или аналоговые сигналы, представленные двумя байтами.

Физические значения
- значения, получаемые при обработке аппаратных значений в элементами в системе HMI/Scada.

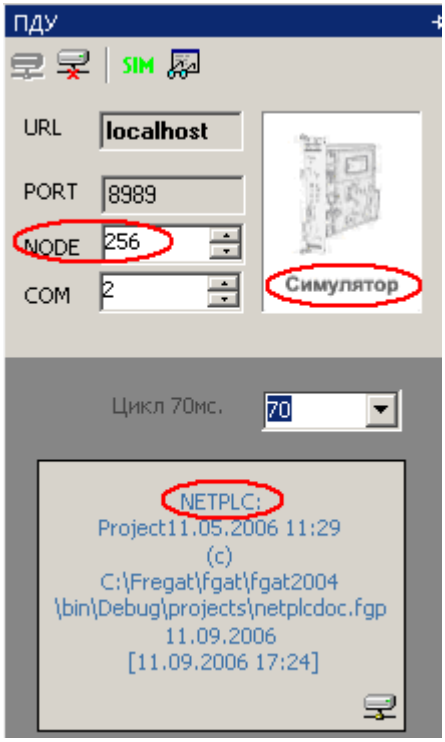


Рис. 2.4.1 Установка связи с NetPlc

1. node = 256,
2. устройство = симулятор,
3. При успешном соединении в окне подключения текст "NetPlc".

Физические значения представляются в соответствии с типом измеряемых величин. Для аналоговых сигналов используется представление в виде числе с плавающей точкой двойной точности. Способ преобразования аппаратных значений в физические и обратно для аналоговых сигналов определяется через минимальное/максимальное значение для обоих значений. Для представления дискретных сигналов используется булевые величины.

Логический контроллер NetPlc - программная реализация контроллера на основе архитектуры памяти и логической системы команд CPU-165, предназначенная для сбора данных с одного или нескольки ПЛК и выполнения программ обработки принятых данных, регистрации данных и событий.

Логическая группа команд NetPlc , включая таймеры и счетчики, соответствует системе команд CPU-165.

Функциональные команды отличаются от прототипа. В данную группу включены команды обмена с сетью ПЛК, обработки и записи событий и данных в регистратор.

Программа для NetPlc разрабатывается в редакторе Lad Фрегат. Для выбора типа контроллера добавлена опция «платформа» в настройках проекта. Для отладки NetPlc используются те же возможности Фрегат2005, что и для CPU-165.

Для установлении связи с NetPlc необходимо задавать параметр NODE больше 255. При этом настройки номера COM порта и тип ПЛК не действует.

При успешном соединении с NetPlc в окне «Модуля связи» отображается сообщение “NetPlc”.

2.4.2 Среда исполнения SCADA задач

Среда исполнения организуется модулем "ФрегатСервер":

1. доставка данных от контроллеров по заданному расписанию;
2. запуск программ обработки данных, выявление и индикация аварий;

3. регистрация данных и сообщений в системе для последующего анализа;
4. визуализация форм HMI в режиме реального времени

Первые три задачи выполняются программно в NetPlc. Для поддержки данного режима работы необходимо включить настройку

- NetPlcLoad="true"

и разместить в каталоге ФрегатСервер файл соответствующей программы управления NetPlc.

Формы HMI обеспечивают обработку и визуализацию данных.

Элементы управления форм HMI проводят вычисление физических значений измеряемых сигналов выбирая их из памяти NetPlc и выводят данные для визуализации на экран или печать. Каждый элемент управления обрабатывает один или несколько цифровых или аналоговых сигналов.

Загрузка и исполнение форм HMI начинает действовать при включении настройки сервера HMILoad="путь к каталогу форм". Вид и содержание работы формы задается при создании форм в дизайнере интерфейса в модуле «ФрегатБаза».

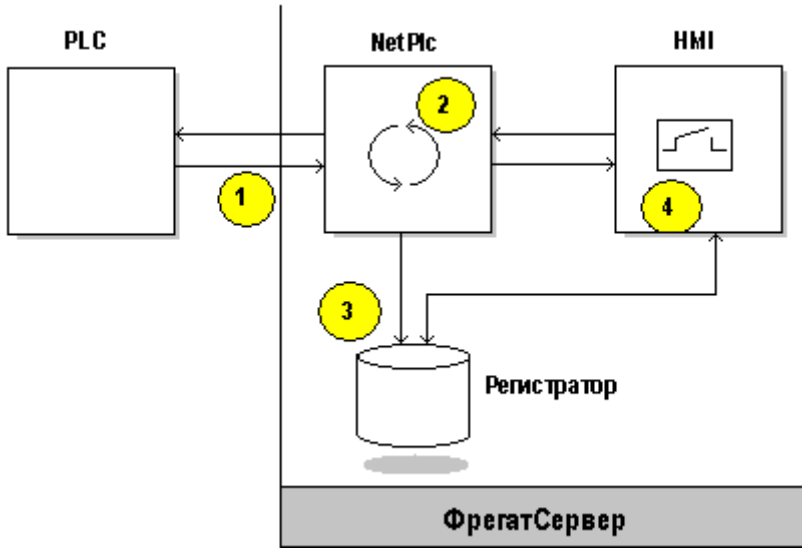


Рис. 2.4.2 Среда исполнения SCADA

1. доставка данных,
2. обработка данных,
3. регистрация данных,
4. визуализация форм HMI.

Доставка данных из контроллеров в память NetPlc выполняется функциональными командами NODE, NWRITE, NREAD. Расписание доставки задается с помощью программирования таймеров, счетчиков или функциональной команды NSYNC.

Обработка аппаратных значений сигналов выполняется командами:

- Сравнения: NLESS, NGREAT, NMAX, NLESSI, NGREATI;
- Поиск максимума: NMAX;
- Усреднение: NAVG.

2.4.3 Элементы управления форм HMI

Элементы HMI используют связанные ячейки в памяти NetPlc. Таким образом, связанные ячейки доступны и программе NetPlc и элементам HMI.

Свойства элементов доступны в окне дизайнера в правой части экрана. Общие свойства для большинства элементов:

- Address, InAddr: адрес связанной ячейки NetPlc состояние которой используется как входной сигнал,
- OutAddr: адрес связанной ячейки NetPlc состояние которой используется как выходной сигнал,
- Enabled: разрешает или запрещает изменение состояния элемента,
- Name: имя элемента. Регистрируется при записи сообщений в журнал,
- SyncType: синхронизация обновления элемента. При 0 в данном поле элемент обновляется на каждом такте работы системы (задается параметром NetSyncTime настроек сервера). Значение, отличное от 0 рассматривается как адрес бита в памяти NET PLC, из которого выполняется синхронизация уровнем 1,
- RegData: регистрировать значение элемента в журнале,
- RegMessages: регистрировать сообщения для элемента.

Ряд свойств доступны при открытии свойства Uobject

- BackColor: цвет фона;
- BackgroundImage: фоновое изображение;
- BorderStyle: тип границы элемента;
- Cursor: тип курсора при перемещении указателя мыши над элементом;
- ForeColor: цвет текста, границ;
- ImageAlign: выравнивание фонового изображения;
- TextAlign: выравнивание текста внутри элемента.

Установка элементов на форме HMI производится из главного меню:

- Правка - Добавить элемент -...

Синхронизация - автогенератор

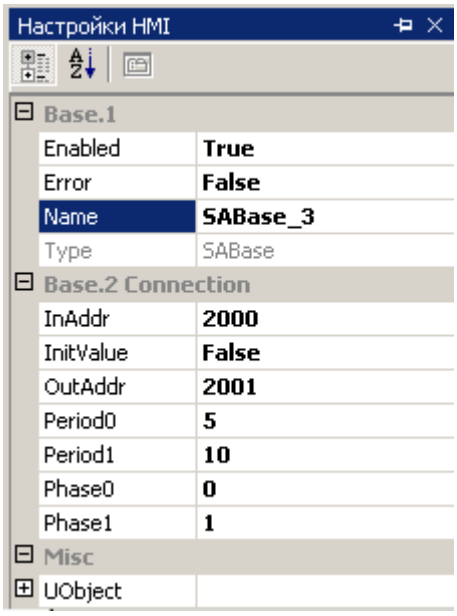


Рис. 2.4.4 Элемент Синхронизация - автогенератор

Вырабатывает серию сигналов синхронизации длительностью в один отсчет с управляемым периодом и фазой. Два возможных значения для пары (период, фаза) задаются свойствами:

- (Period0,Phase0) ,
- (Period1,Phase1).

Входное значение типа «бит» берется из адреса InAddr и управляет установкой пары значений (Period0,Phase0) или (Period1,Phase1).

Выходное значение типа «бит» помещается в память NetPlc с адресом заданным в свойстве OutAddr.

При выполнении элемента выходное значение 1 формируется через каждые (Period) тактов на (Phase) такте. В остальные такты выходное значение равно 0. Для подсчета используется счетчик сканов NetPlc, период сканов задан в настройках сервера. Временные диаграммы работы автогенератора приведены на Рис 2.4.3.

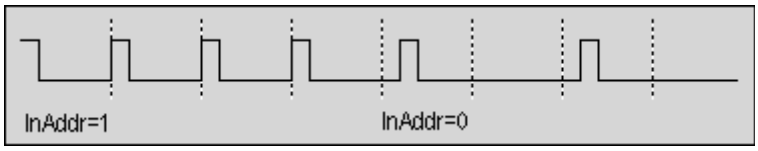


Рис. 2.4.3 Элемент Синхронизация - автогенератор

В данном примере использовались значения пар (период, фаза): (5,0) и (10,1). Входной сигнал для элемента - бит NetPlc по адресу 02000_8, управляет выбором периода и фазы. Выходное значение помещается в память NetPlc по адресу 02001_8.

Графики формируются элементами «График».

Элемент: Синхронизация - таймер

Использует входное значение Ainput для разрешения работы. В остальном действует аналогично элементу "Автогенератор".

Элемент: Дискретный - базовый

Основные свойства:

- Enabled: способность изменять данные в связанной ячейке при нажатии на область элемента; может быть в состоянии:
  - запрещено (false) - элемент только отражает изменение связанной ячейки
  - разрешено (true) - элемент может записывать в ячейку.
- ImageFalse, ImageTrue: индексы изображения при значении в связанной ячейке 0 и 1 соответственно;
- TextFalse,TextTrue: то же для отображаемого текста.

Элемент: Дискретный - цветная метка

Дополняет свойства "базового дискретного" элемента цветами фона и текста в состояниях 0 и 1.

Элемент: Дискретный - управляемое разрешение

Дополняет свойства элемента "цветная метка" битом разрешения индикации из указанной ячейки NetPlc.

Элемент: Дискретный - анимация

Дополняет свойства элемента "управляемое разрешение" индексами изображений. При значении 1 в связанной ячейке индекс изображения меняется в заданных пределах, обеспечивая анимационное изображение.

Элемент: Аналоговый - базовый

Основные свойства

- Device: тип преобразователя аппаратного значения в физическое;
  - AppMin, AppMax: диапазон изменения аппаратного значения;

- PhyMin, PhyMax: диапазон изменения физической величины;
- Name: имя преобразователя; данное поле позволяет выбрать один из заданных типов преобразования из таблицы "Devices";
- KeepMaxValue: указывает элементу сохранять максимум;
- Precision: количество знаков после запятой при выводе физической величины;
- Units: наименование единиц измерения физической величины.

- Percent scale: процентная шкала значений.
  - PhyLow, PhyHigh: минимальное или максимальное значение физической величины;
  - HighMessage, LowMessage, NormalMessage: Текстовые сообщения при нормальном, пониженном или повышенном уровнях физической величины.
- Appearance: вид представления
  - цифровой;
  - горизонтальная шкала;
  - круговая шкала;
  - полукруговая шкала;
  - график.

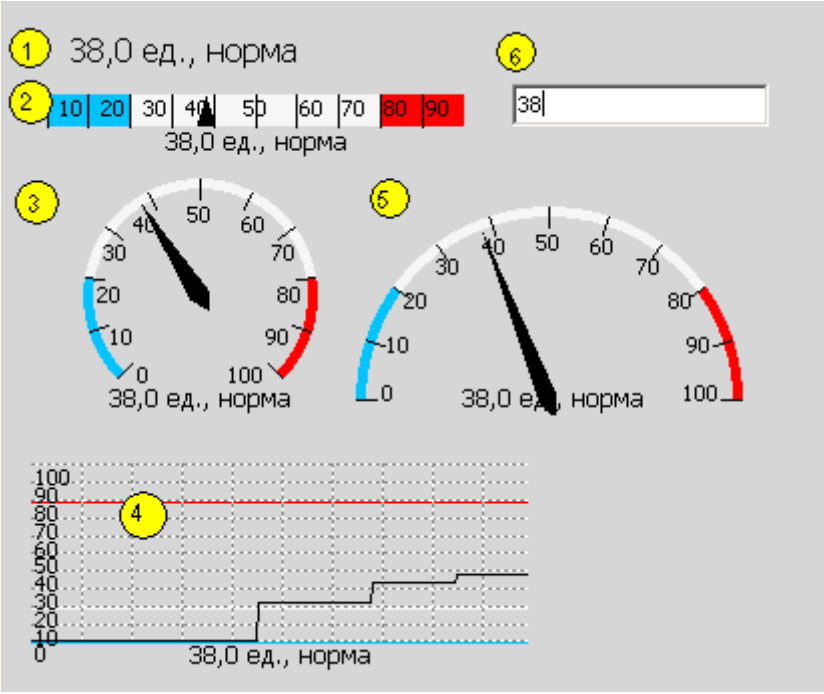


Рис. 2.4.5 Аналоговые элементы

1. цифровой,
2. горизонтальная шкала,
3. круговая шкала,
4. график
5. полукруговая шкала,
6. аналоговый выход.

Элемент: Аналоговый выход

Позволяет вводить значения физических величин в память NetPlc с преобразованием в аппаратную.

Элемент: Статический - картинка

Статическое изображение задается свойством Uobject-Image

Элемент: Статический - текст

Задает текстовое статическое сообщение. Параметры шрифта задаются в свойстве UObject-Font

Элемент: График

Элемент график накапливает и выводит на экран состояния битовой переменной из областей памяти или ввода-вывода NetPlc.

Свойства:

- Address: адрес связанной переменной:
  - адреса ввода-вывода обозначаются обычным способом, как на схемах РКС:
  - к адресам байтовой области добавляется младшая цифра, обозначающая бит, младший бит 0, старший 7.
- Colors: цвет графика;
- Elements: число сохраненных значений, число точек графика ;
- LabelInt: интервал меток;
- LineWidth: толщина линий графика;

Элемент: Журнал сообщений

Журнал сообщений в режиме просмотра данных обеспечивает выбор сохраненных файлов за указанную дату, и необходимую фильтрацию для отображения графиков и журналов событий по группам.

Элемент: Дата и время

Обеспечивает индикацию текущего времени в формате ЧЧ.ММ.СС и интервала времени с момента включения.

Регистратор

Регистратор сообщений и данных обеспечивает запись данных и событий в файл.

Данные в регистратор записываются:

- командами NetPlc: NMESSAGE, NSTORE.

Реализован режим накопления данных в течение суток, после чего данные записываются в файл с именем «DD_MM_YYYY.datatable», где:

- DD - код дня (1-31);
- MM - код месяца (1-12);
- YYYY - код года (например, 2005);

3. Функции по группам

3.1. Пересылки данных

Код функции	Мнемоника	Выходы	S	D	R	Действие
0	**MOV**		M	M		Пересылка слова
1	**MVIBCD**		B	M		Занесение BCD константы в память
2	**XCG**		M	M		Обмен слов
7	**MVID**		D	M		Занесение десятичной константы в память
8	**MVIO**		O	M		Занесение восьмеричной константы в память
68	**PUSH**		M	M		Ввод в стек
69	**POP**		M	M	M	Вывод из стека
70	**MOVBLK**		M	M	M	Пересылка блока
71	**MVMBLK**		M	M	M	Косвенная пересылка блока
74	**STM**		M	M		Запись слова с косвенной адресацией
75	**LDM**		M	M		Чтение слова с косвенной адресацией
76	**MVM**		M	M		Пересылка слова с косвенной адресацией
90	**MOVB**		M	M		Пересылка байта
91	**SHFTBLK**		M	M		Сдвиг блока по байту

3.2. Преобразования

Код функции	Мнемоника	Выходы	S	D	R	Действие
3	**BCDW**		M	M		Преобразование слова из BCD
4	**WBCD**		M	M		Преобразование слова в BCD
50	**ENCOD**		M	M		Кодирование в позиционный код
51	**DECOD**	Z	M	M		Декодирование позиционного кода
53	**SWPB**		M	M		Обмен тетрад в байте

3.3. Логические

Код функции	Мнемоника	Выходы	S	D	R	Действие

9	**NEG**	Z	M	M		Инверсия байта
13	**AND**	Z	M	M	M	Логическое умножение байта
14	**OR**	Z	M	M	M	Логическое сложение байта
18	**XOR**	Z	M	M	M	Исключающее "ИЛИ" с байтом
38	**BMASK**		M	O	O	Байтовая маска
39	**WMASK**		M	O	O	Двухбайтовая маска
81	**PRTO**		M	M		Контрольный бит четность
82	**PRTOCH**	=	M			Проверка на четность
83	**PRTE**		M	M		Контрольный бит нечетность
84	**PRTECH**	=	M			Проверка на нечетность

3.4. Арифметические

Код функции	Мнемоника	Выходы	S	D	R	Действие
10	**ADDBCD**	>Z	M	M	M	Сложение слов BCD
11	**SUBBCD**	-Z	M	M	M	Вычитание BCD
12	**CMP**	>=<	M	M		Сравнение слов
36	**ROR**	F08	M			Сдвиг слова вправо
37	**ROL**	0F7	M			Сдвиг слова влево
54	**SCALE**		M	M	O	Масштабирование
55	**MATH**	<=>	M	M	D	Специальные математические функции
92	**ADD**	>Z	M	M	M	Сложение
93	**SUB**	<Z	M	M	M	Вычитание
94	**MUL**		M	M	M	Умножение
95	**DIV**		M	M	M	Деление

3.5. Таймеры,счетчики

Код функции	Мнемоника	Выходы	S	D	R	Действие
56	**TIMSET**		M	M	O	Уставки в формате времени
57	**TIMFIX**		M	M	O	Преобразование уставки во время
63	**INCW**	>=	M	M		Инкремент слова

64	**DECW**	<Z	M			Декремент слова
65	**TGRP**		O			Группа таймеров
66	**TGRP2**		M	M	O	Группа таймеров 2

3.6. Специальные

Код функции	Мнемоника	Выходы	S	D	R	Действие
19	**WRITE**		M	O	O	Запись блока в модуль ввода-вывода
20	**READ**		O	M	M	Чтение блока из модуля ввода-вывода
21	**MVAMOD**	E	M	O	O	Задать режимы МВА-08
22	**MVASET**	E	M	O		Запись уставок МВА-08
23	**MVARD**		O	M		Чтение данных МВА-08
41	**PID**		O	O		PID-регулятор
58	**ERCODE**		M	M		Индикатор кода аварии
98	**ADCWR**		O	O	O	Запись уставок.
99	**ADCRD**		M	O	O	Считывание кодов АЦП

3.7. NETPLC, SCADA

Код функции	Мнемоника	Выходы	S	D	R	Действие
300	**NODE**		D			Установить сетевой адрес
301	**NREAD**	O	O	M	D	Чтение данных удаленного ПЛК
302	**NWRITE**	O	M	O	D	Запись в удаленный ПЛК
303	**NMSG**		G	S		Запись сообщения
304	**NSTORE**		G	S	M	Запись в хранилище
305	**NSYNC**	=	D	D		Автогенератор
306	**NGREAT**	>	M	M		Больше
307	**NLESS**	<	M	M		Меньше
308	**NEQUAL**	=	M	M		Равно
309	**NMEM**		M			Тестовая команда
310	**NAVG**	R	M	M		Усреднитель
311	**NSETDI**		M	D		Задать слово

312	**NMOVB**		M	M	D	Пересылка байт
313	**NMOVW**		M	M	D	Пересылка слов
314	**NMAX**	R	M	M		Максимум
315	**NLESSI**	<	M	D		Меньше значения
316	**NGREATI**	>	M	D		Больше числа

4. Детальное описание функций

[0] MOV Пересылка слова

Параметры

S	Адрес пересылаемого слова
D	Адрес в который пересылаются данные
R	

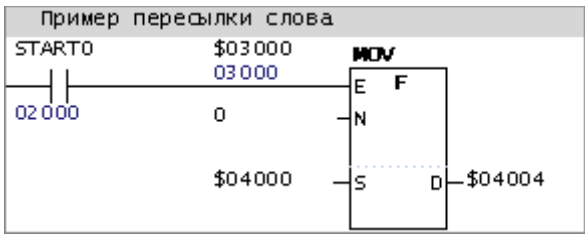
Выходы
Описание

Передается слово в памяти данных из адреса (S) в адрес (D). После выполнения команды данные по адресу (S) сохраняются.

Пример :

Команда выполняется при значении 1 бита START0 (адрес 02000**8**). Слово 9876**16** находящееся по адресу 04000**8** передаётся в адрес 04004**8**.

Цепь



До выполнения

После выполнения

[1] MVIBCD Занесение BCD константы в память

Параметры

S	Константа в двоично-десятичной кодировке
D	Адрес слова, в который заносятся данные
R	

Выходы
Описание

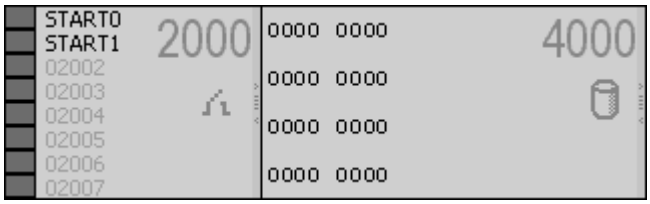
Константа в двоично-десятичной кодировке заданная в параметре S, заносятся в память данных по адресу в D.
Константа содержит до четырех цифр.

Пример :

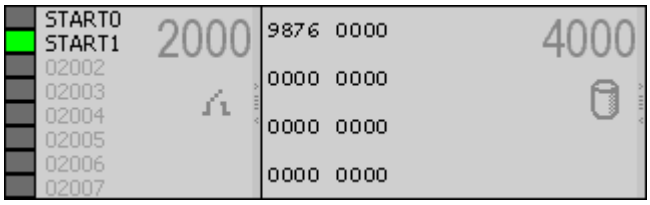
Команда выполняется при значении 1 бита START1 (адрес 02001**8**). Слово 9876**16** заданное в параметре S заносится в адрес 04000**8**.

Цепь

До выполнения



После выполнения



[2] XCG Обмен слов

Параметры

S	Адрес обмениваемых данных
D	Адрес обмениваемых данных
R	

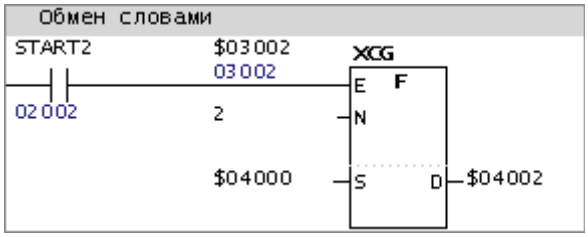
Выходы
Описание

Обмен словами в памяти данных по адресам в (S) и (D).

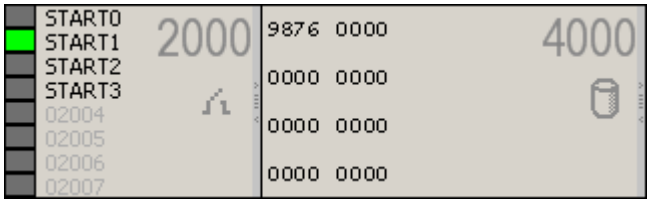
Пример :

Команда выполняется при значении 1 бита START2 (адрес 02002**8**). Извлекаются слова по адресам (S) 04000**8** = 9876**16** и (D) 04002**8** = 0000**16**, затем сохраняются по адресам 04002**8** и 04000**8** соответственно.

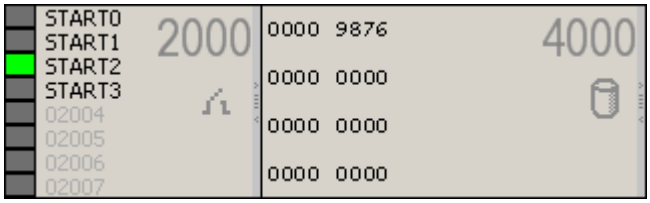
Цепь



До выполнения



После выполнения



[3] BCDW Преобразование слова из BCD

Параметры

S	Адрес исходных данных
D	Адрес результата преобразования
R	

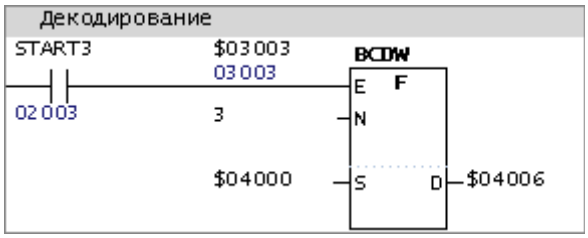
Выходы
Описание

Слово по адресу (S) в двоично-десятичной системе (BCD) преобразуется в двоичный код и записывается в память по адресу (D).

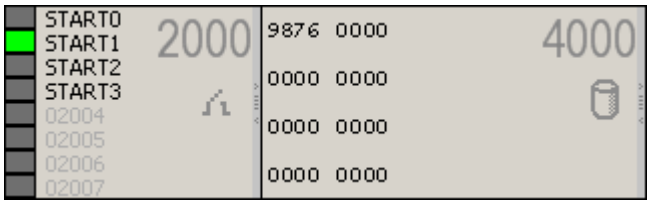
Пример :

Команда выполняется при значении 1 бита START3 (адрес 02003**8**). Извлекается слово по адресу (S) 04000**8** = 9876**16**, преобразуется в двоичный код 2694**16**. Результат сохраняется по адресу (D) 04006**8**.

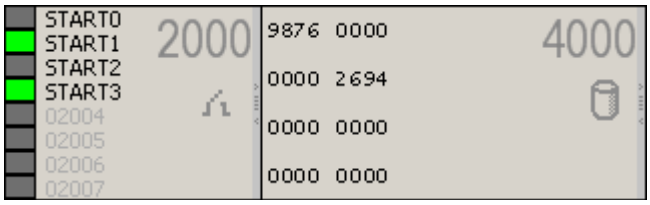
Цепь



До выполнения



После выполнения



[4] WBCD Преобразование слова в BCD

Параметры

S	Адрес преобразуемого слова
D	Адрес результата преобразования
R	

Выходы
Описание

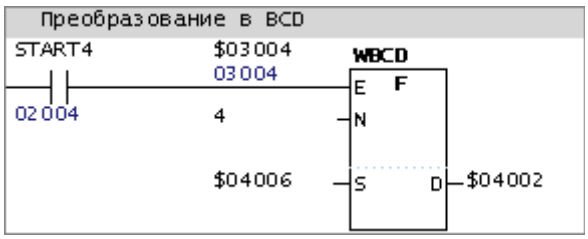
Преобразование слова по адресу (S) в доичном коде в четырехразрядные по BCD. Результат заносится в память по (D).

Примечание:
Если слово в двоичнм коде подлежащее преобразованию превышают 9999**10**, то при выполнении команды включается ввод-вывод флажка ошибки FErr1 (адрес 3771**8**) и преобразование не выполняется.

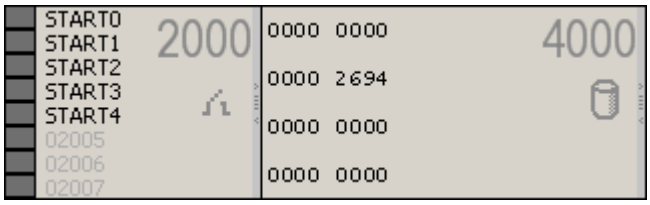
Пример :

Команда выполняется при значении 1 бита START4 (адрес 02004**8**). Извлекается слово по адресу (S) 04006**8** = 2694**16**, преобразуется в двоичный код 9876**16**. Результат сохраняется по адресу (D) 04002**8**.

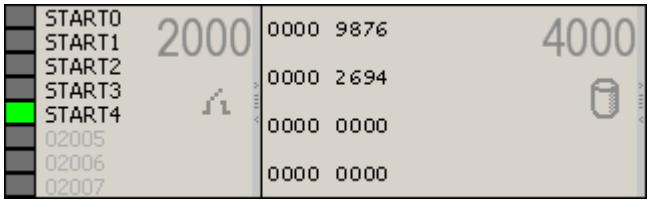
Цепь



До выполнения



После выполнения



[7] MVID Занесение десятичной константы в память

Параметры

S	Константа в десятичном коде
D	Адрес слова, в который заносятся данные
R	

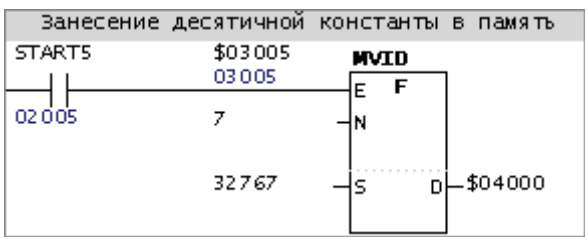
Выходы
Описание

Константа в десятичной кодировке заданная в параметре S, заносятся в память данных по адресу в D. Константа может быть числом в диапазоне от 0 до 65535**10**.

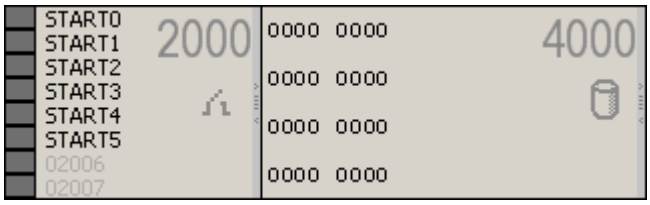
Пример :

Команда выполняется при значении 1 бита START5 (адрес 02005**8**). Слово 32767**10**, заданное в параметре S, заносится в адрес заданный в D=04000**8**.

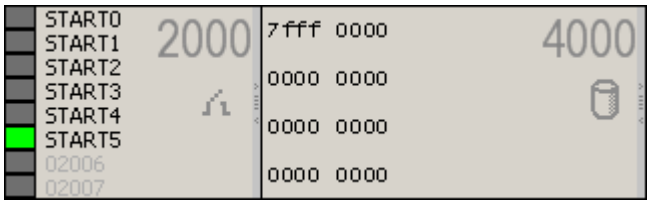
Цепь



До выполнения



После выполнения



[8] MVIO Занесение восьмеричной константы в память

Параметры

S	Константа в восьмеричном коде
D	Адрес слова, в который заносятся данные
R	

Выходы
Описание

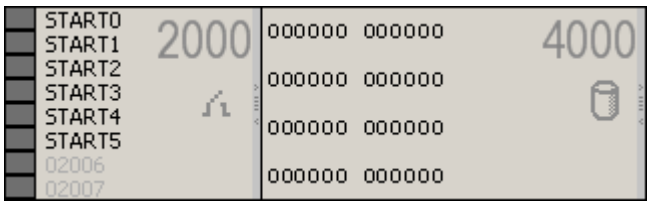
Константа в восьмеричной кодировке заданная в параметре S, заносятся в память данных по адресу в D.
Константа может быть числом в диапазоне от 0 до 177776**8**.

Пример :

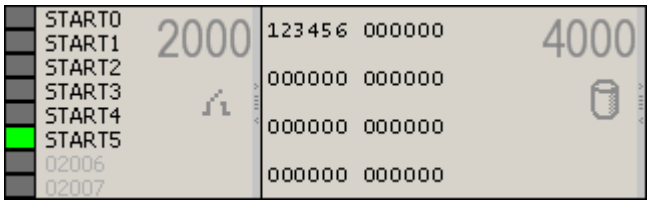
Команда выполняется при значении 1 бита START5 (адрес 02005**8**). Слово 123456**8**, заданное в параметре S, заносится в адрес заданный в D=04000**8**.

Цепь

До выполнения



После выполнения



[9] NEG Инверсия байта

Параметры

S	Адрес байта для инверсии
D	Адрес результата
R	

Выходы

Обозначение	Адрес	Условие
Z (нуль)	$+1	Устанавливается в "1" , если результат инверсии равен 0.

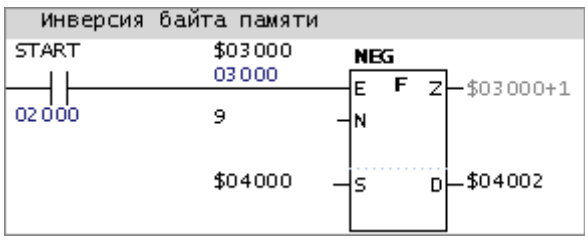
Описание

Инвертируются биты одного байта данных из (S), результат инверсии передается в (D). В зависимости от результата выполнения команды устанавливается флажок нуля.

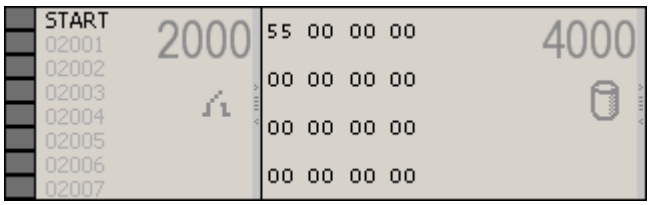
Пример :

Команда выполняется при значении 1 бита START (адрес 02000**8**). Байт 55**16** извлекается по адресу в параметре S = 04000 и инвертируется. Байт результата AA**16** заносится в адрес заданный в D=04002**8**.

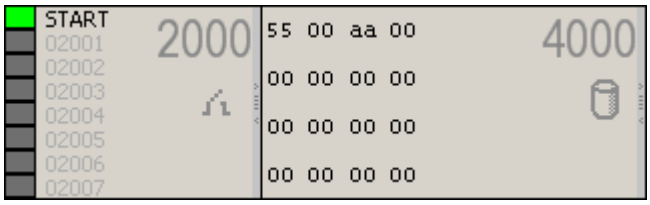
Цепь



До выполнения



После выполнения



[10] ADDBCD Сложение слов BCD

Параметры

S	Адрес первого слагаемого
D	Адрес второго слагаемого
R	Адрес результата сложения

Выходы

Обозначение	Адрес	Условие
> (перенос)	$+1	Устанавливается в "1" , если результат сложения превысит 9999**BCD**, иначе "0"..
Z (нуль)	$+2	Устанавливается в "1" , если результат равен 0.

Описание

Сложение четырехзначных слов в двоично-десятичном коде по адресам (S) и (D). Результат заносится в память по адресу (R).

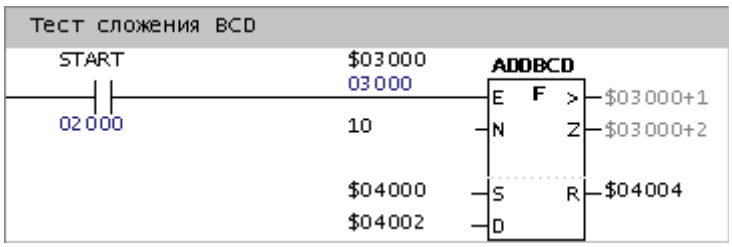
Ошибки

Имя	Адрес	Условие
FErr1	03771**8**	Если слагаемые не являются двоично-десятичными числами, функция не выполняется.

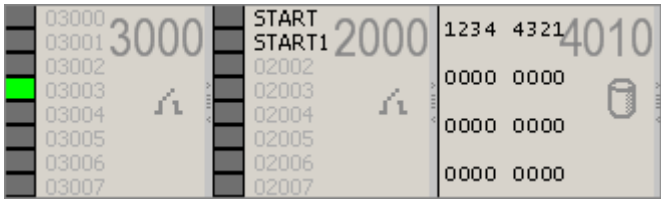
Пример 1:

Команда выполняется при значении 1 бита START (адрес 02000**8**).
Первое слагаемое 1234**BCD** извлекается по адресу S = 4000**8**.
Второе слагаемое 4321**BCD** извлекается по адресу D = 4002**8**.
Результат сложения 5555**BCD** заносится по адресу R = 4004**8**.
Выходы переноса и нуля сброшены.

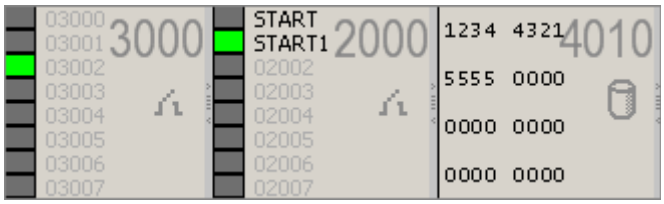
Цепь



До выполнения



После выполнения

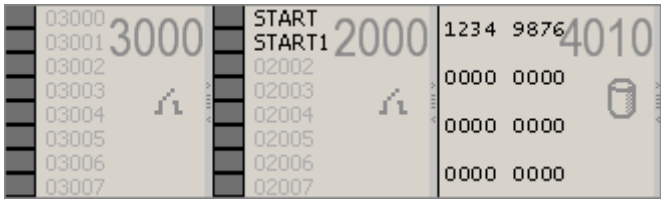


Пример 2:

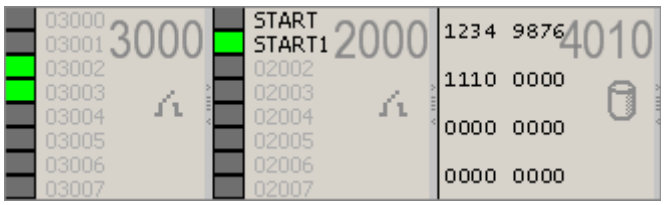
Первое слагаемое 1234**BCD** извлекается по адресу S = 4010**8**.
Второе слагаемое 9876**BCD** извлекается по адресу D = 4012**8**.
Результат сложения 11110**BCD** превышает 9999**BCD** , младшие 4 цифры заносится по адресу R = 4014**8**.
Выход переноса 03003**8** установлен в 1.

Цепь

До выполнения



После выполнения



[11] SUBBCD Вычитание BCD

Параметры

S	Адрес уменьшаемого
D	Адрес вычитаемого
R	Адрес результата

Выходы

Обозначение	Адрес	Условие
- (заем)	$+1	Устанавливается в "1" , если результат вычитания получен с отрицательным знаком, иначе "0"..
Z (нуль)	$+2	Устанавливается в "1" , если результат равен 0.

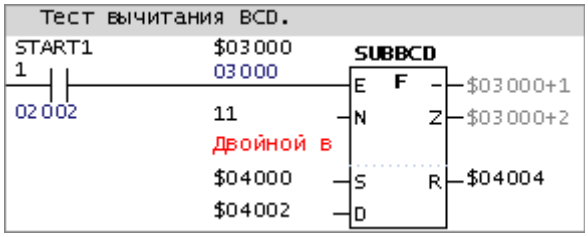
Описание

Выполняется вычитание четырехзначных слов в двоично-десятичном коде по адресам (S) и (D). Результат заносится в память по адресу (R).

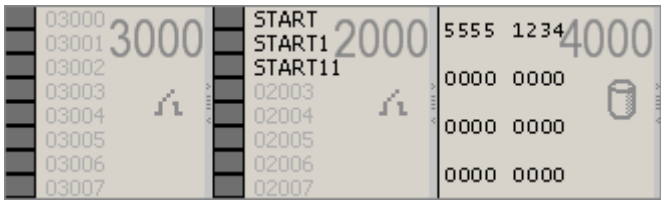
Пример :

Команда выполняется при значении 1 бита START11 (адрес 02002**8**). Уменьшаемое 5555**16** извлекается по адресу S = 4000**8**. Вычитаемое 1234**16** извлекается по адресу D = 4002**8**. Результат вычитания 4321**16** заносится по адресу R = 4004**8**. Выходы переноса и нуля сброшены.

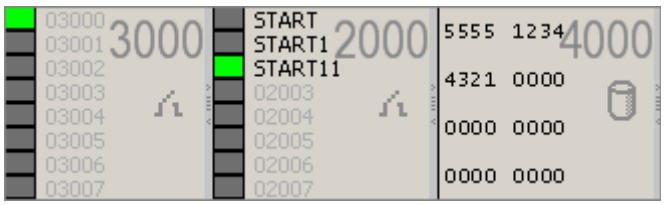
Цепь



До выполнения



После выполнения



[12] CMP Сравнение слов

Параметры

S	Адрес сравниваемого слова
D	Адрес сравниваемого слова
R	

Выходы

Обозначение	Адрес	Условие
S > D (заем)	$+1	Устанавливается в "1" , если слово в (S) больше чем в (D).
= (нуль)	$+2	Устанавливается в "1" , если слова в (S) и в (D) равны.
S < D (нуль)	$+3	Устанавливается в "1" , если слово в (S) льше чем в (D).

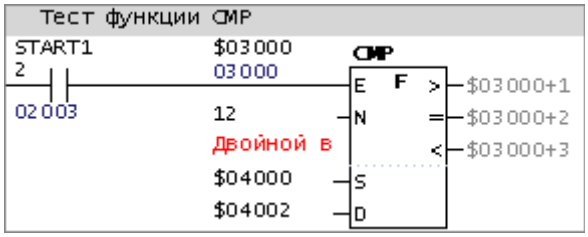
Описание

Вычитание четырехзначных слов в двоично-десятичном коде по адресам (S) и (D). Результат сравнения заносится в биты результата.

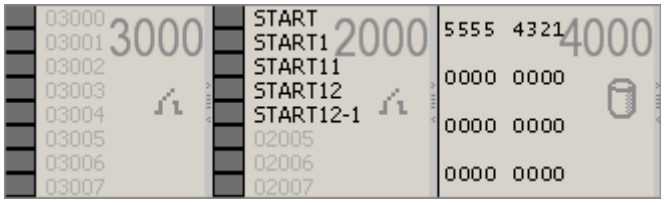
Пример 1:

Команда выполняется при значении 1 бита START12 (адрес 02003**8**). Сравниваемые слова 5555**16** извлекается по адресу S = 4000**8** и 4321**16** извлекается по адресу D = 4002**8**.
Бит результата "меньше" с адресом 3001**8** принимает значение 1. Биты результата "равно" с адресом 3002**8**и "больше" с адресом 3003**8** сброшены в 0.

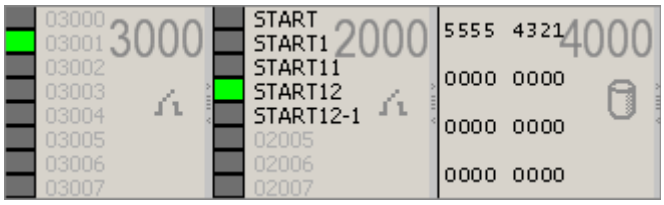
Цепь



До выполнения



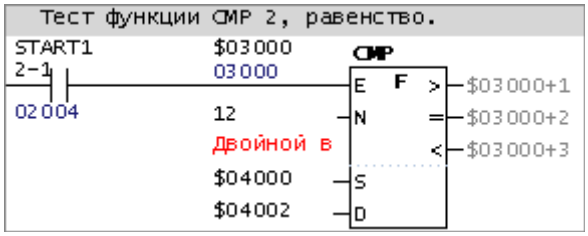
После выполнения



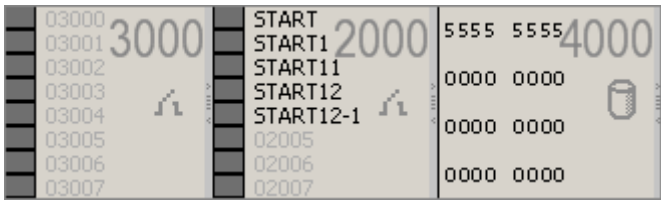
Пример 2:

Команда выполняется при значении 1 бита START12-1 (адрес 02004**8**). Сравниваемые слова 5555**16** извлекается по адресу S = 4000**8** и 5555**16** извлекается по адресу D = 4002**8**.
Бит результата "равно" с адресом 3002**8** принимает значение 1. Биты результата "меньше" с адресом 3001**8** и "больше" с адресом 3003**8** сброшены в 0.

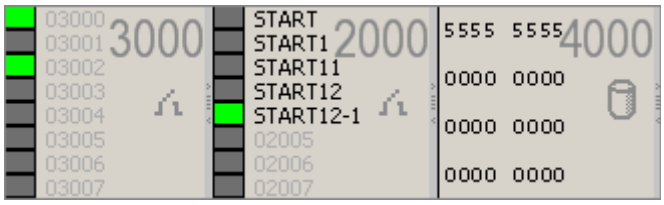
Цепь



До выполнения



После выполнения



[13] AND Логическое умножение байта

Параметры

S	Адрес байта множимого
D	Адрес байта множителя
R	Адрес байта результата

Выходы

Обозначение	Адрес	Условие

Z (нуль)	$+1	Устанавливается в "1" , если результат равен 0.

Описание

Логическое умножение однобайтных величин по адресам (S) и (D). Результат заносится в память по адресу (R). Флаг "нуль" устанавливается при нулевом результате.

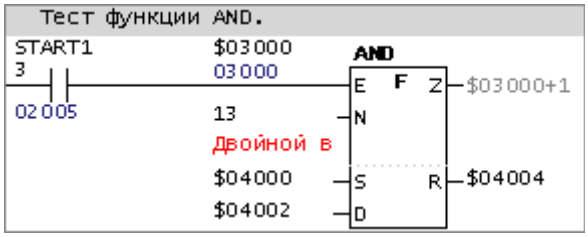
Таблица логического умножения

Бит множимого	1 1 0 0
Бит множителя	1 0 1 0
Бит результата	1 0 0 0

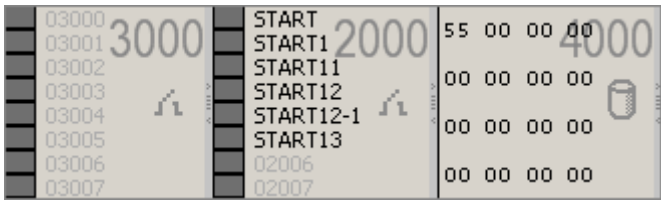
Пример :

Команда выполняется при значении 1 бита START13 (адрес 02005**8**). Байт множимого 55**16** извлекается по адресу S = 4000**8**, байт множителя 0f**16** извлекается по адресу D = 4002**8**. Результат логического умножения 05**16** заносится по адресу R = 4004**8**. Бит результата "нуль" 3001**8** сброшен.

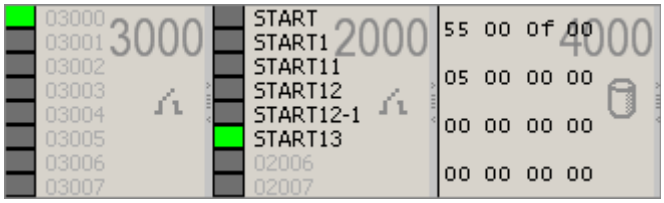
Цепь



До выполнения



После выполнения



[14] OR Логическое сложение байта

Параметры

S	Адрес байта слагаемого

D	Адрес байта слагаемого
R	Адрес байта результата

Выходы

Обозначение	Адрес	Условие
Z (нуль)	$+1	Устанавливается в "1" , если результат равен 0.

Описание

Логическое сложение однобайтных величин по адресам (S) и (D). Результат заносится в память по адресу (R). Флаг "нуль" устанавливается при нулевом результате.

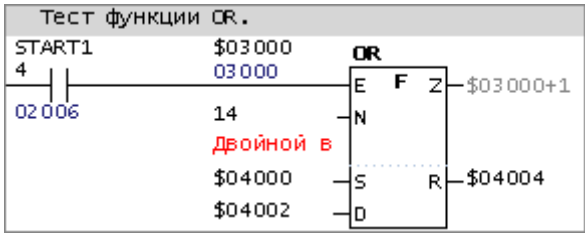
Таблица логического сложения

Бит множимого	1 1 0 0
Бит множителя	1 0 1 0
Бит результата	1 1 1 0

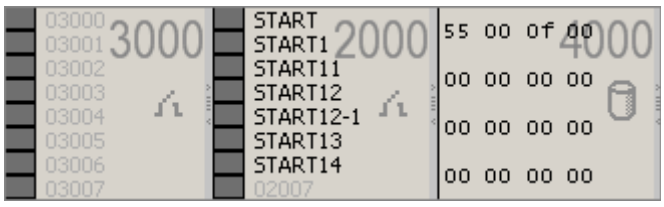
Пример :

Команда выполняется при значении 1 бита START14 (адрес 02006**8**). Байт первого слагаемого 55**16** извлекается по адресу S = 4000**8**, байт второго слагаемого 0f**16** извлекается по адресу D = 4002**8**. Результат логического сложения 5f**16** заносится по адресу R = 4004**8**. Бит результата "нуль" 3001**8** сброшен.

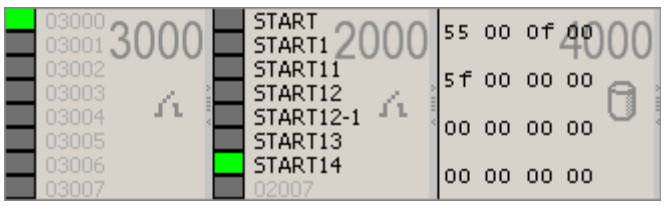
Цепь



До выполнения



После выполнения



[18] XOR Исключающее "ИЛИ" с байтом

Параметры

S	Адрес байта
D	Адрес байта
R	Адрес байта результата

Выходы

Обозначение	Адрес	Условие
Z (нуль)	$+1	Устанавливается в "1" , если результат равен 0.

Описание

Исключающее ИЛИ однобайтных величин по адресам (S) и (D). Результат заносится в память по адресу (R). Флаг "нуль" устанавливается при нулевом результате.

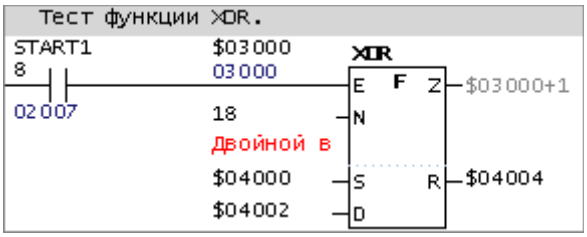
Таблица исключающего ИЛИ

Бит множимого	1 1 0 0
Бит множителя	1 0 1 0
Бит результата	0 1 1 0

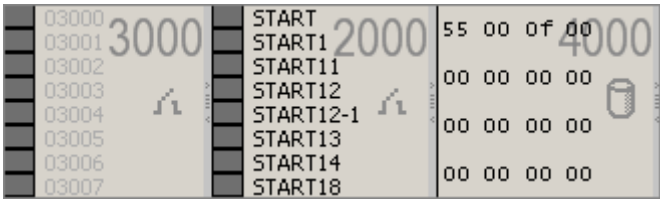
Пример :

Команда выполняется при значении 1 бита START18(адрес 02007**8**). Байт первого слагаемого 55**16** извлекается по адресу S = 4000**8**, байт второго слагаемого 0f**16** извлекается по адресу D = 4002**8**. Результат 5a**16** заносится по адресу R = 4004**8**. Бит результата "нуль" 3001**8** сброшен.

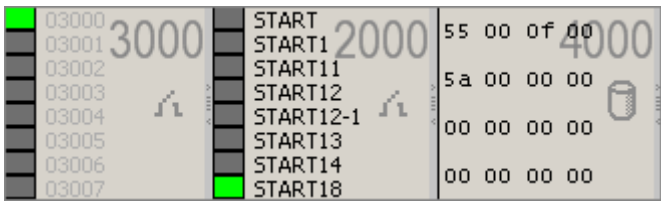
Цепь



До выполнения



После выполнения



[19] WRITE Запись блока в модуль ввода-вывода

Параметры

S	Адрес блока данных
D	Адрес модуля ввода-вывода
R	Адрес слова, содержащего код передаваемого блока (старший байт) и счетчик блоков (младший байт)

Выходы
Описание

Запись блока данных в модули ввода-вывода МВА-02, -03. Данные пересылаются из заданной области памяти данных в модуль, где происходит контроль данных блока.
Содержимое блока данных сохраняется.
Длина блока не должна превышать 256 байт. Код передаваемого блока не должен превышать 376**8**.
Примечание
Подробное описание форматов параметров смотреть в описании модуля ввода-вывода.

Ошибки

Имя	Адрес	Условие
FErr1	03771**8**	Если длина или код передаваемого блока превышают максимальные значения, пересылка прерывается. В ячейки памяти данных 17766**8**, 17767**8** записывается адрес первого слова команды.
FErr2	03712**8**	Ошибка передачи данных, пересылка прерывается. В ячейки памяти данных 17766**8**, 17767**8** записывается адрес первого слова команды.

[20] READ Чтение блока из модуля ввода-вывода

Параметры

S	Адрес модуля ввода-вывода
D	Адрес первого байта считываемого блока в ПЛК
R	Адрес последнего байта считываемого блока в ПЛК

Выходы
Описание

Чтение блока данных из модулей ввода-вывода МВА-02, -03. Данные пересылаются из модуля в заданную область памяти данных ПЛК. Длина блока не должна превышать 256 байт.
Примечание
Подробное описание форматов параметров смотреть в описании модуля ввода-вывода.

Ошибки

Имя	Адрес	Условие
FErr1	03771**8**	Если длина блока превышает максимальное значение, пересылка прерывается. В ячейки памяти данных 17766**8**, 17767**8** записывается адрес первого слова команды.

[21] MVAMOD Задать режимы МВА-08

Параметры

S	Адрес блока данных ПЛК с информацией о режимах работы модуля
D	Адрес модуля ввода-вывода
R	Адрес параметра фильтрации АЦП

Выходы

Обозначение	Адрес	Условие
E (ошибка)	$+1	Устанавливается в "1" , при сбое в обмене информацией между ПЛК модулем

Описание

Запись данных в модуль МВА-08 при калибровке или изменении режима работы.
Примечание
Подробное описание форматов параметров смотреть в описании модуля ввода-вывода.

[22] MVASET Запись уставок МВА-08

Параметры

S	Адрес начала блока уставок в памяти ПЛК
D	Адрес модуля ввода-вывода
R	

Выходы

Обозначение	Адрес	Условие
E (ошибка)	$+1	Устанавливается в "1" , при сбое в обмене информацией между ПЛК модулем

Описание

Запись уставок в модуль ввода-вывода МВА-08.
Примечание
Подробное описание форматов параметров смотреть в описании модуля ввода-вывода.

[23] MVARD Чтение данных МВА-08

Параметры

S	Адрес модуля ввода-вывода

D	Адрес начала блока результатов в памяти ПЛК
R	

Выходы

Обозначение	Адрес	Условие
E (ошибка)	$+1	Устанавливается в "1" , при сбое в обмене информацией между ПЛК модулем

Описание

Пересылка результатов преобразования АЦП из модуля МВА-08 в память данных ПЛК.
Примечание
Подробное описание форматов параметров смотреть в описании модуля ввода-вывода.

[36] ROR Сдвиг слова вправо

Параметры

S	Адрес слова
D	
R	

Выходы

Обозначение	Адрес	Условие
F (15-й бит)	$+1	запишется в старший бит слова
0 (0-й бит)	$+2	Содержимое 0 бита до сдвига
8 (8-й бит)	$+3	Содержимое 8 бита до сдвига

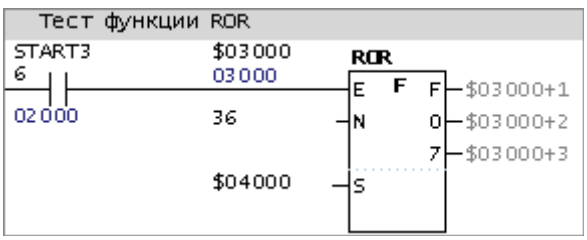
Описание

Сдвиг вправо на один бит слова данных, записанных по адресу (S). Бит **$+1** записывается в старший бит слова (S). 8-й бит слова (S) записывается по адресу **$+2**. 1-й бит слова записывается по адресу **$+3**.

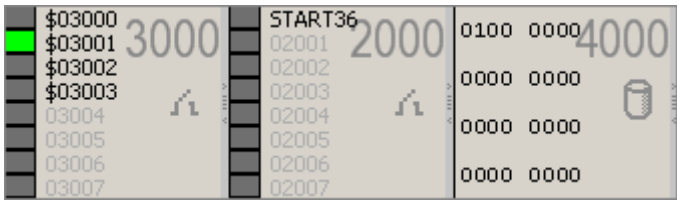
Пример :

Перед выполнением слово 0100**16** записано по адресу S = 4000**8**. Бит "1" записанный по адресу **$+1** = 3001**8** - бит результата **F** - будет заполнять старший бит слова. Команда выполняется при значении 1 бита START36(адрес 02000**8**). В результате сдвига в (S) содержится слово 8080**16**. Бит результата **0** по адресу 3002**8** сброшен, что соответствует значению бита 0 сдвигаемого слова. Бит результата **8** по адресу 3003**8** установлен в 1, что соответствует значению бита 8 сдвигаемого слова.

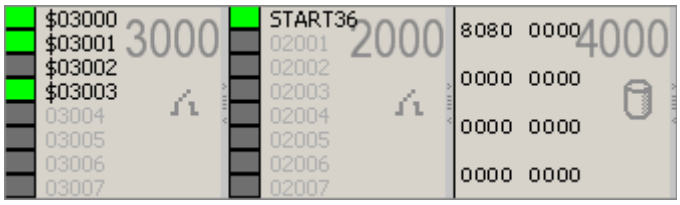
Цепь



До выполнения



После выполнения



[37] ROL Сдвиг слова влево

Параметры

S	Адрес слова
D	
R	

Выходы

Обозначение	Адрес	Условие
0 (15-й бит)	$+1	Запишется в младший бит слова
F (0-й бит)	$+2	Содержимое 15 бита до сдвига
7 (7-й бит)	$+3	Содержимое 7 бита до сдвига

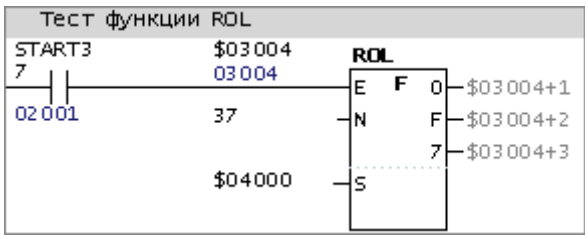
Описание

Сдвиг влево на один бит слова данных, записанных по адресу (S). Бит **$+1** записывается в младший бит слова (S). 15-й бит слова (S) записывается по адресу **$+2**. 7-й бит слова записывается по адресу **$+3**.

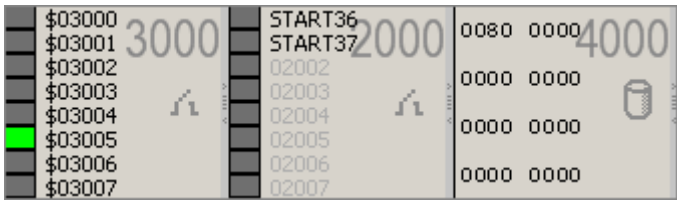
Пример :

Перед выполнением слово 0800**16** записано по адресу S = 4000**8**. Бит "1" записанный по адресу **$+1** = 3005**8** - бит результата **0** - будет заполнять младший бит слова. Команда выполняется при значении 1 бита START37(адрес 02001**8**). В результате сдвига в (S) содержится слово 1010**16**. Бит результата **F** по адресу 3006**8** сброшен, что соответствует значению бита 15 сдвигаемого слова. Бит результата **7** по адресу 3007**8** установлен в 1, что соответствует значению бита 7 сдвигаемого слова.

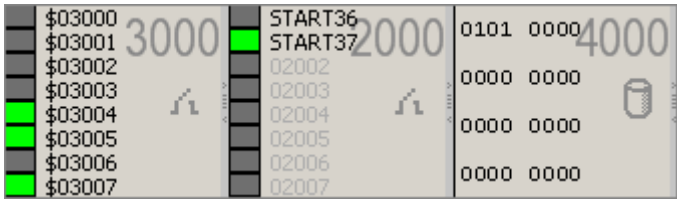
Цепь



До выполнения



После выполнения



[38] BMASK Байтовая маска

Параметры

S	Адрес байта
D	Маска AND
R	Маска OR

Выходы
Описание

Байт по адресу (S) логически умножается на инвертированное значение параметра D и логически складывается со значением параметра R. Полученный результат записывается обратно в (S). Параметры масок задаются числами в восьмеричной системе.

Пример 1:

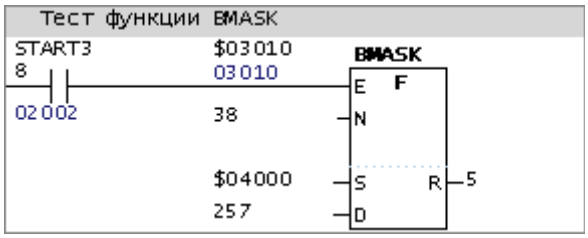
Перед выполнением байт ff**16** записан по адресу S = 4000**8**. Параметр **D** - маска **AND** задана числом 257**8** или 1010 1111**2**. Параметр **R** - маска **OR** задана числом 5**8** или 0000 0101**2**. Команда выполняется при значении 1 бита START38(адрес 02002**8**).

Порядок выполнения:

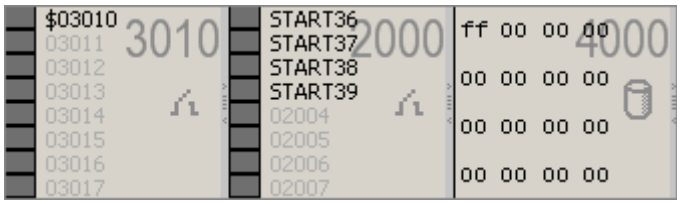
1. байт загружается из адреса (S) - 1111 1111**2**;
2. значение маски AND инвертируется - 0101 0000**2**;
3. инверсия маски AND логически умножается на байт - 0101 0000**2**;
4. к результату прибавляется значение маски OR - 0000 0101**2**;
5. результат 0101 0101**2** записывается в (S).

В результате в (S) = 4000**8** содержится байт 55**16** или 0101 0101**2**.

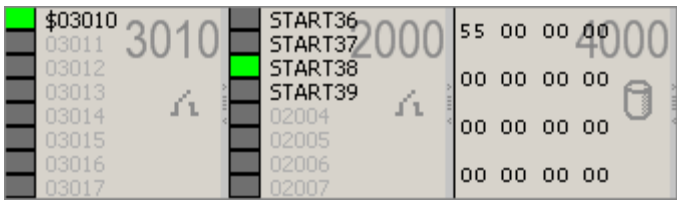
Цепь



До выполнения



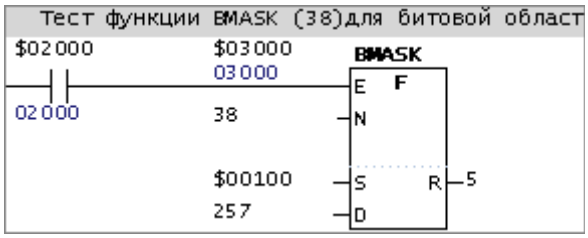
После выполнения



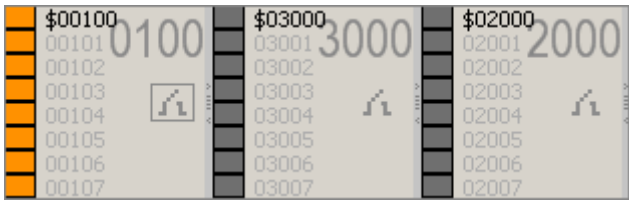
Пример 2:

Тест функции для байта в битовой области. Параметры маски и результат как в Примере 1.

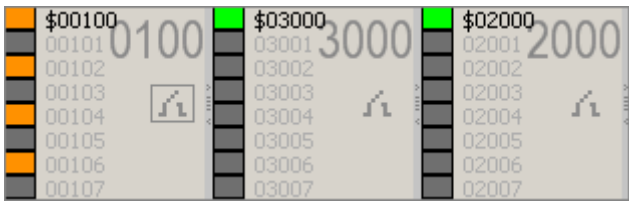
Цепь



До выполнения



После выполнения



[39] WMASK Двухбайтовая маска

Параметры

S	Адрес слова
D	Маска AND
R	Маска OR

Выходы
Описание

Слово по адресу (S) логически умножается на инвертированное значение параметра D и логически складывается со значением параметра R. Полученный результат записывается обратно в (S). Параметры масок задаются числами в восьмеричной системе.

Пример :

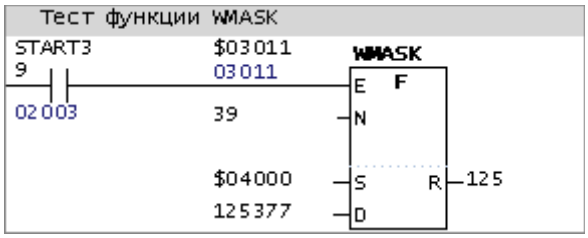
Перед выполнением слово ffff**16** записано по адресу S = 4000**8**.
Параметр **D** - маска **AND** задана числом 125377**8** или 1010 0101 1111 1111**2**.
Параметр **R** - маска **OR** задана числом 125**8** или 0000 0000 0101 0101**2**.
Команда выполняется при значении 1 бита START39адрес 02003**8**).

Порядок выполнения:

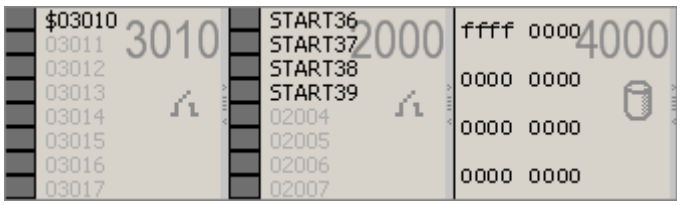
1. слово загружается из адреса (S) - 1111 1111 1111 1111**2**;
2. значение маски AND инвертируется - 0101 0101 0000 0000**2**;
3. инверсия маски AND логически умножается на байт - 0101 0101 0000 0000**2**;
4. к результату прибавляется значение маски OR - 0000 0000 0101 0101**2**;
5. результат 0101 0101 0101 0101**2** записывается в (S).

В результате в (S) = 4000**8** содержится слово 5555**16** или 0101 0101 0101 0101**2**.

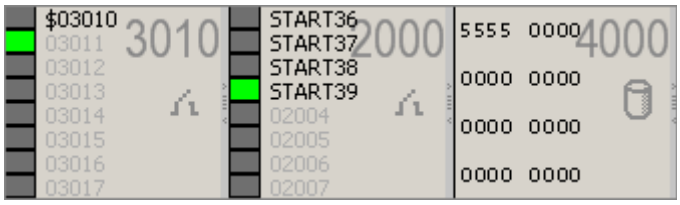
Цепь



До выполнения



После выполнения



[41] PID PID-регулятор

Параметры

S	Байт разрешения регуляторов
D	Формула
R	

Выходы

Обозначение	Адрес	Условие
Err	3720-3727	ошибки регуляторов
K	3730-3737	пересчитывать коэффициенты
Ctrl	3740-3747	Управление

Описание

ПИД-регулирование по 8 каналам по заданной формуле.

Параметр S побитно управляет разрешением работы каналов регулятора, значения битов:

- **0** - запрещено;
- **1** - разрешено.

Номер формулы в параметре D - число 0...4.

Бит **7** параметра D имеет специальное значение:

- **0** - управление аналоговым способом (АЦП);
- **1** - управление двоичным способом (ШИМ).

Формулы ПИД-регуляции:	
параметр D, [8]	**Формула:**
0,100	XOUT=pterm + KP*(ITERM/TR + dterm*TD);

1,101	XOUT=pterm + ITERM/TR + dterm*TD;
2,102	XOUT= U1 +Q**0***error + Q**1***E + Q**2***E**2**; Q**0** = KP * (1 + delta_time/2*TR + TD/delta_time); Q**1** = -KP * (1 + 2*TD/delta_time - delta_time/2*TR); Q**2** = KP * TD/delta_time.
3,103,4,104	XOUT=pterm + dterm + ITERM;

Условные обозначения:

- SV - заданное значение;
- PV - полученное значение;
- KP - пропорциональная константа;
- TR - интегральная или дифференциальная константа при 2-ой производной(ШИМ);
- TD - дифференциальная константа;
- ITERM - интегральная или дифференциальная сумма при 2-ой производной(ШИМ);
- dterm - дифференциальная составляющая;
- pterm - пропорциональная составляющая;
- U1 - предыдущее значение XOUT;
- error - ошибка;
- E**1** - предыдущие значения ошибки;
- E**2** - предыдущие значения ошибки;
- E**3** - предыдущие значения ошибки;
- delta_time - интервал времени;

Адреса памяти данных 74000-74777 используемые ПИД-регулятором:

Адрес слова [8], *1)	**Значение**
74x00	KP
74x02	делитель KP
74x04	TR
74x06	делитель TR
74x10	TD
74x12	делитель TD
74x14	XMIN
74x16	XMAX
74x20	единица квантования (должно быть 0)
74x22	минимальная длина импульса(ШИМ): • 0,1-10ms; • 2-20ms • ... • 100-1000ms
74x24	начальное значение ITERM
74x26	ЗАДАНИЕ

74x30	полученное значение с датчика
74x32	резерв
74x34	фильтр (Error =(Error – E1)*KFILTR + E1; KFILTR = фильтр/100)
74x36	зона нечуствительности (если Error < ЗОНЫ, то Error=0)
74x40	XOUT управляющее воздействие
74x42	error ошибка рассогласования
74x44	интегральная составляющая ITERM
74x46	адрес 1-го бита модуля вывода управления ШИМ: • +0-"вправо"; • +1- "влево".
74x50	ошибки регулятора (0x0000 - 0xffff)
74x52	error ошибка рассогласования по модулю
74x54	резерв до 74x77
Примечания: *1) x - номер регулятора 0...7;	

[50] ENCOD Кодирование в позиционный код

Параметры

S	Адрес байта для кодирования
D	Адрес записи слова результата
R	

Выходы
Описание

Кодирование четырех младших бит байта по адресу (S) в шестнадцатибитный позиционный код.

Примечание
Функции ENCOD [50] и DECOD [51] взаимно обратимы.

Таблица позиционного кода

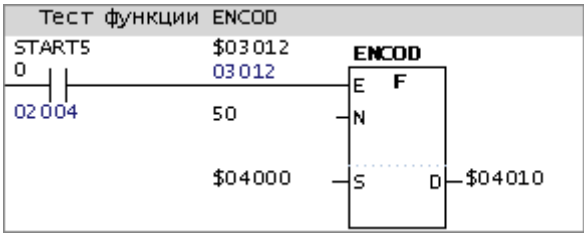
Вход 16	Выход 2
0	0000000000000001
1	0000000000000010
2	0000000000000100
3	0000000000001000
4	0000000000010000

5	0000000000100000
6	0000000001000000
7	0000000010000000
8	0000000100000000
9	0000001000000000
A	0000010000000000
B	0000100000000000
C	0001000000000000
D	0010000000000000
E	0100000000000000
F	1000000000000000

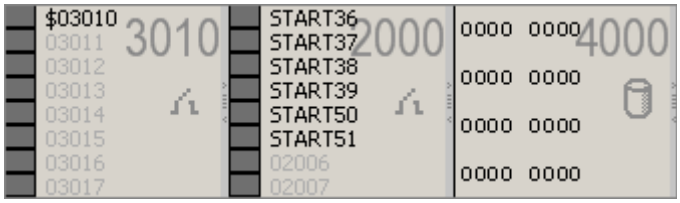
Пример :

Перед выполнением байт 01**16** записан по адресу S = 4000**8**.
Команда выполняется при значении 1 бита START50 (адрес 02004**8**).
В результате в (D) = 4010**8** запишется слово 0001**16** или 0000 0000 0000 0001**2**.

Цепь



До выполнения



После выполнения

[51] DECOD Декодирование позиционного кода

Параметры

S	Адрес слова для декодирования
D	Адрес байта результата декодирования
R	

Выходы

Обозначение	Адрес	Условие
Z (нуль)	$+1	Устанавливается в "1" , если слово в (S) равно 0.

Описание

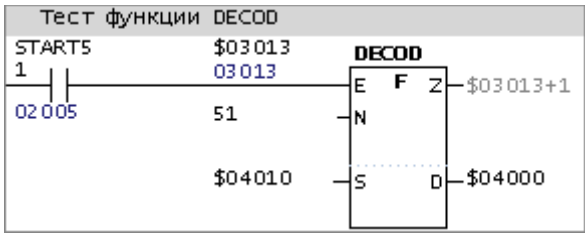
Декодирование слова по адресу (S) в позиционном коде в четырехбитное двоичное число.

Примечание
Функции ENCOD [50] и DECOD [51] взаимно обратимы.

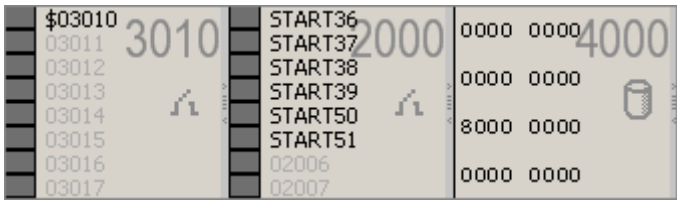
Пример :

Перед выполнением слово 8000**16** записан по адресу S = 4010**8**. Команда выполняется при значении 1 бита START51 (адрес 020058). В результате в (D) = 4000**8** запишется байт 0f**16** или 0000 1111**2**.

Цепь



До выполнения



После выполнения

[53] SWPB Обмен тетрад в байте

Параметры

S	Адрес байта
D	Адрес результата
R	

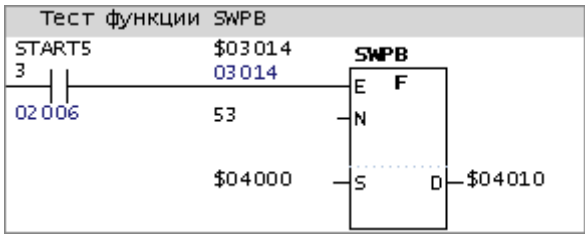
Выходы
Описание

Выполняется обмен данными, записанными в старших и младших тетрадах байта (S). Результат записывается в (D).

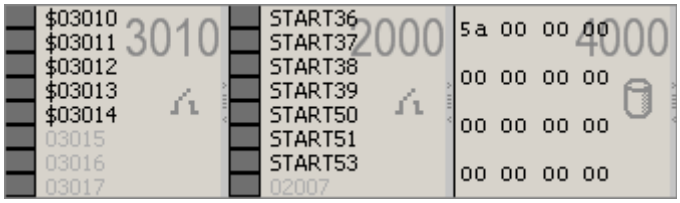
Пример :

Перед выполнением байт 5a**16** записан по адресу S = 4000**8**. Команда выполняется при значении 1 бита START53 (адрес 020068). В результате в (D) = 4010**8** запишется байт a5**16**.

Цепь



До выполнения



После выполнения

[54] SCALE Масштабирование

Параметры

S	Адрес входного слова
D	Адрес результата и коэффициентов
R	Параметр

Выходы
Описание

Масштабирование данных по формуле Y=(A/B)*X+S*C,
где:

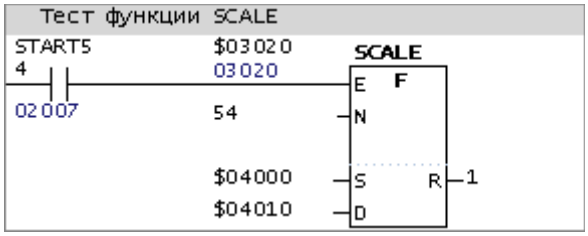
- X – входное значение по адресу (S);
- Y – результат масштабирования (D);
- A – коэффициент масштабирования (D+2);
- B – коэффициент масштабирования (D+4);
- C – коэффициент масштабирования (D+6).
- S – параметр масштабирования (R).

Входные, выходные данные и коэффициенты представлены словами.
Параметр R принимает значения 0 (S=1) или 1 (S=-1).

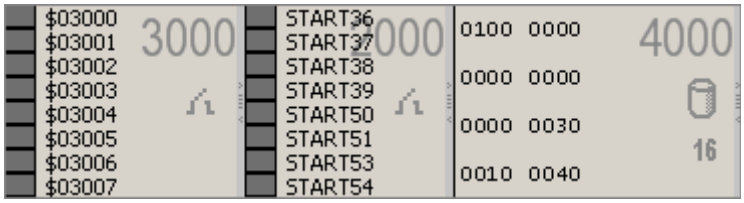
Пример :

Перед выполнением слово 0100**16** записано по адресу S = 4000**8**.
Коэффициенты масштабирования:
A по адресу 4012**8** равен 30**16**;
B по адресу 4014**8** равен 10**16**;
C по адресу 4016**8** равен 40**16**;
параметр S равен -1.
Команда выполняется при значении 1 бита START54 (адрес 02007**8**).
В результате в (D) = 4010**8** запишется результат (30**8**:10**8**)*100**8** -40**8** равный 2c0**16**.

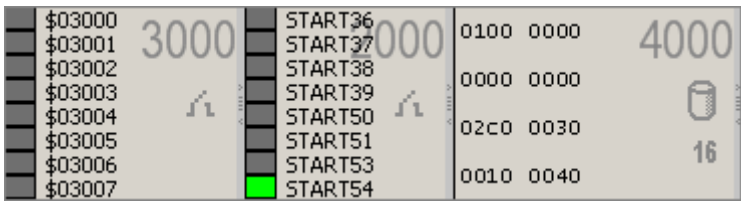
Цепь



До выполнения



После выполнения



[55] MATH Специальные математические функции

Параметры

S	Адрес первого параметра

D	Адрес второго параметра
R	Адрес третьего параметра

Выходы
Описание

[56] TIMSET Уставки в формате времени

Параметры

S	Адрес поля параметров
D	Адрес области таймеров
R	Параметр установки

Выходы
Описание

Преобразует время заданное в формате 'часы,минуты,секунды,0.1сек' в слово и записывает уставку одного или нескольких таймеров для функций **TGRP** [65], **TGRP2** [66]

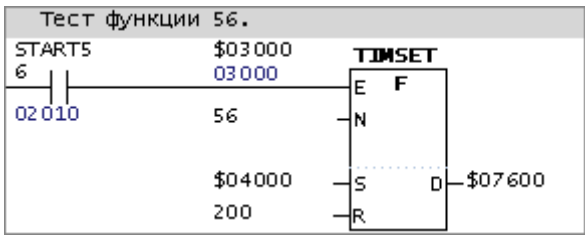
Значения параметров:

S	адрес структуры из четырех слов, определяющему уставку таймера: • S+0 - часы • S+2 - минуты • S+4 - секунды • S+6 - 0,1 сек
D	076xx или 074xx - адрес в области уставок группы таймеров.
R	0.000.000.0aa.0bb.bbb где aa - дискретность(00-0,01с; 01-0,1с; 10-1с; 11-10с); bbbbb – количество (счетчик) таймеров (00000 -1таймер; 11111-32 таймера)
Примечания: 1. область 074XX может применяться только для функции **TGRP2**	

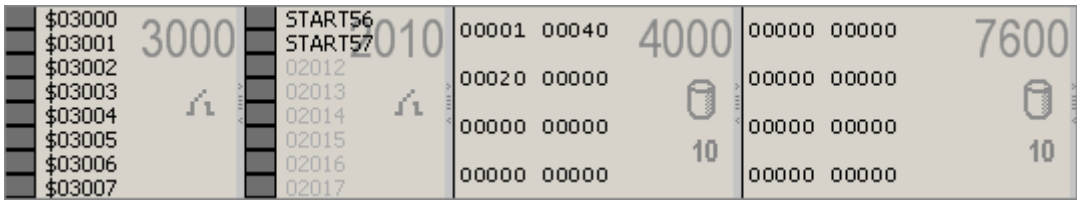
Пример :

	Дискрет 1с	**Дискрет 10с**	**Дискр 0,1с**
4000 01 час	3600	360	36000
4002 40 минут	2400	240	24000
4004 20 секунд	20	2	200
4006 0	0	0	0
07600	6020	602	60200

Цепь



До выполнения



После выполнения

[57] TIMFIX Преобразование уставки во время

Параметры

S	Адрес области таймеров
D	Значение установок
R	Адрес поля параметров

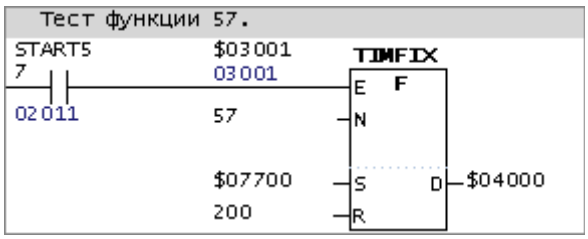
Выходы
Описание

Преобразует уставку таймера для функций 65,66 в формат чмс+0.1

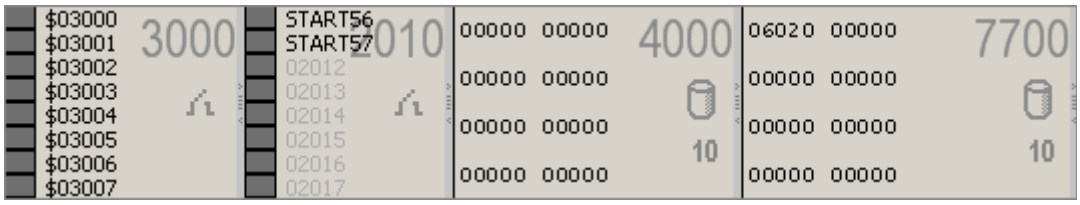
Пример :

| 2024 |57 |---------| |---------(F)----|S=07700 | |D=5000 |R=000200 Дискрет 1с Дискрет 10с Дискрет 0,1с 07700 6020 602 60200 5000 01 час 3600 360 36000 5002 40 минут 2400 240 24000 5004 20 секунд 20 2 200 5006 0 0 0 0 S = 077XX(075XX) (адрес ПД, откуда читается суммарная текущее состояние таймера D = 04000-077777 (адрес ПД, куда запишется текущее значение таймера в формате час+мин+сек+0,1с R = X.XXX.XXX.Xaa.Xbb bbb где aa - дискретность (00-0,01с; 01-0,1с; 10-1с; 11-10с); bbbbb – количество (счетчик) таймеров (00000 -1таймер; 11111-32 таймера)

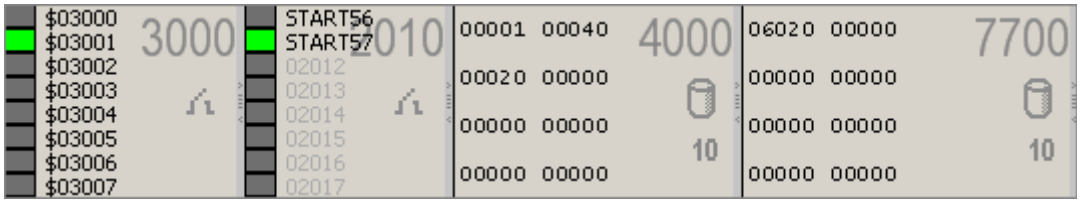
Цепь



До выполнения



После выполнения



[58] ERCODE Индикатор кода аварии

Параметры

S	Адрес байта кода аварии в памяти
D	Адрес бита индикатора
R	

Выходы

Описание

Если установлен хотя бы один бит в байте по адресу (S) кода аварии, бит индикатора по адресу (D). Код аварии также выводится на светодиод.

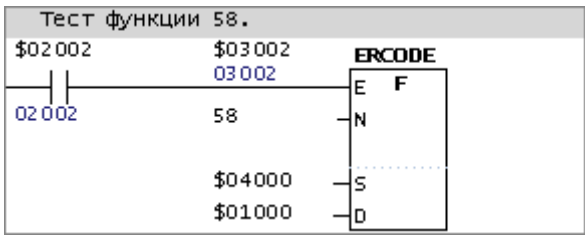
Ошибки

Имя	Адрес	Условие
FErr1	03771**8**	Если адрес в S превышает 01777**8** или адрес в D меньше 04000**8**

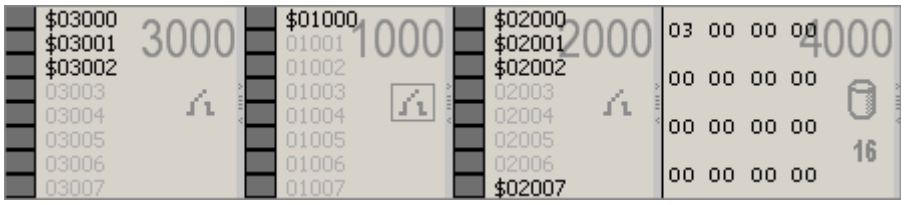
Пример :

Перед выполнением байт 03**16** записан по адресу S = 4000**8**.
Команда выполняется при значении 1 бита $ 02002 (адрес 02002**8**).
В результате бит индикатора в (D) = 1000**8** мигает 3 раза с периодом 1сек с паузой между группами около 2 сек.

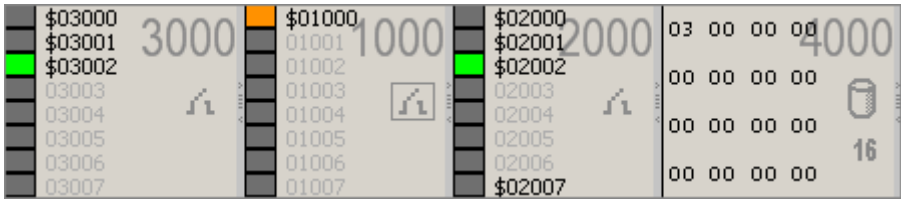
Цепь



До выполнения



После выполнения



[63] INCW Инкремент слова

Параметры

S	Адрес слова уставки счетчика
D	Адрес слова текущего значения
R	

Выходы

Обозначение	Адрес	Условие
>	$+1	Значение счетчика превысило максимальное значение
= (0-й бит)	$+2	Значение счетчика равно уставке

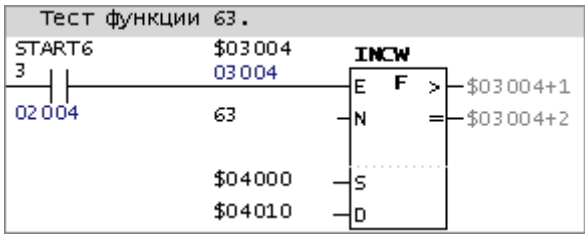
Описание

Прибавляется единица к слову по адресу (S), после чего сравнивается со значением слова уставки по адресу (D).
Максимальное значение уставки - 177777 **8** (65535**10** или FFFF**16**).
Счет продолжается и после совпадения со значением уставки. Сброс текущего значения осуществляется путем записи текущего значения 0.

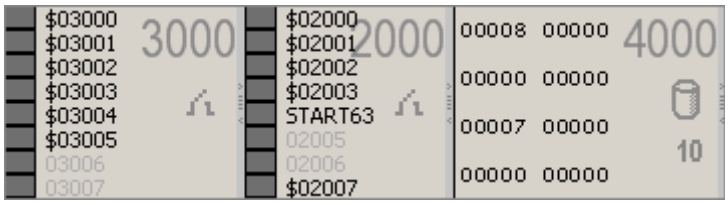
Пример :

Перед выполнением (S) содержит адрес уставки счетчика - слово 0008 записано по адресу S = 4000**8**.
Текущее значение счетчика - слово 0007 записано по адресу D = 4010**8**.
Команда выполняется при значении 1 бита START63 (адрес 02004**8**).
В результате прибавления единицы, текущее значение счетчика 4010**8** будет равно 0008 и сравняется со значением уставки.
Флаг совпадения по адресу 03006**8** устанавливается в 1.

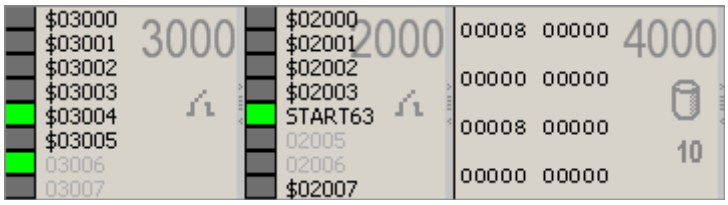
Цепь



До выполнения



После выполнения



[64] DECW Декремент слова

Параметры

S	Адрес слова текущего значения
D	
R	

Выходы

Обозначение	Адрес	Условие
<	$+1	Значение счетчика перешло через 0000
Z (0-й бит)	$+2	Значение счетчика равно 0000

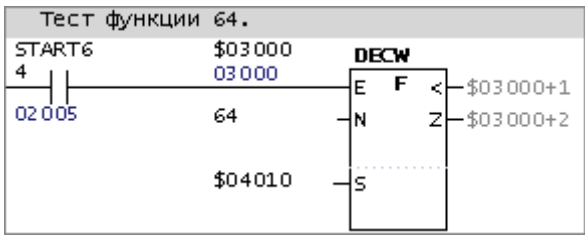
Описание

Вычитается единица от слова по адресу (S).
Счет продолжается и после перехода через нулевое значение.

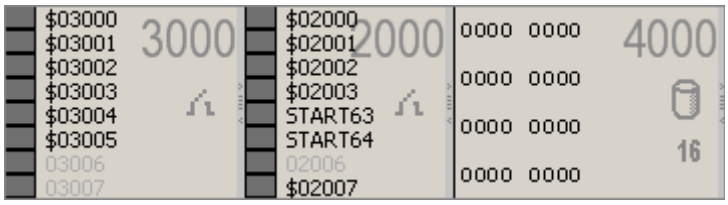
Пример :

Перед выполнением адрес S = 4010**8** содержит текущее значение счетчика - слово 0000.
Команда выполняется при значении 1 бита START64 (адрес 02004**8**).
В результате вычитания единицы, текущее значение счетчика 4010**8** будет равно FFFF**16**.
Флаг **<** по адресу 03001**8** устанавливается в 1.

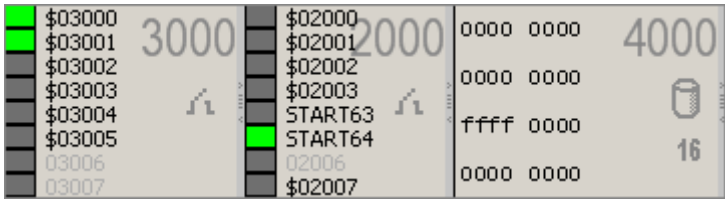
Цепь



До выполнения



После выполнения



[65] TGRP Группа таймеров

Параметры

S	Количество установок
D	
R	

Данная команда выполняет функцию цифрового таймера с дискретностью 0,1 сек и max значением 6553,5 с. Максимальное количество таймеров - 32.

Параметр S — количество установок, задает количество таймеров. Данное значение может быть — 8,16,24,32(в десятичной системе исчисления). Параметр S должен задаваться в восьмеричной системе исчисления, т.е.10,20,30,40. Задание значения отличного от указанного равнозначного заданию значения ближайшего большего из указанного выше набора. Значение >32 вызывает ошибку.

Повторное использование данной функции в скане (в программе) не эффективно, т.к. расчет приращения таймеров выполняется в конце скана. Назначение адресов уставок, текущего значения и сигналов включения счета результата и сравнения с уставкой выполняется по следующему правилу:

1) При S=8

Таймер	Уставка	Тек. значение	Разреш. счета	Сигнал срабатывания счета
0	7660	7760	3660	3661
1	7662	7762	3662	3663
2	7664	7764	3664	3665
-	-	-	-	-
7	7676	7776	3676	3677

1) При S=16

Таймер	Уставка	Тек. значение	Разреш. счета	Сигнал срабатывания счета
0	7640	7740	3640	3641
1	7642	7742	3642	3643
-	-	-	-	-
15	7676	7776	3676	3677

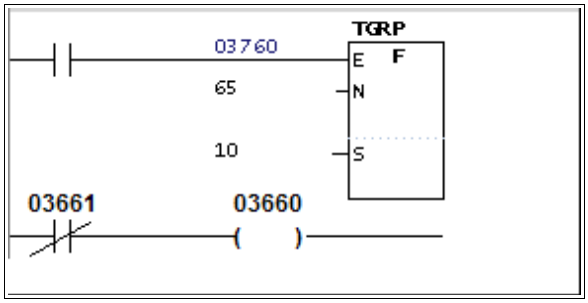
1) При S=24

Таймер	Уставка	Тек. значение	Разреш. счета	Сигнал срабатывания счета
0	7620	7720	3620	3621
1	7622	7722	3622	3623
-	-	-	-	-
29	7676	7776	3676	3677

1) При S=32

Таймер	Уставка	Тек. значение	Разреш. счета	Сигнал срабатывания счета
0	7600	7700	3600	3601
1	7602	7702	3602	3603
-	-	-	-	-
31	7676	7776	3676	3677

Сравнение с уставками выполняется в момент выполнения функции с включением бита результата сравнения с уставкой.
Если бит разрешения выключить до достижения текущего значения уставки, счет приостанавливается. При дальнейшем включении бита разрешения счет продолжается до достижения уставки. Выше уставки счет не выполняется. При достижении уставки включается бит сравнения.
Счетчик сбрасывается в момент выполнения функции при условии достижения уставки и выключения бита разрешения.
Для циклического выполнения счета может использоваться следующая схема.



Следует иметь ввиду, что некоторые функциональные команды (например, функция 36) а так же таймеры -(T)-| и счетчики -(C)-| при работе используют 2 и более бит мгновенного включения (задействован бит с адресом при функции и один или несколько следующих), то использование данной функции для управления битом включения таймера приведет к непредсказуемым результатам.

```
                          3660
|-----| |--------------------(T)--|
```
Запрещено, т.к. бит 3661 используется функцией 65 и таймером -(T)-|.

Значение уставки и текущее значение счетчиков может быть изменена в ходе выполнения скана другими функциональными командами а так же командой записи внешним устройством или программой (Например, Фрегатом 2005). При этом если счетчик достиг уставки, счет приостанавливается, значение бита сравнения включается.
При увеличениии уставки значение бита сравнения выключается, и счет продолжается до достижения нового значения уставки.

Значение уставки и текущее значение представлены как данные BIN.

Примечания:

1. Максимальное число установки - 32, если задать большее, то обнаруживается ошибка установки и таймер не настраивается. MAX S =40 (восьмеричное число);
2. В области внутренних выводов для установки функции таймера нельзя использовать команды -(T)-|, и -(C)-|. Для программирования мгновенного вывода (включения работы) катушки следует использовать команду –( )-|;
3. Установка осуществляется группами по 8 точек. Если установка является числом не кратным 8, то автоматически берется число кратное 8, которое больше установленного.
4. Данная установка сохраняется до момента выключения электропитания.

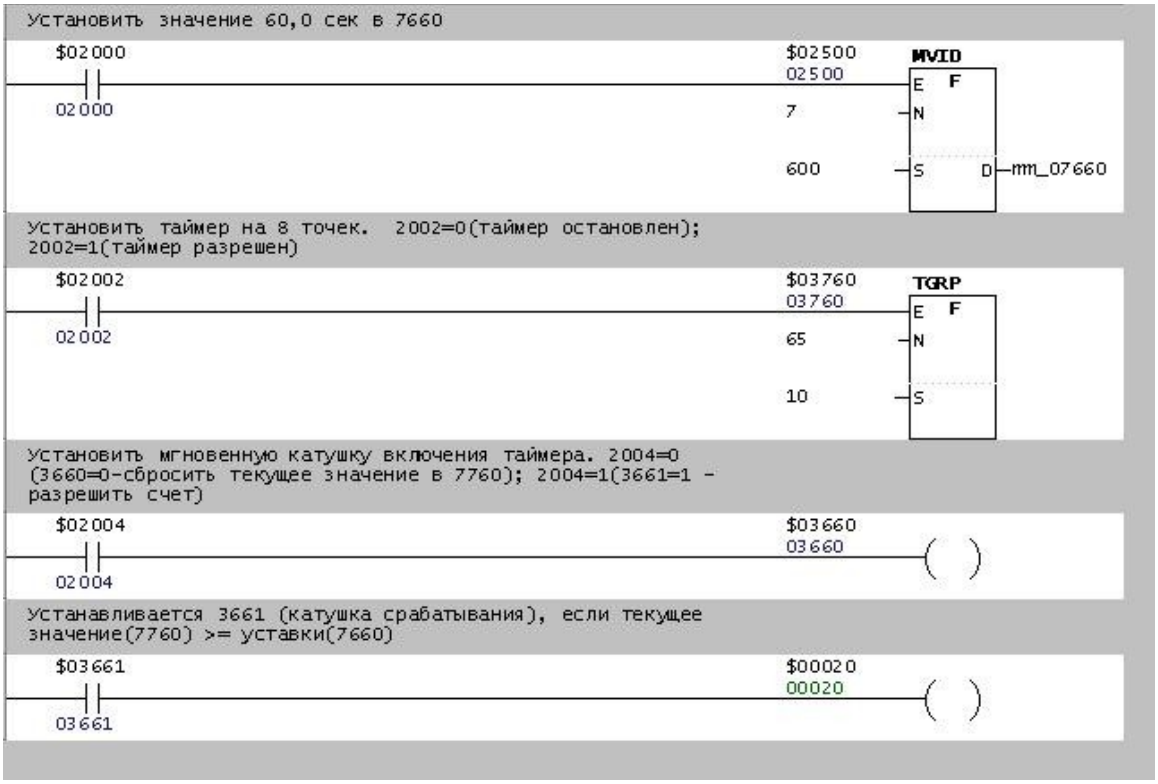
Мгновенная катушка (включения)	**3600, 3602, 3604… 3676**
Контакт срабатывания	**3601, 3603, 3605…3677**
Регистр значения установки	**7600 - 7676**
Регистр текущего значения	**7700 - 7776**

Эти внутренние выводы и регистры выполняют функцию, аналогичную функции --(T).
Мгновенная катушка Выкл. - Текущее значение = 0.
Мгновенная катушка Вкл. - Текущее значение + 1 по 0,1 с.
(Если значение установки < = текущему значению, то текущее значение не меняется.)
Значение установки > Текущего значения - Контакт срабатывания Выкл.
Значение установки <= Текущего значения - Контакт срабатывания Вкл.
(Значение настройки и текущее значение представлены как данные по BIN).

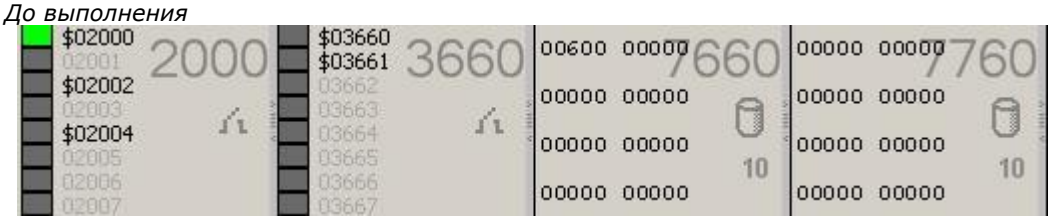
Внимание.
Отсчет времени начнется, если таймер разрешен и мгновенная катушка включена.

Пример :

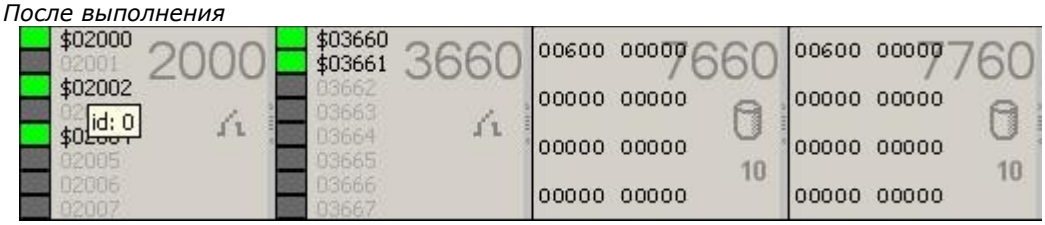
В этом примере внутренние выводы с адресами 3660 - 3677 будут выполнять функцию мгновенной (включения) катушки и контакта срабатывания.



До выполнения



После выполнения



[66] TGRP2 Группа таймеров 2

Параметры

S	начальный адрес уставки таймера
D	начальный адрес ПД или В/В, где располагается СТОП БИТЫ
R	Параметры

Выходы
Описание

Функция установки группы таймеров 2.S = начальный адрес уставки таймера (074XX; 076XX). R = 0.000.00a.0bb.0cc.ccc a -> 0-нет стопа; 1-есть стоп bb -> дискретность cc.ccc -> кол-во таймеров .D = начальный адрес ПД или В/В, где располагается СТОП БИТЫ.

[68] PUSH Ввод в стек

Параметры

S	Адрес дна стека
D	Адрес вершины стека
R	

Выходы
Описание

Запись в назначенный стек байта данных извлеченного по адресу дна стека. При использовании данной команды в сочетании с командой POP (69) область памяти с адресами от S до D используется как стек FIFO " первым пришел - первым вышел " .

Примечания:

1. Емкость области стека составляет максимально 64**10** байта.
2. Значение данных "0" в стеке считается отсутствием данных, поэтому сдвиг не выполняется;

3. При нахождении в назначенной области стека не нулевых данных, сдвиг осуществляется обходя эти данные.

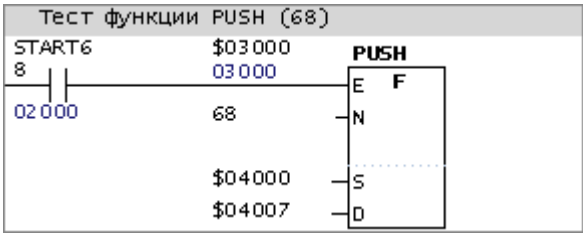
Ошибки

Имя	Адрес	Условие
FErr1	03771**8**	4. При назначении области стека больше 64**10** байт; 5. Если начальный адрес области стека больше чем конечный; 6. Назначенная область стека располагается на двух банках памяти.

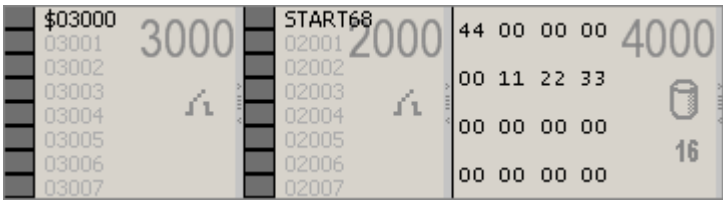
Пример :

Перед выполнением параметры S = 4000 **8** и D = 4007 **8** задают область стека. Вводимый байт 11**16** расположен по адресу дна стека 4000 **8**.
Команда выполняется при значении 1 бита START68 (адрес 02000**8**).
В результате содержимое стека сдвигается по направлению к младшим адресам, вводимое значение записывается в адрес вершины стека .

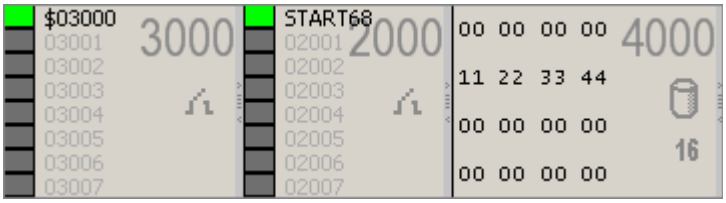
Цепь



До выполнения



После выполнения



[69] POP Вывод из стека

Параметры

S	Адрес дна стека
D	Адрес вершины стека
R	Адрес вывода

Выходы
Описание

Вывод из стека байта данных извлеченного по адресу вершины стека. При использовании данной команды в сочетании с командой PUSH (68) область памяти с адресами от S до D используется как стек FIFO " первым пришел - первым вышел " .

Примечания:

7. Емкость области стека составляет максимально 64**10** байта.
8. Значение данных "0" в стеке считается отсутствием данных, поэтому сдвиг не выполняется;
9. При нахождении в назначенной области стека не нулевых данных в прерывном порядке данная команда не выполняет упорядочение этих данных. Выполняется сдвиг без упорядочения.

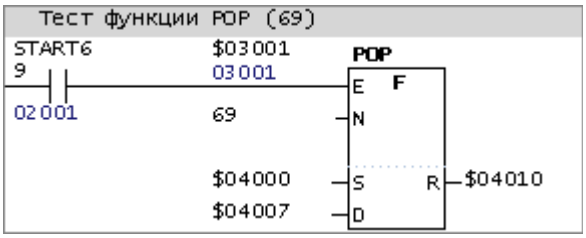
Ошибки

Имя	Адрес	Условие
FErr1	03771**8**	10. При назначении области стека больше 64**10** байт; 11. Если начальный адрес области стека больше чем конечный; 12. Назначенная область стека располагается на двух банках памяти.

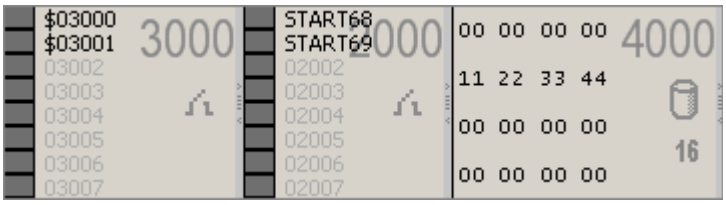
Пример :

Перед выполнением параметры S = 4000**8** и D = 4007**8** задают область стека. Выводимый байт 44**16** расположен по адресу вершины стека 4007**8**. Команда выполняется при значении 1 бита START69 (адрес 02001**8**). В результате содержимое стека сдвигается по направлению от младших адресов к старшим, выводимое значение записывается в адрес R = 4010**8**.

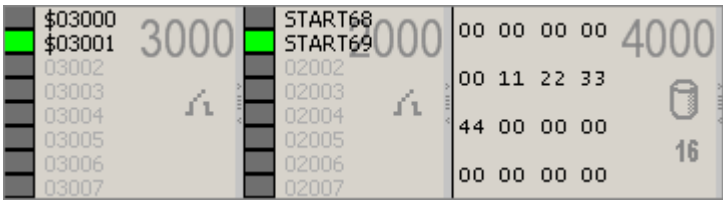
Цепь



До выполнения



После выполнения



[70] MOVBLK Пересылка блока

Параметры

S	регистр начального адреса данных для передачи
D	начальный регистр, в который передаются данные
R	конечный регистр, в который передаются данные

Выходы
Описание

Передача блока данных из S в D длиной R байт. Длина блока не более 256 байт.

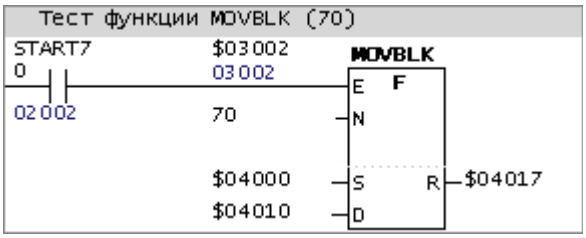
Ошибки

Имя	Адрес	Условие
FErr1	03771**8**	Количество передаваемых данных больше 256

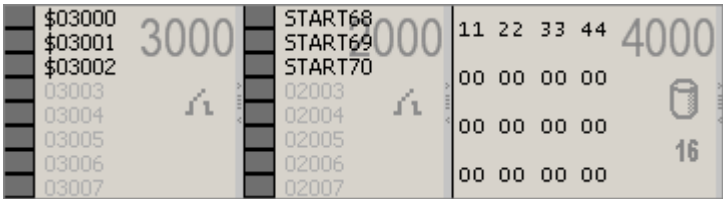
Пример :

Команда выполняется при значении 1 бита START70 (адрес 02002**8**). В результате содержимое области D = 4000**8**-4007**8** переписывается в область начиная с адреса R = 4010**8**.

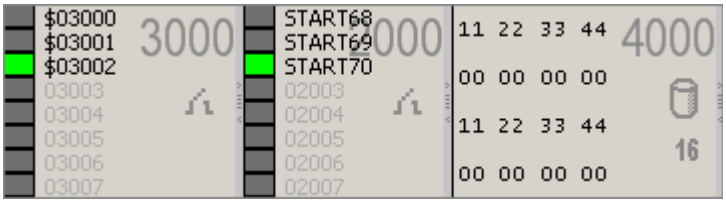
Цепь



До выполнения



После выполнения



[71] MVMBLK Косвенная пересылка блока

Параметры

S	Указатель адреса блока источника

D	Указатель адреса блока назначения
R	Указатель размера блока в байтах

Выходы
Описание

Пересылка блока данных длиной не более 256 байт из одной области памяти в другую. Данная команда использует указатели на слова в памяти содержащие адреса блоков (S) (D) и счетчик длины (R).

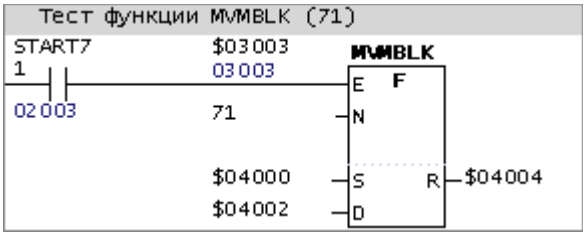
Ошибки

Имя	Адрес	Условие
FErr1	03771**8**	13. Длина блока больше 256**10** байт; 14. Блок назначения частично или полностью выходит за пределы памяти.

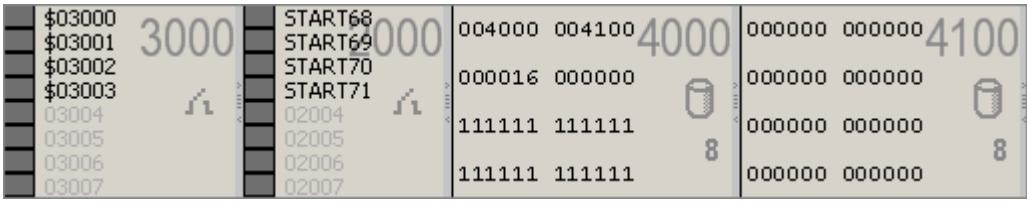
Пример :

Команда выполняется при значении 1 бита START71 (адрес 02003**8**). В результате содержимое области (S) = 4000**8** длиной 16**8**байт переписывается в область начиная с адреса (R) = 4100**8**.

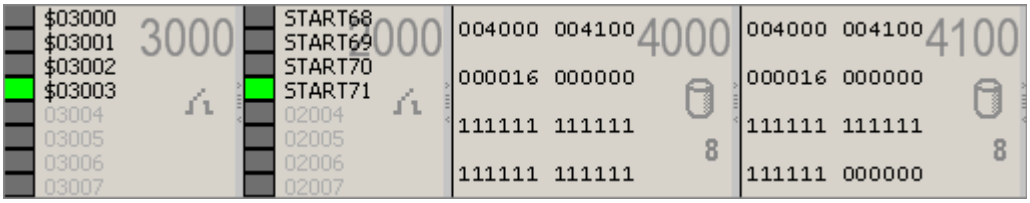
Цепь



До выполнения



После выполнения



[74] STM Запись слова с косвенной адресацией

Параметры

S	Адрес слова
D	Указатель адреса слова назначения
R	

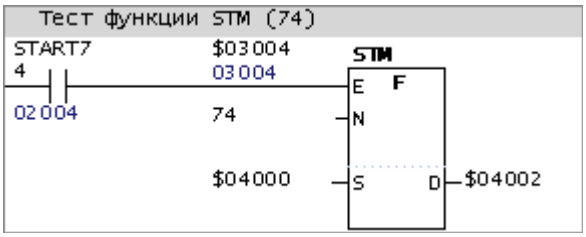
Выходы
Описание

Передача слова из S в (D).
Данная команда использует указатель на адрес слова назначения (D).

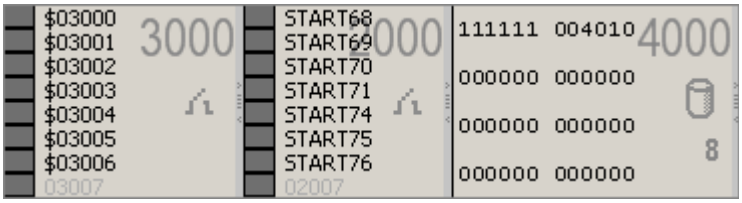
Пример :

Команда выполняется при значении 1 бита START74 (адрес 02004**8**).
В результате слово 111111**8** из (S) = 4000**8** записывается в область (D) = 4010**8**.

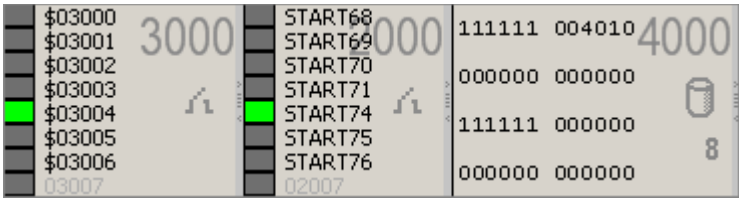
Цепь



До выполнения



После выполнения



[75] LDM Чтение слова с косвенной адресацией

Параметры

S	Указатель адреса слова источника
D	Адрес слова
R	

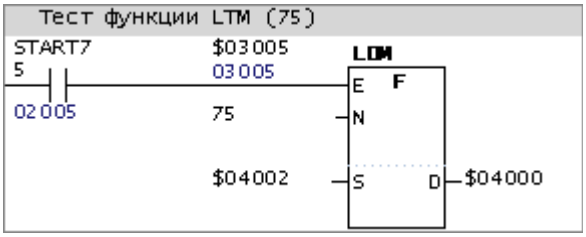
Выходы
Описание

Передача слова из (S) в D.
Данная команда использует указатель на адрес слова (S).

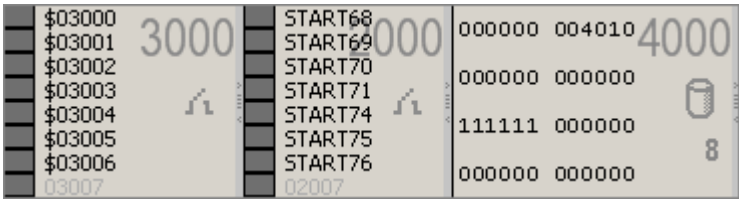
Пример :

Команда выполняется при значении 1 бита START75 (адрес 02005**8**). В результате слово 111111**8** из (S) = 4010**8** записывается в область D = 4000**8**.

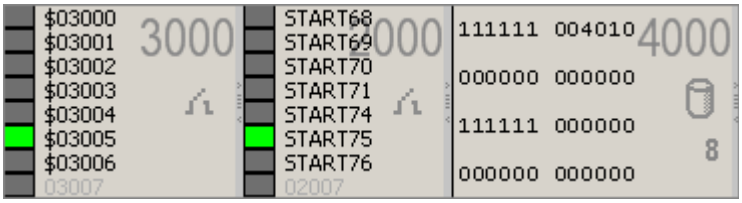
Цепь



До выполнения



После выполнения



[76] MVM Пересылка слова с косвенной адресацией

Параметры

S	регистр, в котором записан адрес регистра, откуда передаются данные
D	регистр, в котором записан адрес регистра, куда передаются данные.
R	

Выходы
Описание

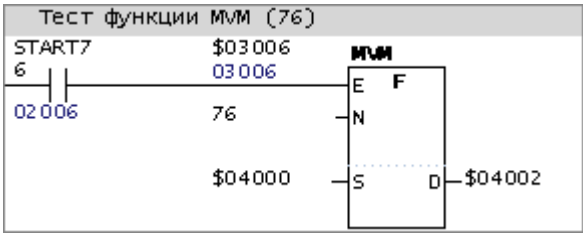
Выполняется передача данных, записанных в косвенно назначенном регистре, в косвенно назначенный регистр. S - регистр, в котором записан адрес регистра, откуда передаются данные. Данные составлены шестнадцатибитным кодом по BIN. D - регистр, в котором записан адрес регистра, куда передаются данные.

Пример :

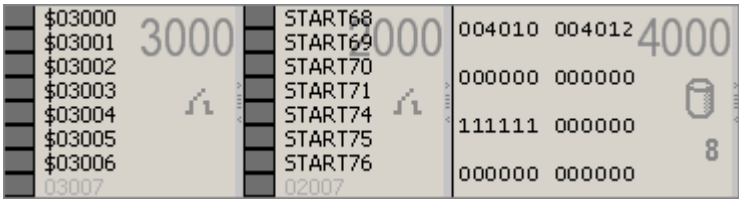
| 3024 | 76 |---------| |---------(F)-----| S=06100 | | D=06322 Предположим, что в адресе 6100 записаны цифры 6524(восьмеричное число), а в адресе 6322 записано 7421(восьмеричное

число). Команда осуществляет передачу данных, записанных по адресу 6322, в адрес 7421. 06322 11010001 06323 00001110 Перед выполнением команды MSB LSB После выполнения команды MSB LSB 6524 00111011 00111011 7421 01010101 00111011

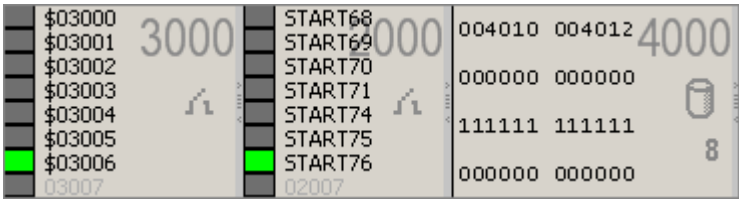
Цепь



До выполнения



После выполнения



[81] PRTO Контрольный бит четность

Параметры

S	Адрес слова
D	Адрес слова результата
R	

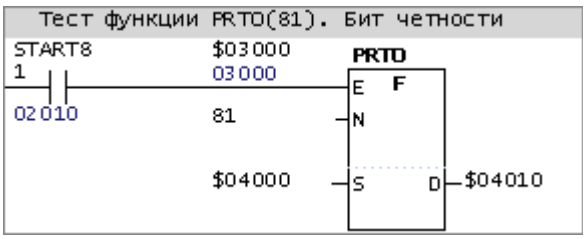
Выходы
Описание

Добавление бита для контроля на четность.
В старший бит записывается значение, дополняющее количество единиц в слове до четного.
Старший бит слова не учитывается.
Значение исходного слова не меняется.

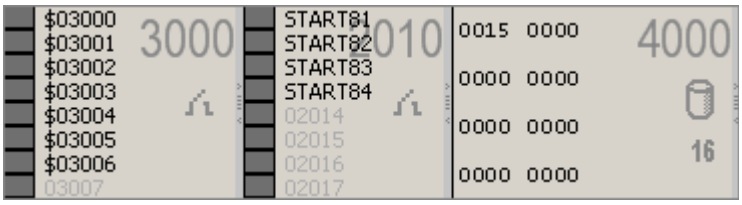
Пример :

Перед выполнением по адресу S = 4000**8** записано тестовое слово 0015**16**. Команда выполняется при значении 1 бита START81(адрес 02010**8**). В результате слово 0015 извлекается из памяти, старший бит заменяется на бит контроля. Результат 8015**16** записывается в адрес R = 4010**8**.

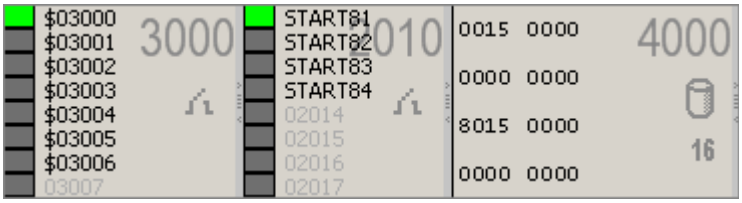
Цепь



До выполнения



После выполнения



[82] PRTOCH Проверка на четность

Параметры

S	Адрес слова
D	
R	

Выходы

Обозначение	Адрес	Условие
=	$+1	o 0 - нет ошибки o 1 - ошибка

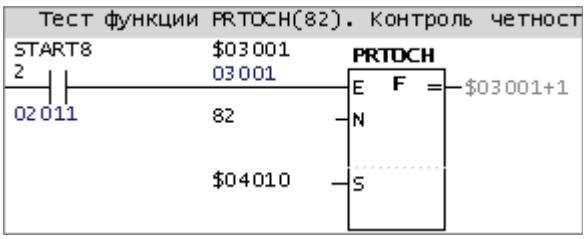
Описание

Контроль 15 бит слова по адресу S на четность. Бит результата контроля выводится в адрес $+1.

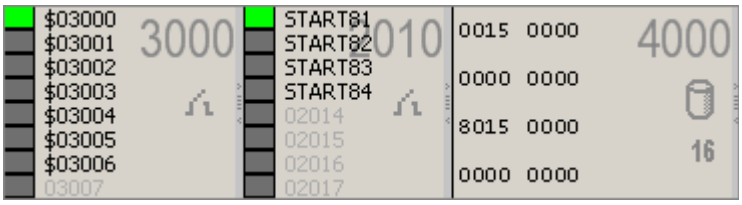
Пример 1:

Перед выполнением по адресу S = 4010**8** записано тестовое слово 8015**16**. Команда выполняется при значении 1 бита START82(адрес 02011**8**). Слово 8015 извлекается из памяти и проверяется на соответствие биту контроля. Результат - нет ошибки - в адрес $+1 = 03002**8** записывается 0 .

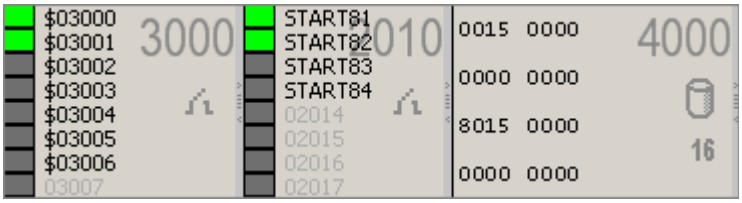
Цепь



До выполнения



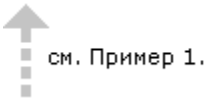
После выполнения



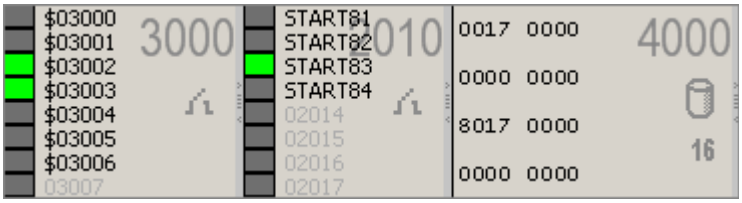
Пример 2:

Перед выполнением по адресу S = 4010**8** записано тестовое слово 8017**16**. Команда выполняется при значении 1 бита START82(адрес 02011**8**). Слово 8017 извлекается из памяти и проверяется на соответствие биту контроля. Результат - ошибка четности - в адрес $+1 = 03002**8** записывается 1.

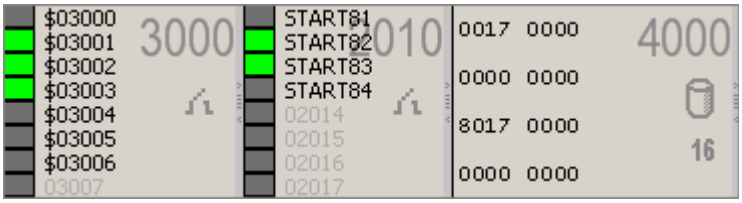
Цепь



До выполнения



После выполнения



[83] PRTE Контрольный бит нечетность

Параметры

S	Адрес слова
D	Адрес слова результата
R	

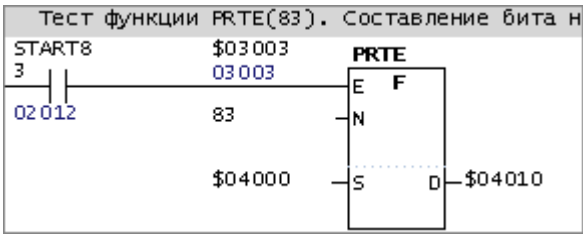
Выходы
Описание

Добавление бита для контроля на четность.
В старший бит записывается значение, дополняющее количество единиц в слове до четного.
Старший бит слова не учитывается.
Значение исходного слова не меняется.

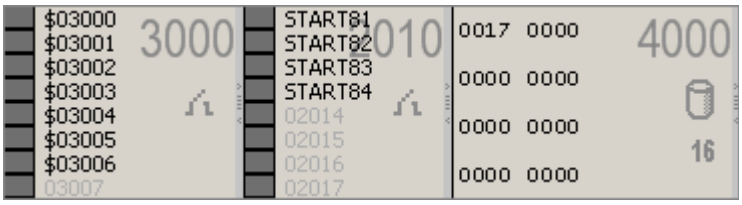
Пример :

Перед выполнением по адресу S = 4000**8** записано тестовое слово 0017**16**. Команда выполняется при значении 1 бита START83(адрес 02012**8**). В результате слово 0017 извлекается из памяти, старший бит заменяется на бит контроля. Результат 8017**16** записывается в адрес R = 4010**8**.

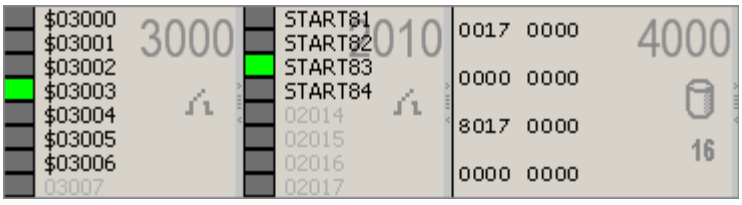
Цепь



До выполнения



После выполнения



[84] PRTECH Проверка на нечетность

Параметры

S	Адрес слова
D	
R	

Выходы

Обозначение	Адрес	Условие
=	$+1	o 0 - нет ошибки o 1 - ошибка

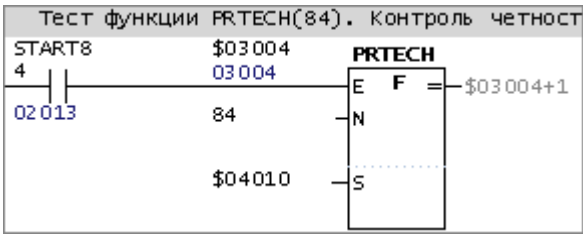
Описание

Контроль 15 бит слова по адресу S на нечетность. Бит результата контроля выводится в адрес $+1.

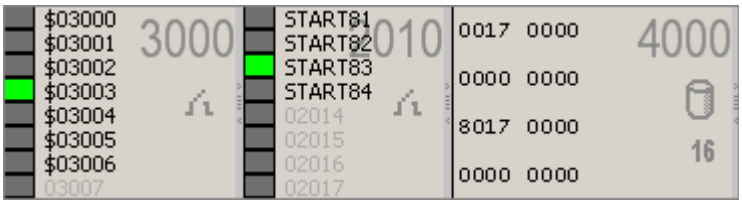
Пример 1:

Перед выполнением по адресу S = 4010**8** записано тестовое слово 8017**16**. Команда выполняется при значении 1 бита START84(адрес 02013**8**). Слово 8017 извлекается из памяти и проверяется на соответствие биту контроля. Результат - нет ошибки - в адрес $+1 = 03005**8** записывается 0 .

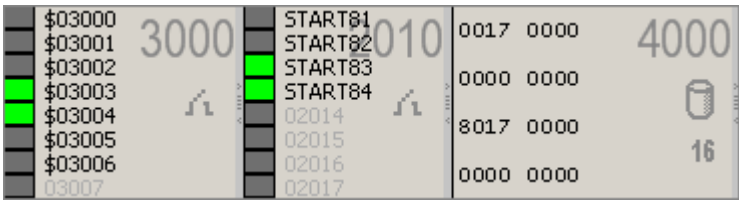
Цепь



До выполнения



После выполнения



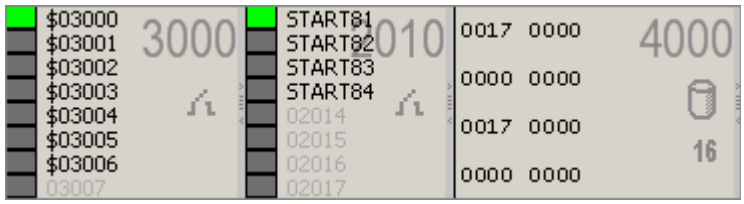
Пример 2:

Перед выполнением по адресу S = 4010**8** записано тестовое слово 0017**16**. Команда выполняется при значении 1 бита START84(адрес 02013**8**). Слово 0017 извлекается из памяти и проверяется на соответствие биту контроля. Результат - ошибка четности - в адрес $+1 = 03005**8** записывается 1.

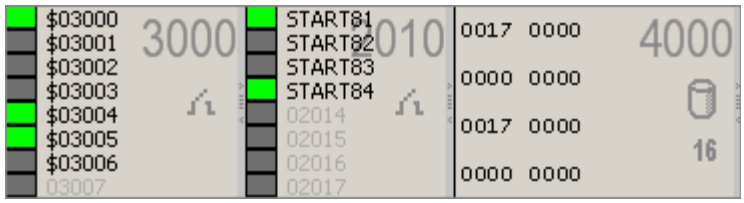
Цепь

см. Пример 1.

До выполнения



После выполнения



[90] MOVB Пересылка байта

Параметры

S	Адрес байта
D	Адрес байта
R	

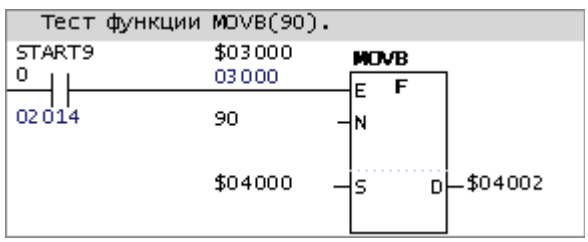
Выходы
Описание

Передача байта из адреса (S) в (D).

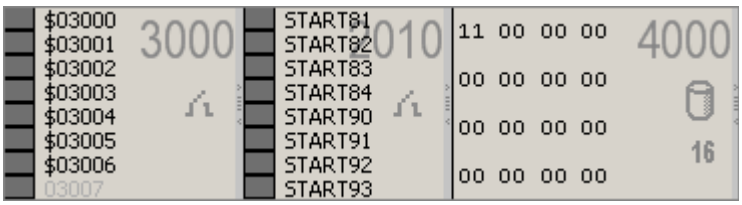
Пример :

Перед выполнением по адресу S = 4010**8** записано тестовое слово 8017**16**. Команда выполняется при значении 1 бита START84(адрес 02013**8**). Слово 8017 извлекается из памяти и проверяется на соответствие биту контроля. Результат - нет ошибки - в адрес $+1 = 03005**8** записывается 0 .

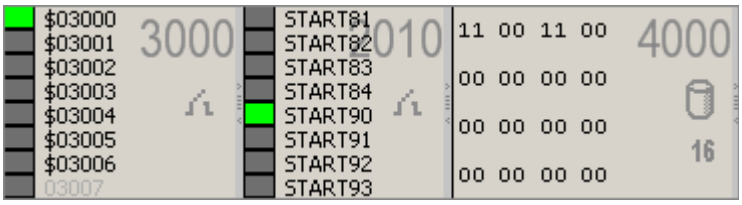
Цепь



До выполнения



После выполнения



[91] SHFTBLK Сдвиг блока по байту

Параметры

S	Адрес начала блока
D	Адрес конца блока
R	

Выходы

Описание

Сдвиг байт в блоке заданном начальным S и конечным D адресами. Блок сдвигаемых данных составляет максимально 64 байта.

Примечания:

1. Если начальный адрес сдвигаемых данных больше, чем конечный адрес, то сдвиг осуществляется в противоположном направлении.
2. При совпадении начального адреса с конечным, сдвиг не выполняется.

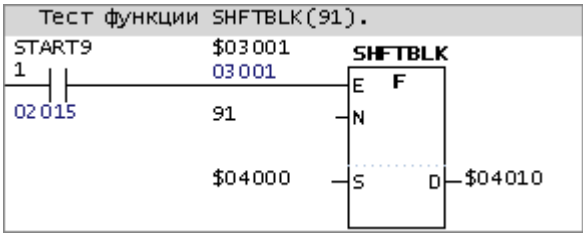
Ошибки

Имя	Адрес	Условие
FErr1	03771**8**	3. Длина блока больше 64**10** байт;

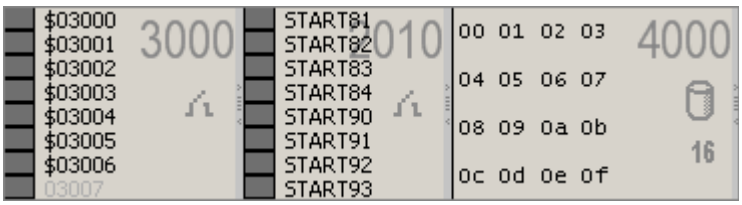
Пример :

Выполняется сдвиг данных, записанных в адресах от S=04000**8** до D=04010**8**.

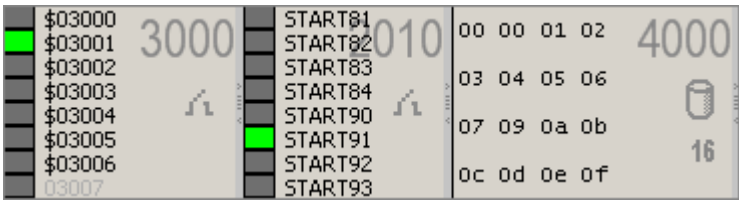
Цепь



До выполнения



После выполнения



[92] ADD Сложение

Параметры

S	Адрес слагаемого
D	Адрес слагаемого
R	Адрес результата

Выходы

Обозначение	Адрес	Условие
>	$+1	Флажок переноса. Выводится в адрес ввода-вывода следующим за адресом катушки установки. Включается, когда результат сложения превысит четыре разряда и переносится на пятый разряд. Если переноса в пятый разряд нет, то флажок выключается.
Z	$+2	Флажок нуля. Выводится в адрес " адрес катушки установки + 2". Включается, если результат сложения равен нулю, в противном случае выключается.

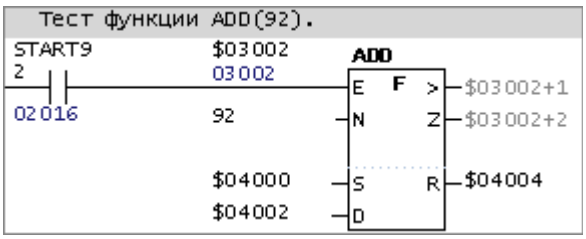
Описание

Выполняется сложение данных в двоичной системе, записанных в назначенных регистрах. S - регистр первого слагаемого. D - регистр второго слагаемого. R - регистр результата сложения. Данные представлены шестнадцатью разрядами. Флажок переноса. Выводится в адрес ввода-вывода следующим за адресом катушки установки. Включается, когда результат сложения превысит четыре разряда и переносится на пятый разряд. Если переноса в пятый разряд нет, то флажок выключается. Флажок нуля. Выводится в адрес " адрес катушки установки + 2". Включается, если результат сложения равен нулю, в противном случае выключается.

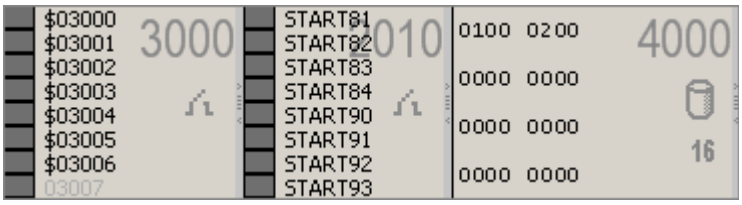
Пример :

Слагаемые записаны по адресам: (S) 04000**8** = 0100**16** и (D) 04002**8**= 0200**16**. Команда выполняется при значении 1 бита START92(адрес 02016**8**). Результат сложения 0300**16** записывается по адресу (R) 04004**8**.

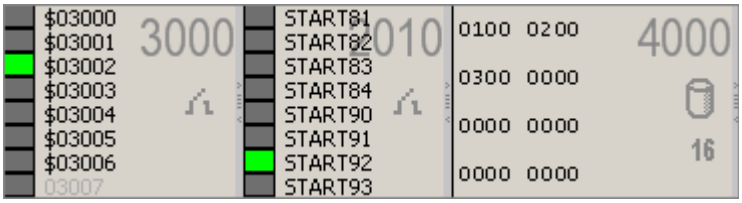
Цепь



До выполнения



После выполнения



[93] SUB Вычитание

Параметры

S	Адрес уменьшаемого
D	Адрес вычитаемого
R	Адрес результата

Выходы

Обозначение	Адрес	Условие
<	$+1	Флажок заема
Z	$+2	Флажок нуля. Выводится в адрес " адрес катушки установки + 2". Включается, если результат сложения равен нулю, в противном случае выключается.

Описание

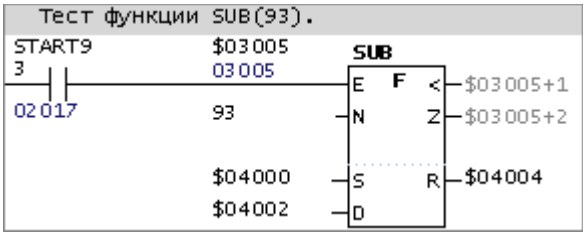
Вычитание данных. S - регистр уменьшаемого. D - регистр вычитаемого. R - регистр результата вычитания. Данные представлены шестнадцатью разрядами. Флажок заема. Выводится в адрес ввода-вывода, следующим за адресом катушки установки. Флажок нуля. Выводится в адрес "адрес катушки установки + 2 ". Включается, если результат вычитания равен нулю, в противном случае выключается.

Пример :

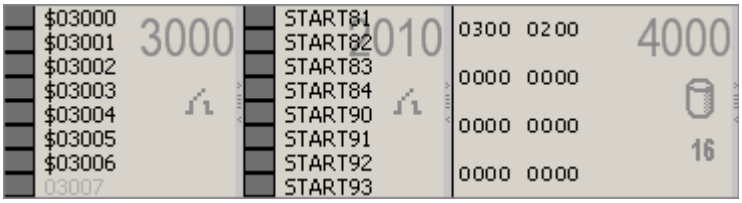
Слова записаны по адресам: уменьшаемое (S) 04000**8** = 0300**16** и вычитаемое (D) 04002**8**= 0200**16**.

Команда выполняется при значении 1 бита START93(адрес 02017**8**).
Результат вычитания 0100**16** записывается по адресу (R) 04004**8**.

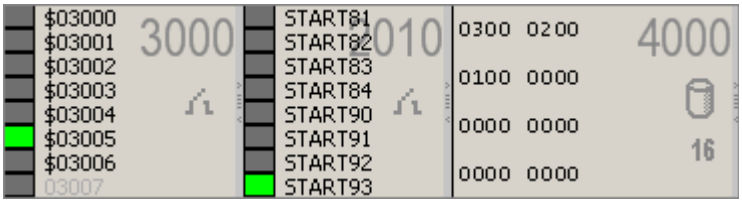
Цепь



До выполнения



После выполнения



[94] MUL Умножение

Параметры

S	Адрес множимого
D	Адрес множителя
R	Адрес результата

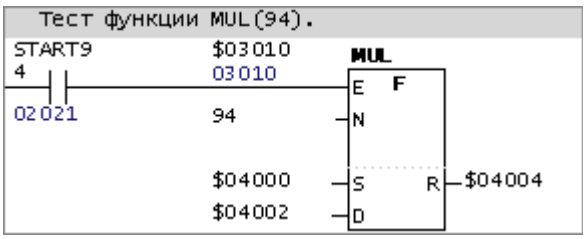
Выходы
Описание

Выполняется беззнаковое умножение данных.

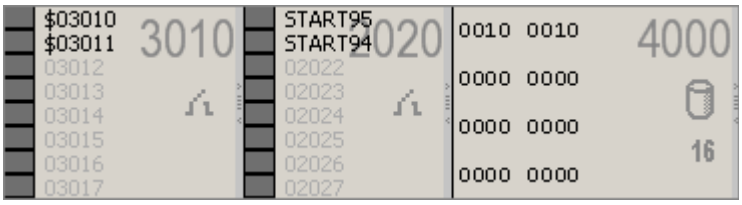
Пример :

Слова записаны по адресам: множимое (S) 04000**8** = 0010**16** и множитель (D) 04002**8**= 0010**16**. Команда выполняется при значении 1 бита START94(адрес 02021**8**). Результат умножения 0000010**016** записывается по адресу (R) 04004**8**.

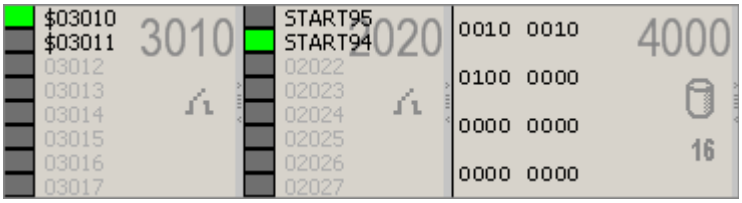
Цепь



До выполнения



После выполнения



[95] DIV Деление

Параметры

S	Адрес делимого
D	Адрес делителя
R	Адрес частного

Выходы

Описание

Выполняется деление данных в системе BIN, записанных в назначенных регистрах. S - регистр делимого. Данные шестнадцатиразрядные. D - регистр делителя. Данные восьмиразрядные. R - регистр результата деления. Данные двадцатичетырехразрядные.

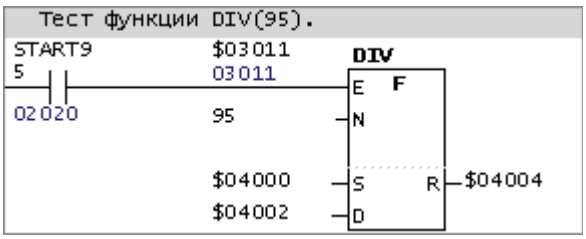
Ошибки

Имя	Адрес	Условие
FErr1	03771**8**	Если делитель равен нулю

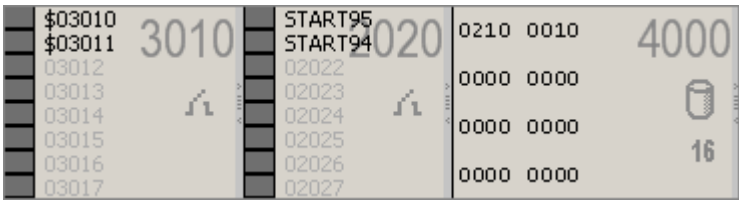
Пример :

Слова записаны по адресам: делимое (S) 04000**8** = 0210**16** и делитель (D) 04002**8**= 0010**16**. Команда выполняется при значении 1 бита START95(адрес 02020**8**). Результат деления 00002**16** записывается по адресу (R) 04004**8**.

Цепь



До выполнения



После выполнения

[98] ADCWR Запись уставок.

Параметры

S	кодовое значение верхней уставки
D	кодовое значение нижней уставки
R	задается адрес модуля, номер канала и режим

Выходы
Описание

Коды уставок, представленные в восьмеричной системе счисления, задаются в параметрах команды. S - кодовое значение верхней уставки. D - кодовое значение нижней уставки. R - задается адрес модуля, номер канала и режим. Диапазон изменения параметров S и D от 0000 до 7777. На модуле ввода должен быть выбран такой диапазон, чтобы значения верхней и нижней уставок входили в него. Если модуль настроен на работу в диапазоне от 0 до Umax, параметры вычисляются по формуле : U S(D)= ----------* 2048 Umax где U - напряжение задаваемой уставки; Umax - максимальное значение напряжение заданного диапазона. Если модуль настроен на работу в диапазоне от минус Umax до Umax, параметры S и D вычисляются по формуле: U+Umax S(D)= --------------------* 2048 Umax

Результат вычисления по формулам необходимо округлить до ближайшего целого числа и перевести в восьмеричную систему счисления. Параметр R располагается в двух байтах, занимая в них 13 младших бит. В старших трех битах должны быть записаны "0".

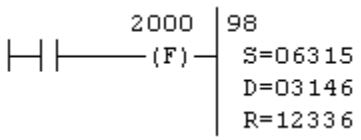
0 0 0 R12 R11 R10 R9 R8 R7 R6 R5 R4 R3 R2 R1 R0 Адрес модуля Номер канала Режим Модуль, подключенный к УПУ-ТП, имеет восьмеричный адрес и принимает значения в диапазоне от 000 до 177 в битах R6 - R12. Номер канала записывается в биты R3 - R5 и принимает значения от 0 до 7: 0 - 3 - номера каналов ввода, 4 - 7 - номера тестовых каналов. Режимы принимают значения от 0 до 7 и записываются в биты R0 - R2, каждый из которых имеет свое функциональное назначение: R2 - признак аналого-цифрового преобразования. Единица в этом бите вызывает выработку кода аналого-цифрового преобразования сигнала,

подключенного к заданному каналу. R1 - признак "ТЕСТ". Единица в этом бите вызывает однократное выполнение тестовой программы. R0 - признак программной фильтрации. Единица в этом бите вызывает вычисление среднего значения последних 32 кодов аналого-цифрового преобразования. Для задания параметра R необходимо полученные восьмеричные числа расположить последовательно друг за другом в порядке: номер модуля, номер канала, режим.

Пример :

Требуется запрограммировать третий канал модуля с адресом 123. Нижняя уставка - минус 2В, верхняя 6В. Диапазон измерений модуля от "-10" до "+10" В. При этом необходимо убедиться в работоспособности модуля, т.е. выполнить тест. Вычисляем параметры функции S и D по вышеприведенным формулам: 6+10 S= ------- * 2048 = 3276,8 3277 10 Число 3277 десятеричное, его восьмеричный эквивалент 6315. Следовательно, первый параметр функции S = 6315. Аналогично вычисляется второй параметр: -2+10 D= ------- * 2048 = 1638,4 1638 10 Восьмеричный эквивалент десятичного числа 1638 = 3146. Следовательно, второй параметр функции D = 3146. Из составляющих параметр R неизвестен только режим. Если пользователя будет интересовать код измеряемой величины напряжения, то задаем признак аналого-цифрового преобразования "1". Для однократного выполнения программы признак "ТЕСТ" также задается равным "1". Программная фильтрация в задании отсутствует, поэтому соответствующий признак принимается "0". В итоге третий параметр функции R = 12336. Следовательно, чтобы выполнить поставленную задачу, нужно использовать F98 следующим образом:

|*Цепь*



[99] ADCRD Считывание кодов АЦП

Параметры

S	задается адрес памяти данных, начиная с которого будут записываться принятые коды аналого-цифрового преобразования или адрес ввода- вывода
D	задается код номера каналов, коды которых будут считаны. Принимает значения от 00 до 17.
R	задается адрес модуля, к которому идет обращение. Принимает значение от 000 до 177.

Описание

По данной команде осуществляется считывание кодов аналого-цифрового преобразования каналов, указанных в команде. Одной командой можно считать до четырех каналов. S - задается адрес памяти данных, начиная с которого будут записываться принятые коды аналого-цифрового преобразования или адрес ввода- вывода. D - задается код номера каналов, коды которых будут считаны. Принимает значения от 00 до 17. R - задается адрес модуля, к которому идет обращение. Принимает значение от 000 до 177. Формат слов, принимаемых функцией 99:

A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0		ОШ	NK1	NK0

- A0 - A11 - код аналого-цифрового преобразования.
- ОШ - признак ошибки модуля.
- NK - номер канала.

Структура параметра D:

0	0	0	0	0	0	0	0	0	0	0	0	D3	D2	D1	D0

Каждый из значащих битов задает на чтение соответствующий ему канал.

```
Канал ввода      тестовый канал
                D3 -     0                  4
                D2 -     1                  5
                D1 -     2                  6
                D0 -     3                  7
```

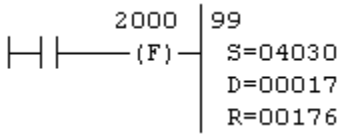
Единица в каждом бите означает, что соответствующий канал задан. Один бит может задавать два разных канала . Параметр R представляет собой восьмеричный адрес модуля, к которому идет обращение по данной команде.

Пример :

```
Вывести в память данных, начиная с адреса 4030,  коды всех четырех каналов ввода модуля с

адресом 176. Для выполнения  этого задания надо использовать F99 следующим образом
```

Цепь



[300] NODE Установить сетевой адрес

Параметры

S	Адрес узла (0...255) для последующих команд NREAD, NWRITE
D	
R	

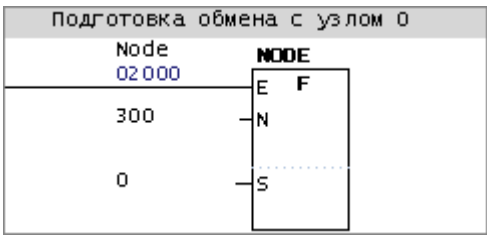
Выходы
Описание

Устанавливает адрес ПЛК в сети для операций NREAD, NWRITE.

Пример :

Команда устанавливает сетевой адрес 0 для последующего обмена.

Цепь



[301] NREAD Чтение данных удаленного ПЛК

Параметры

S	область памяти удаленного ПЛК откуда читать данные
D	область NETPLC куда писать
R	длина блока в байтах

Выходы

Ошибка чтения

Описание

Чтение данных из области S установленного ПЛК в область D NETPLC длиной R. Выход O = 0 при выключенном, 1 - при включенном ПЛК.

Пример :

Команда читает блок данных с удаленного ПЛК: адрес 04010_8 длина - 4 байта, в область NetPlc заданную символическим именем Meminp. Считывание

Цепь

[302] NWRITE Запись в удаленный ПЛК

Параметры

S	область NETPLC откуда читать
D	область памяти удаленного ПЛК куда писать данные
R	длина блока в байтах

Выходы
Описание

Запись данных из области S NETPLC в ПЛК по адресу D длиной R. Выход O = 0 при выключенном, 1 - при включенном ПЛК.

[303] NMSG Запись сообщения

Параметры

S	номер сообщения
D	группа сообщения
R	тип сообщения

Выходы
Описание

Запись сообщения S в журнал группа D тип R.

[304] NSTORE Запись в хранилище

Параметры

S	адрес источника данных
D	группа данных
R	тип данных

Выходы
Описание

Запись 2-х байт значения из адреса S группа D типа R.

[305] NSYNC Автогенератор

Параметры

S	период синхроимпульсов
D	фаза синхроимпульса
R	

Выходы

Выходная ячейка автогенератора

Описание

Синхроимпульсы с уровнем 1 периодом S фазой D в ячейке следующей за адресом настройки.

[306] NGREAT Больше

Параметры

S	адрес данных
D	адрес данных
R	

Выходы

">" в ячейке, следующей за адресом катушки настройки.

Описание

Сравнение слова по адресу S со словом по адресу D. Результат 1 при S>D в ячейке, следующей за адресом катушки настройки.

[307] NLESS Меньше

Параметры

S	адрес данных
D	адрес данных
R	

Выходы

"<" в ячейке, следующей за адресом катушки настройки.

Описание

Сравнение слова по адресу S со словом по адресу D. Результат 1 при S<D в ячейке, следующей за адресом катушки настройки.

[308] NEQUAL Равно

Параметры

S	адрес данных
D	адрес данных
R	

Выходы

"=" в ячейке, следующей за адресом катушки настройки.

Описание

Сравнение слова по адресу S со словом по адресу D. Результат 1 при S=D в ячейке, следующей за адресом катушки настройки.

[309] NMEM Тестовая команда

Параметры

S	адрес куда записать данные
D	
R	

Выходы
Описание

Читает размер памяти в ячейку S

[310] NAVG Усреднитель

Параметры

S	адрес входного значения
D	адрес усредненного значения
R	

Выходы

Вход сброса

Описание

Усредняет значения. Входные значения в (S), усредненные в (D). Использует два байта по адресу (D+2) Для хранения счетчика. Вход сброса - следующий за катушкой настройки.

[311] NSETDI Задать слово

Параметры

S	Куда занести данные
D	Данные для ввода
R	

Выходы
Описание

Установка 16-бит числа в ячейку S

[312] NMOVB Пересылка байт

Параметры

S	адрес блока источника
D	адрес блока приемника
R	длина блока

Выходы
Описание

Пересылка (R) байт из (S) в (D) из любой области в любую.

[313] NMOVW Пересылка слов

Параметры

S	адрес блока источника
D	адрес блока приемника
R	длина блока

Выходы
Описание

Пересылка (R) 2-х байт слов из (S) в (D) из любой области в любую. Длина измеряется в словах.

[314] NMAX Максимум

Параметры

S	Входное значение
D	Максимальное значение
R	

Выходы

Сброс

Описание

Сохраняет максимальное значение в (D). Сброс в 0 по 0 в ячейке следующей за адресом настройки.

[315] NLESSI Меньше значения

Параметры

S	Адрес данных

D	Непосредственное значение
R	

Выходы

"<" меньше

Описание

Сравнение слова по адресу S со значением в D. Результат 1 при S<D в ячейке, следующей за адресом катушки настройки.

[316] NGREATI Больше числа

Параметры

S	Адрес данных
D	Непосредственное значение
R	

Выходы

">" больше

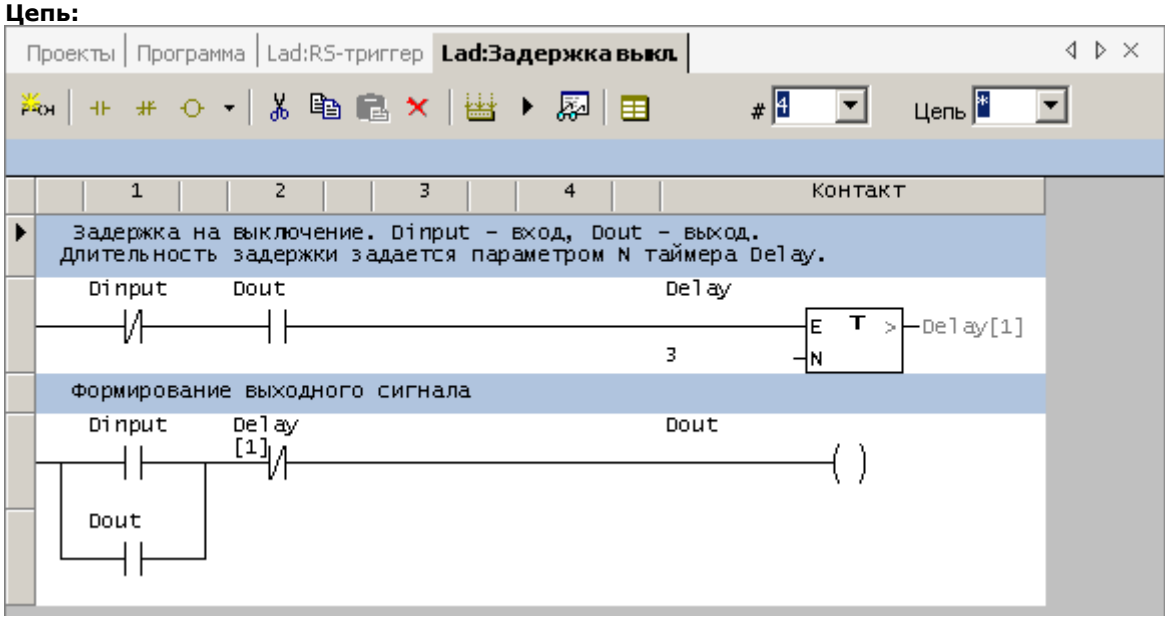
Описание

Сравнение слова по адресу S со значением в D. Результат 1 при S>D в ячейке, следующей за адресом катушки настройки.

ПРИЛОЖЕНИЕ 1. Примеры программирования на РКС.

1. Задержка на выключение

Цепь:



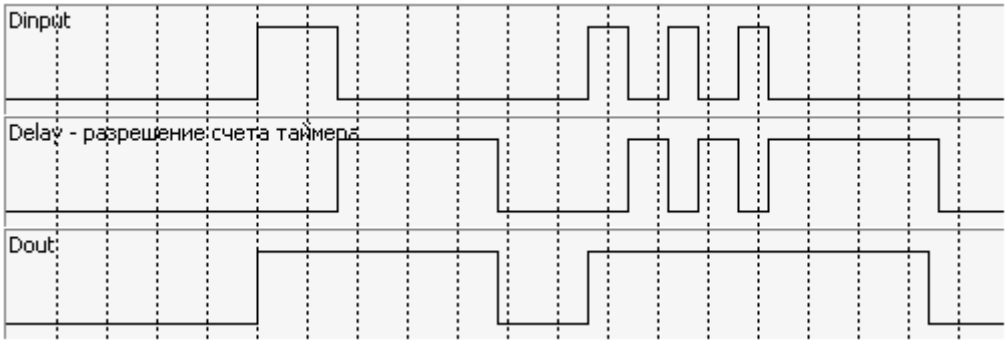
Контакты:

Dinput	входной сигнал;
N	программируемая задержка отключения задается параметром таймера ;
Delay	катушка настройки таймера;
Dout	выходной сигнал.

Описание работы цепи:

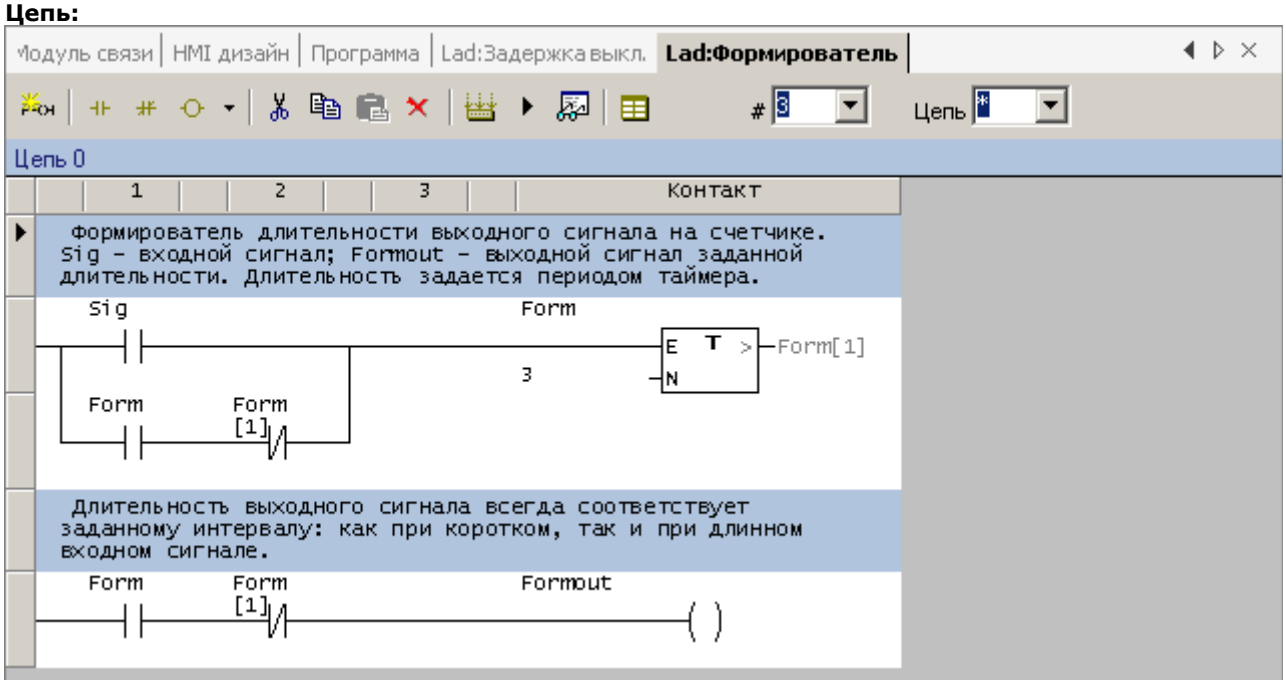
Выходной сигнал Dout переключается в «1» одновременно со включением входного сигнала Dinput . При отключении Dinput- переход из «1» в «0» - начинается отсчет задержки выключения на таймере. По окончании задержки Dout переходит в «0». Цепь не чувствительна к быстрым переключениям входного сигнала («дребезг») - если они происходят за время, меньшее задержки на отключение, выходной сигнал остается в «1».

Временная диаграмма:



2. Формирование длительности выходного сигнала

Цепь:



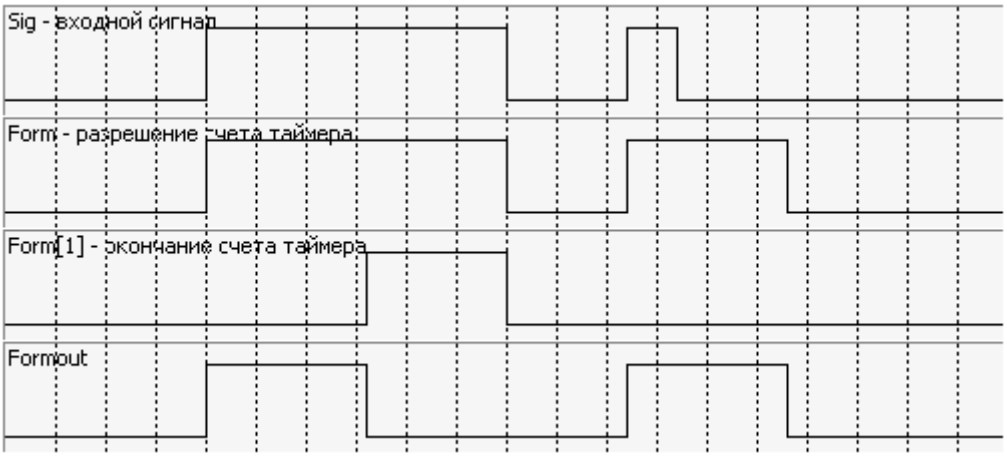
Контакты:

Sig	входной сигнал;
Form	катушка настройки таймера на заданную длительность;
Form[1]	катушка окончания счета таймера;
N	параметр таймера равный текущее время формирователя;
Formout	выходной сигнал.

Описание работы цепи:

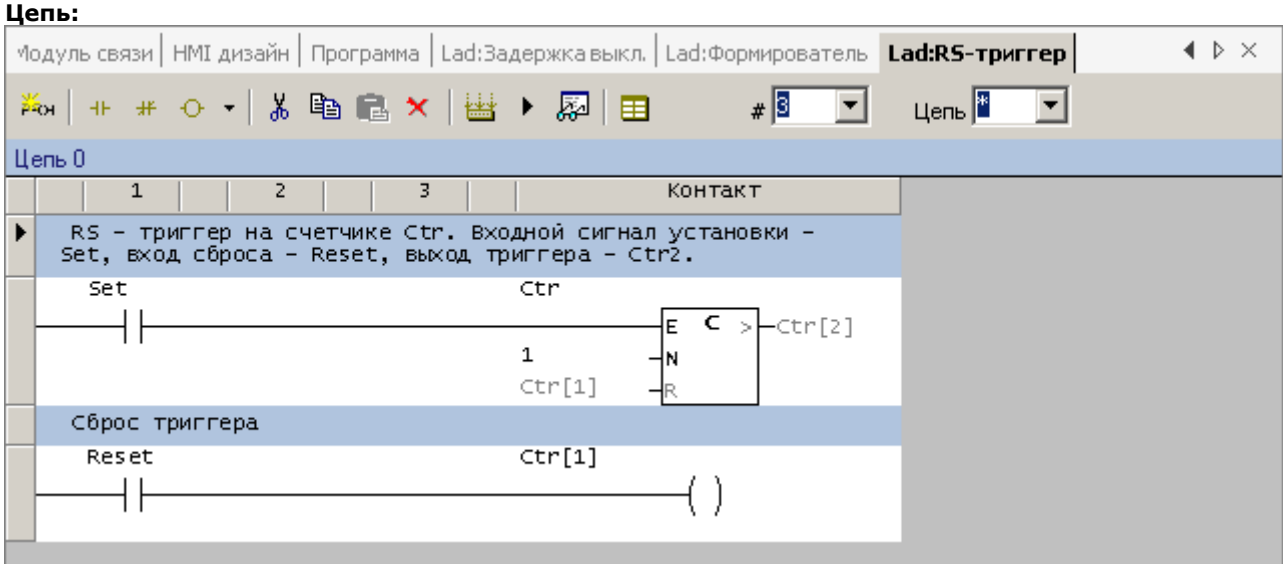
Выходной сигнал Formout переключается в «1» одновременно со включением входного сигнала Sig . Одновременно начинается отсчет задержки выключения на таймере. По окончании задержки Formout переходит в «0». При отключении Sig - переход из «1» в «0» переключения выходного сигнала не происходит.

Временная диаграмма:



3. RS-триггер

Цепь:



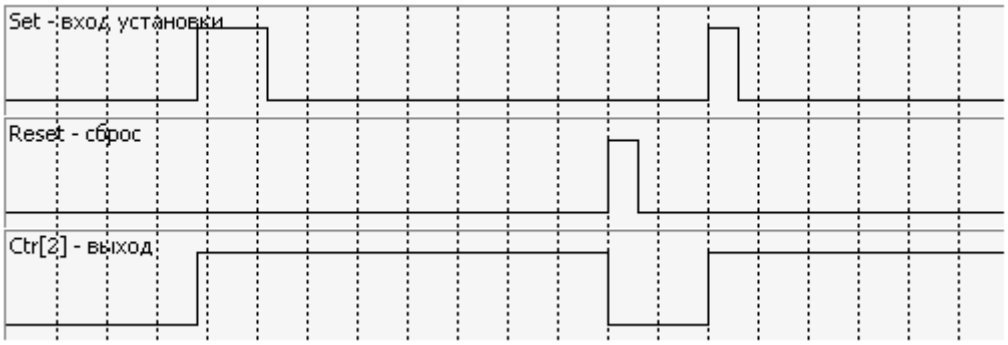
Контакты:

Set	входной сигнал установки триггера;
Reset	входной сигнал сброс триггера.
Ctr	катушка настройки счетчика;
Ctr[1]	выход триггера;
N	параметр счетчика;

Описание работы цепи:

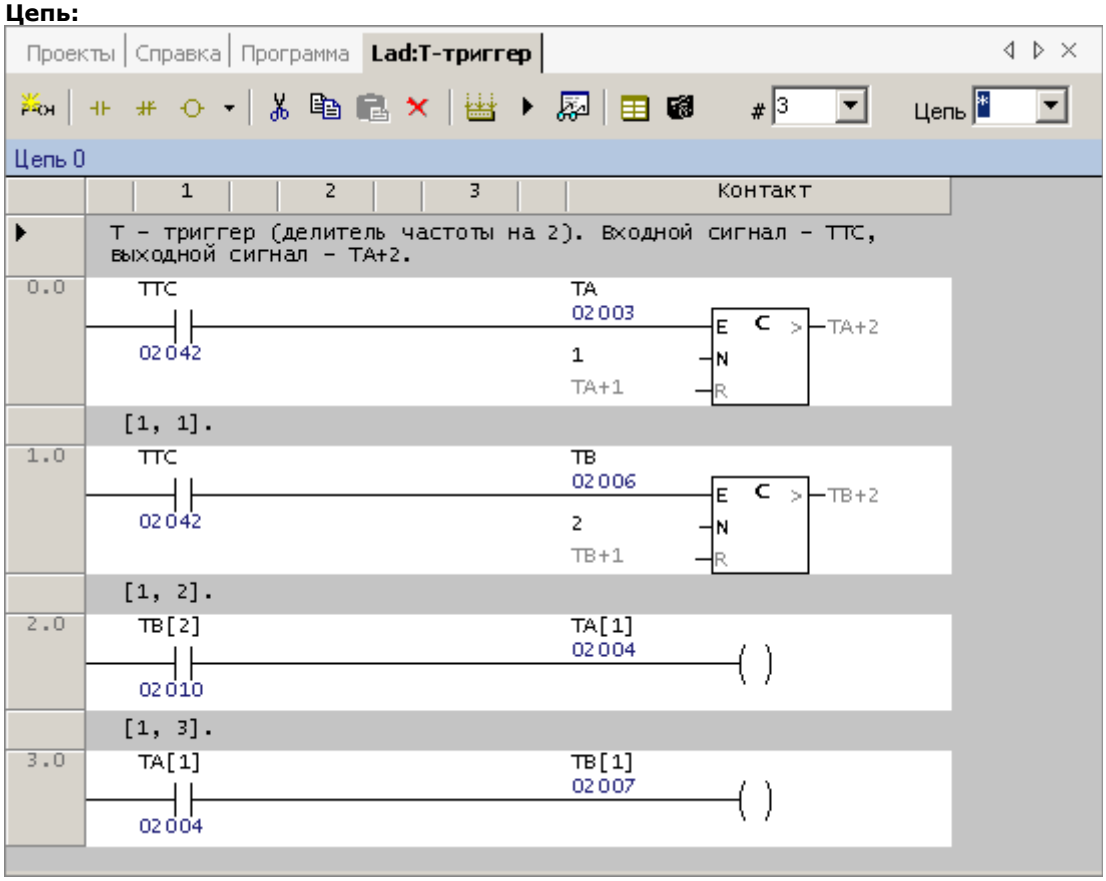
Сигнал установки триггера Set поступает на катушку настройки счетчика Ctr. При Set=1 текущее значение счетчика устанавливается в 1, сигнал окончания счета ">", образующий выход RS-триггера, также в 1. RS-триггер "установлен".
Сигнал Reset поступает на катушку сброса текущего счетчика в 0. При Reset=1 текущее значение счетчика равно 0 и выход окончания ">" также в 0. RS-триггер "сброшен".
Для корректной работы RS-триггера требуется чтобы активным был только один из входов: Set или Reset.

Временная диаграмма:



4. T-триггер

Цепь:



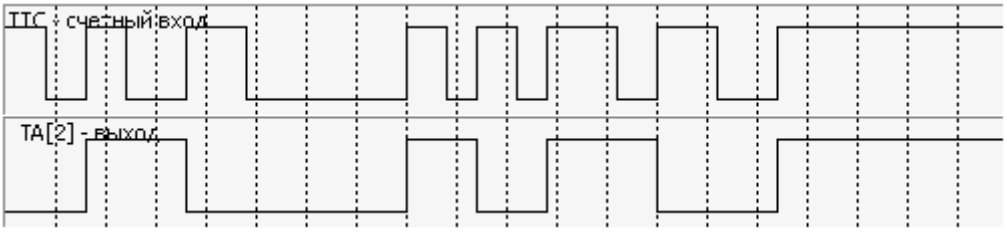
Контакты:

TTC	счетный вход T-триггера;
TA	катушка настройки счетчика TA;
TA[1]	входной сигнал сброс первого счетчика.
TA[2]	выход окончания счета; выход T-триггера;
N=1	параметр первого счетчика;
TB	катушка настройки счетчика TB;
TB[1]	входной сигнал сброс счетчика.
TB[2]	выход окончания счета;
N=2	параметр второго счетчика;

Описание работы цепи:

Счетчик TA считает до первого такта, затем устанавливает сигнал ">", который является выходом T-триггера, в 1.
Счетчик TB считает до двух тактов, затем подает сигнал сброса TB[2] на вход TA[1] счетчика TA. Сигнал сброса TB[2] появляется один раз за два периода входного сигнала TTC и активен лишь в течение одного скана, затем он автоматически снимается. Но в момент активности TB[2] счетчик TA сброшен и не считает.
Таким образом выход счетчика TA[2] работает в режиме деления входной частоты TTC на два.

Временная диаграмма:



5. Пример программы для HMI и NetPlc

Рассмотрим пример простой формы HMI с кнопкой "Пуск", которая посылает сигнал в CPU-165 node=0 на счетный вход Т-триггера, рассмотренный в предыдущем примере. Выходной сигнал триггера принимается элементами HMI "График", образуя простой пример цепи управления.

Требуемые действия:

- группировка входных и выходных данных, пересылка данных из битовой области ПЛК в память для выходных сигналов и обратно - для входных;
- разработка программы NetPlc для обмена данными с CPU-165;
- разработка формы HMI.

Вначале сгруппируем сигналы записи и чтения CPU-165 в отдельные (неперекрывающиеся) области памяти выравненные по байту. Это необходимо чтобы результаты выполнения команд NREAD и NWRITE не портили друг друга, см. Рис 3.5.1.

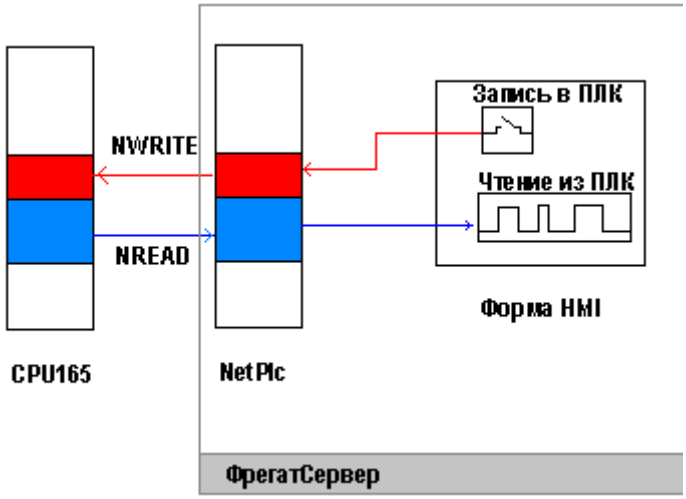


Рис. 5.1 Области обмена на запись и чтение:

- красным цветом выделена область записи из формы в CPU-165 по команде NWRITE, CPU-165 использует ее только для чтения,
- синим - область чтения из CPU-165 в форму по команде NREAD, CPU-165 может использовать область для чтения и записи.

Таблица обмена данными			
Сигнал	**Адрес ПЛК**	**Адрес области обмена**	**Адрес в NetPlc и форме HMI**
Пуск - сигнал от элемента HMI	02042_8	04004 бит 2_8	040042_8
ТА - счетный вход триггера	02003_8	04000 бит 3_8	040003_8

TA[2] - выход триггера	02005_8	04000 бит 5_8	040005_8

Следующие две цепи добавляются к программе CPU-165, соответственно, перед и после блока с T-триггером. Первая цепь копирует слово данных из области обмена в область входных сигналов:

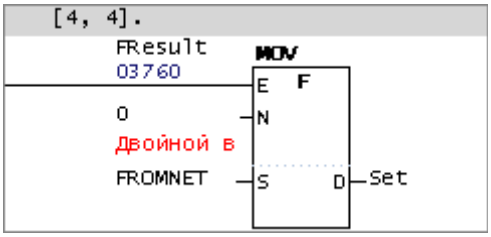


Рис. 5.2 ПЛК: из сети

Вторая цепь копирует выходной результат в область обмена:

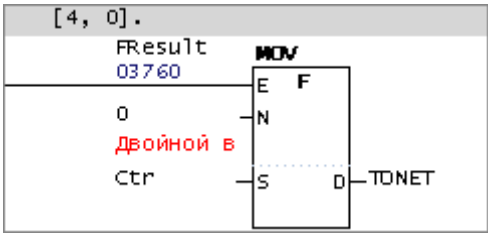


Рис. 5.3 ПЛК: из сети

Программа для NetPlc содержит цепи (Рис. 3.5.4):

- Установка Node=0:
- Команду NWRITE для записи из NetPlc в CPU-165;
- Команду NREAD для чтения из CPU-165 в NetPlc;
- воспомогательные цепи, обеспечивающие выполнение на каждом такте NetPlc.

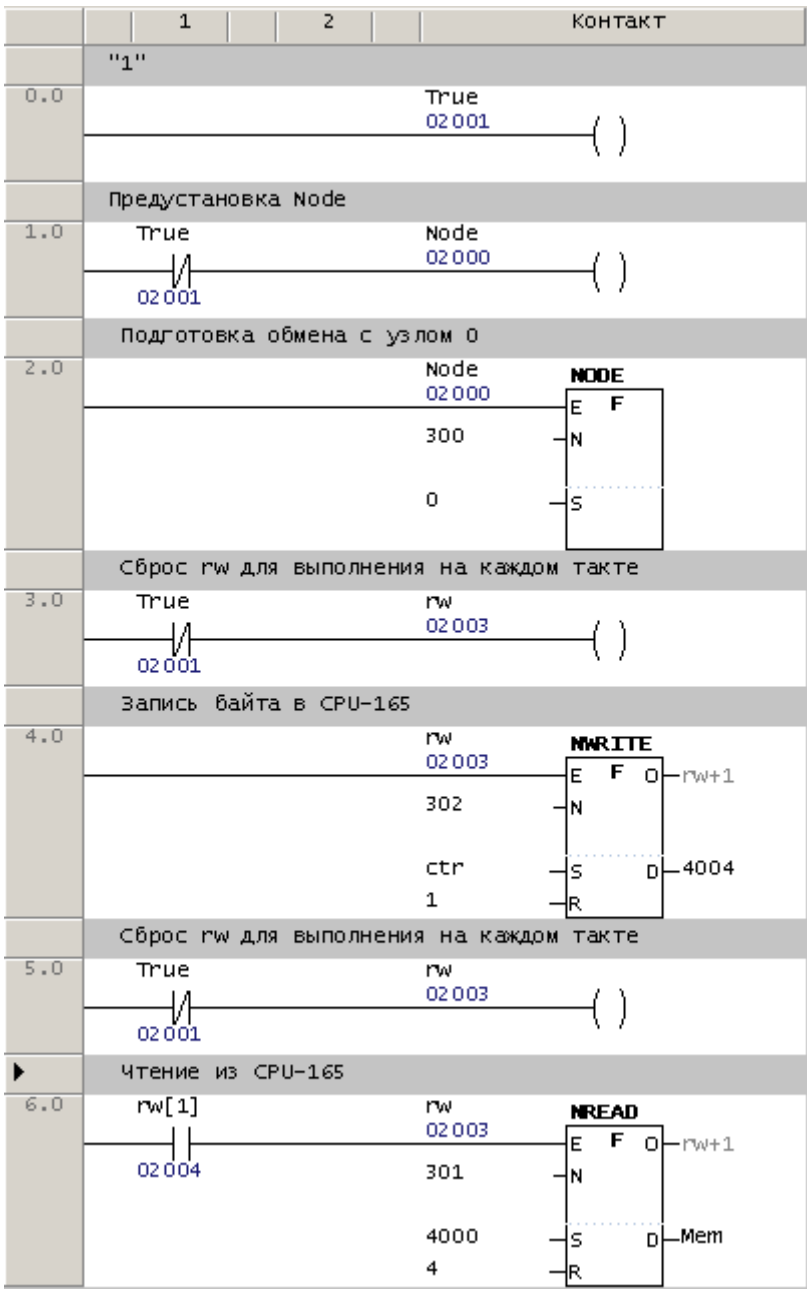


Рис. 5.4 Программа NetPlc

Разработка формы заключается в размещении элементов на форме и задания параметров Рис. 3.5.5.

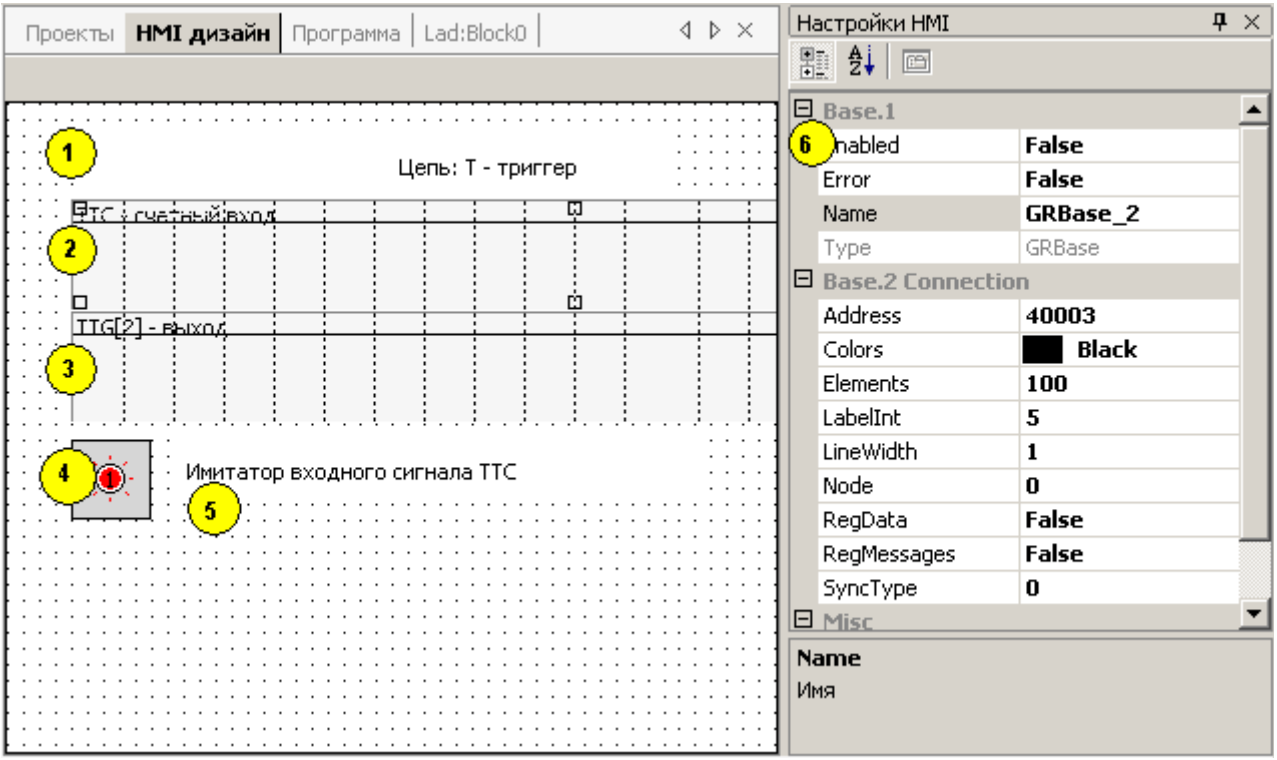


Рис. 5.5 Форма HMI в дизайнере

1. статический элемент "текст";
2. элемент "график" - для входного сигнала;
3. элемент "график" - для вывходного сигнала;
4. дискретный базовый элемент - для формарования входного сигнала триггера;
5. статический элемент "текст";
6. настройки HMI.

Для запуска форму следует сохранить в файл в папку "/netplc", например:

- "netplc/example/пример.hmi"

Затем открыть окно управления ФрегатСервер и внести параметр "HMILoad = netplc/example/" и нажать кнопку "Сохранить изменения". При этом откроются все формы HMI, расположенные в папке "netplc/example/".

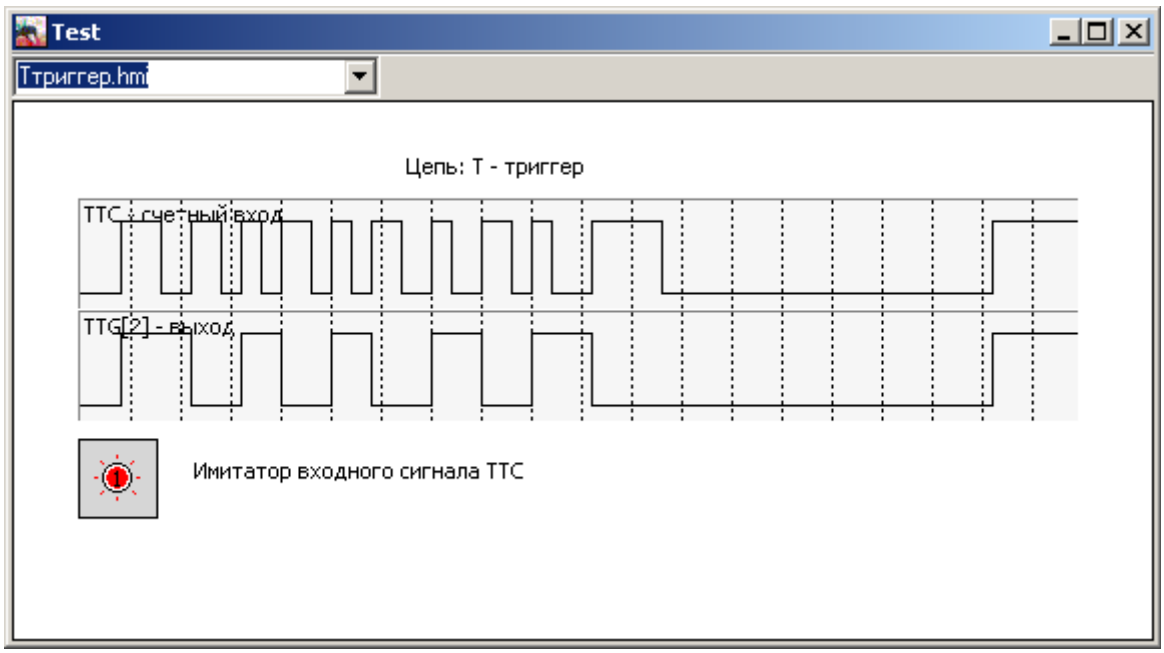


Рис. 5.6 Результат на форме HMI