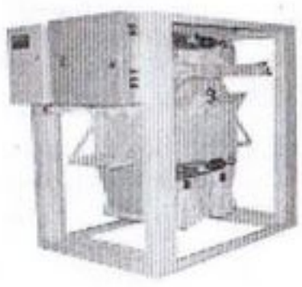
Дозатор в зашивные мешки ДВС-301-70-1

Крановые весы с радиоканалом ВКР-100Р

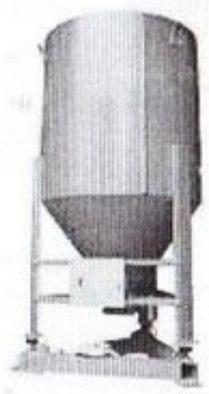
Крановые весы ВКР-100

Бункерные весы СВЕДА ВБА-4-1000

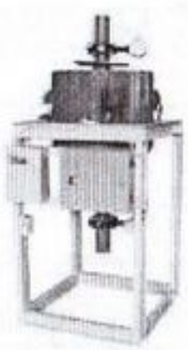


Бункерные весы СВЕДА ВБА-3-150-1-00

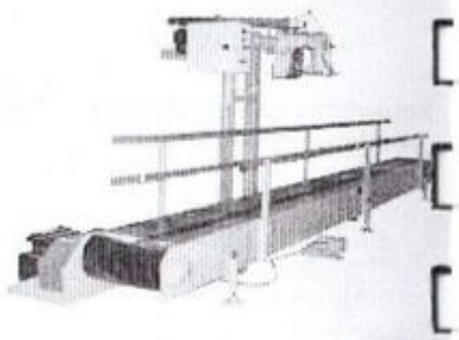
Бункерные весы СВЕДА ВБА-3-300-1-02



Бункерные весы СВЕДА ВБА-3-3000-2



Бункерные весы СВЕДА ВБА-3-300-2



Линия зашивки мещков СВЕДА ЛЗМ

69035, Украина, г. Запорожье, ул. З. Космодемьянской, 3а,
+38(061)233-22-97, (061)213-19-02
E-MAIL: info@sweda.com.ua , http://www.sweda.com.ua

1 ОБЩИЕ УКАЗАНИЯ

1.1 Перед эксплуатацией необходимо:

а) внимательно ознакомиться с эксплуатационной документацией дозатора;

б) паспорт (ПС) должен постоянно находиться с дозатором;

в) в ПС не допускаются подчистки, записи карандашом и смывающимися чернилами;

г) неправильная запись должна быть аккуратно зачеркнута и рядом записана новая, которую заверяет ответственное лицо;

д) после подписи проставляют фамилию и инициалы ответственного лица (вместо подписи допускается проставлять личный штамп исполнителя).

2 ОСНОВНЫЕ СВЕДЕНИЯ

2.1 Дозатор автоматический весовой непрерывного действия ленточный СВЕДА ДВЛ-650

Дата изготовления "15" ноября 2013 г.

Предприятие-изготовитель ООО Научно-производственная фирма "СВЕДА, ЛТД"

Заводской номер 137

3 ОСНОВНЫЕ ТЕХНИЧЕСКИЕ ДАННЫЕ

3.1 Основные технические характеристики дозатора приведены в таблице 1.

Таблица 1

Наименование параметра	Значение
Наибольший предел производительности НПП, т/ч	7,5
Наименьший предел производительности НмПП, т/ч	0,75
Предел допускаемой погрешности дозирования, % от НПП дозатора	±1,0
Потребляемая мощность, не более, ВА	200
Предел допускаемой погрешности дозатора при измерении суммарной массы	±1,0

4 КОМПЛЕКТНОСТЬ

4.1 Комплект поставки дозатора приведен в таблице 2.

Таблица 2

Наименование	Количество
Транспортер	1 шт.
Шкаф управления	1 шт.
Табло-терминал ТВ-330-16	1 шт.
Дозатор автоматический весовой непрерывного действия СВЕДА ДВЛ. Руководство по эксплуатации СВ.205.000 РЭ	1 экз.
Дозатор автоматический весовой непрерывного действия СВЕДА ДВЛ. Паспорт СВ.205.000 ПС	1 экз.
Табло-терминал ТВ-330-16. Руководство по эксплуатации СВ.330.01 РЭ	1 экз.
Мотор-редуктор. BF30Z-04/D05LAV-TF/SP Эксплуатационная документация	1 к-т
Частотный преобразователь. Эксплуатационная документация	1 к-т
Шибер	—

5 СВИДЕТЕЛЬСТВО О ПРИЕМКЕ

5.1 Дозатор автоматический весовой непрерывного действия ленточный СВЕДА ДВЛ- 650 НПП 7,5 т/ч, заводской номер 137 изготовлен и принят в соответствии с обязательными требованиями ГОСТ 30124-94 и признан годным для эксплуатации.

С. Н. С. _____ _____
должность личная подпись расшифровка подписи

2013 11 15
год, месяц, число

МП

6 СВЕДЕНИЯ ОБ УПАКОВЫВАНИИ

6.1 Дозатор автоматический весовой непрерывного действия ленточный СВЕДА ДВЛ-_____, № _____ упакован согласно требованиям, предусмотренным в действующей технической документации.

_____ _____ _____
должность личная подпись расшифровка подписи

_____
год, месяц, число

7 ГАРАНТИИ ИЗГОТОВИТЕЛЯ

7.1 Изготовитель (поставщик) гарантирует соответствие дозатора требованиям технической документации при соблюдении потребителем условий эксплуатации, хранения и монтажа.

7.2 Гарантийный срок - 12 месяцев с момента поставки.

7.3 Дозатор, у которого во время гарантийного срока будет обнаружено несоответствие требованиям технической документации, заменяется другим или ремонтируется изготовителем (поставщиком).

Изготовитель:
ООО Научно-производственная фирма "СВЕДА, ЛТД",
ул. Зои Космодемьянской, 3а,
г. Запорожье, Украина
60935

тел./факс (+380612) 332297;
e-mail: sweda@sweda.com.ua.

ПРИЛОЖЕНИЕ Г

ОПИСАНИЕ ПРОТОКОЛОВ ОБМЕНА.

ПРОТОКОЛ "MODBUS".

Контроллеры ПВ-310 могут быть настроены для работы в стандартных сетях Modbus с использованием одного из режимов передачи: ASCII или RTU. Все контроллеры ПВ-310 являются slave-устройствами и не могут быть использованы для формирования master-запросов. Пользователи выбирают нужный режим вместе с параметрами серийного порта связи (скорость в бодах, четность и т.д.) при конфигурировании каждого контроллера.

Выбор режимов ASCII или RTU имеет отношение только к стандартным сетям Modbus. Он определяет содержание битов полей сообщений, последовательно передаваемых в указанных сетях. Он определяет, как информация будет паковаться в поля сообщений, а затем декодироваться.

Режим ASCII

Если контроллеры настроены на работу в сети Modbus с использованием режима ASCII (Американский стандартный код по обмену информацией), каждый 8-битный байт в сообщении посылается как два знака ASCII.

Формат каждого байта в режиме ASCII следующий:

Система кодировки:

Шестнадцатеричная, знаки ASCII 0-9, A-F. Один шестнадцатеричный знак содержится в каждом знаке ASCII сообщения.

Бит в байте:

1 стартовый бит

7 бит данных, бит с наименьшим значением высылается первым

1 бит на проверку на четность/ нечетность, нет бит на отсутствие четности

1 стоповый бит если четность используется, 2 бит при отсутствии четности

Поле проверки ошибки:

Проверка продольного резервирования (LRC)

Режим RTU

Если контроллеры настроены на работу в сети Modbus в режиме RTU (Удаленный терминал), то каждый 8-битный байт содержит два 4-битных шестнадцатеричных знака. Главным преимуществом данного

режима является то, что его бóльшая плотность знаков обеспечивает бóльшую по сравнению с ASCII пропускную способность при одинаковой скорости в бодах. Каждое сообщение должно передаваться непрерывным потоком.

Формат каждого байта в режиме RTU следующий:

Система кодировки:

8-битная двоичная, шестнадцатеричная 0-9, A-F.

Два шестнадцатеричных знака содержатся в каждом 8-битном поле сообщения.

Бит в байте:

1 стартовый бит

8 бит данных, бит с наименьшим значением высылается первым

1 бит на проверку на четность/ нечетность, нет бит на отсутствие четности

1 стоповый бит если четность используется, 2 бит при отсутствии четности

Поле проверки ошибки:

Проверка циклического резервирования (CRC).

ASCII фрейминг

В режиме ASCII сообщения начинаются со знака двоеточия (:) (ASCII 3A hex2), а заканчиваются парой «возврат каретки – перевод строки» (CLRF) (ASCII 0D и 0A hex).

Разрешенные для передачи знаки для всех остальных полей – шестнадцатеричные 0-9, A-F. Связанные в сеть устройства осуществляют непрерывный мониторинг сетевой шины для знака «двоеточие». При получении одного знака каждое устройство декодирует следующее поле (адресное поле), чтобы выяснить, является ли оно адресуемым устройством.

Между знаками в сообщении могут быть интервалы до 1 секунды. При появлении большего интервала, получающее устройство воспринимает это как ошибку. Фрейм типичного сообщения приведен ниже.

Таблица 1. Фрейм сообщения ASCII

Старт	Адрес	Функция	Данные	Проверка LRC (продольный контроль по избыточности)	Конец
1 знак	2 знака	2 знака	n знаков	2 знака	2 знака CRLF

RTU фрейминг

В режиме RTU сообщения начинаются с паузы, соответствующей как минимум 3,5 знакам (в приведенном ниже рисунке – T1-T2-T3-T4). Передаваемое затем первое поле является адресом устройства.

Вслед за последним переданным знаком, аналогичный интервал, соответствующий как минимум 3,5 знакам, означает конец сообщения. После такого интервала может начинаться новое сообщение.

Полный фрейм сообщения должен передаваться непрерывным потоком. При возникновении паузы дольше 1,5 знака до окончания фрейма, принимающее устройство информирует о неоконченном сообщении и предполагает, что следующий байт будет адресным полем нового сообщения.

Аналогично, если новое сообщение начинается раньше, чем 3,5-знаковый интервал после предыдущего сообщения, то принимающее устройство воспринимает его как продолжение предыдущего сообщения. Этим самым вызывается ошибка, т.к. значение заключительного поля CRC будет недействительным для комбинированных сообщений. Фрейм типичного сообщения приведен ниже.

Таблица 2. Фрейм сообщения RTU

Старт	Адрес	Функция	Данные	Проверка LRC	Конец
T1-T2-T3-T4	8 бит	8 бит	n x 8 бит	16 бит	T1-T2-T3-T4

Как осуществляется последовательная передача знаков

При передаче сообщений в стандартных последовательных сетях Modbus, каждый знак или байт передается в следующем порядке (слева направо):

Бит с наименьшим значением (LSB) Бит с наибольшим значением (MSB)

При фрейминге знаков ASCII последовательность бит такова:

С проверкой четности

Start 1 2 3 4 5 6 7 Par Stop

Без проверки четности

Start 1 2 3 4 5 6 7 Stop Stop

При фрейминге знаков RTU последовательность бит такова:

С проверкой четности

Start 1 2 3 4 5 6 7 8 Par Stop

Без проверки четности

Start 1 2 3 4 5 6 7 8 Stop Stop

Методы проверки ошибок

Стандартные последовательные сети Modbus используют два типа проверки ошибок. Проверка четности (четность/нечетность) может дополнительно применяться к каждому знаку. Проверка фрейма (LRC или CRC) применяется ко всему сообщению. Как проверка знака, так и проверка фрейма сообщения генерируются в устройстве master и применяются к содержанию сообщения до его передачи. Устройство slave проверяет каждый знак и фрейм всего сообщения во время приема.

Master конфигурируется пользователем для ожидания в течение предварительно определенного времени простоя перед отменой действия. Такой интервал настраивается довольно продолжительным, чтобы дать возможность slave-устройству ответить нормально. Если slave обнаруживает ошибку передачи, то действий по сообщениям предприниматься не будет. Slave не будет создавать ответа для master. Таким образом, время простоя истечет, и это позволит программе master-устройства обработать ошибку. Обратите внимание, что сообщение, адресуемое несуществующему устройству slave, также вызовет простой.

Функции данных и контроля

- Форматы функций Modbus
- Краткий обзор кодов функций

Форматы функций Modbus

Как выражаются цифровые значения

Если не оговорено иное, цифровые значения (такие как адреса, коды или информация) выражаются как десятичные значения в тексте данного раздела. В полях сообщений они выражаются как шестнадцатеричные значения.

Адреса данных в сообщениях Modbus

Все адреса данных в сообщениях Modbus обращаются к нулю. Первое появление элемента данных адресуется как элемент номер ноль. Например:

- Обмотка (обмотка – термин, применяемый в описании протокола Modbus, в данном описании следует понимать как выход), известная как «обмотка 1» в программируемом контроллере, адресуется как обмотка 0000 в поле адресных данных сообщения Modbus.
- Обмотка 127 десятичная адресуется как обмотка 007E hex (126 десятичная).
- Регистр хранения информации 40001 адресуется как регистр 0000 в поле адресных данных сообщения. Поле кода функции уже определяет операцию «регистр хранения информации». Таким образом, подразумевается ссылка 4XXXX.
- Регистр хранения информации 40108 адресуется как регистр 006B hex (107 десятичный).

Содержание поля в сообщениях Modbus

Таблица 3 показывает пример запроса Modbus. Таблица 4 – пример стандартного ответа. Оба примера показывают содержание поля в шестнадцатеричных цифрах, а также как можно создать фрейм сообщения в режимах ASCII и RTU.

Запрос master-устройства является запросом для адреса 06 устройства slave. Сообщение запрашивает данные из указанных регистров хранения информации, от 40108 до 40110. Обратите внимание, что сообщение определяет начальный адрес регистра как 0107 (006B hex).

Ответ slave повторяет код функции, что означает нормальный ответ. Поле Подсчета Байт определяет, сколько 8-битовых элементов данных возвращено.

Показывается, скольким 8-битовым байтам следовать в данных, ASCII или RTU. В случае с ASCII, значение равно половине фактических знаков ASCII, имеющихся в данных. В ASCII каждое 4-битовое шестнадцатеричное значение требует один знак ASCII, таким образом, два знака ASCII должны следовать в сообщении, чтобы содержать каждый 8-битовый элемент данных.

Например, значение 63 hex послано как один 8-битовый байт в режиме RTU (01100011). Это же значение, отправленное в режиме ASCII, требует двух байт, "6" (00110110) и "3" (00110011). Поле подсчета Байт подсчитывает эти данные как один 8-битовый элемент, независимо от метода фрейминга знаков (ASCII или RTU).

Как использовать Поле Подсчета: При построении ответов в буферах, используйте значение Подсчета Байт, равное 8-битовым байт данных вашего сообщения. Таблица 4 показывает, как поле подсчета данных используется в типичном ответе.

Таблица 3. **Запрос master с фреймингом ASCII/RTU**

ЗАПРОС			
Название поля	Пример (Hex)	Знаки ASCII	8-битовое поле RTU
Заголовок		: (двоеточие)	нет
Адрес slave	06	0 6	0000 0110
Функция	03	0 3	0000 0011
Стартовый адрес Hi	00	0 0	0000 0000
Стартовый адрес Lo	6B	6 B	0110 1011
Кол-во регистров Hi	00	0 0	0000 0000
Кол-во регистров Lo	03	0 3	0000 0011
Проверка на ошибку		LRC (2 знака)	CRC (16 бит)
Трейлер		CRLF	Нет
	Всего байт	17	8

Таблица 4. **Ответ slave с фреймингом ASCII/ RTU**

ОТВЕТ			
Название поля	Пример (Hex)	Знаки ASCII	8-битовое поле RTU
Заголовок		: (двоеточие)	нет
Адрес slave	06	0 6	0000 0110
Функция	03	0 3	0000 0011
Подсчет байт	06	0 6	0000 0110
Данные Hi	02	0 2	0000 0010
Данные Lo	2B	2 B	0010 1011
Данные Hi	00	0 0	0000 0000
Данные Lo	00	0 0	0000 0000
Данные Hi	00	0 0	0000 0000
Данные Lo	63	6 3	0110 0011
Проверка на ошибку		LRC (2 знака)	CRC (16 бит)
Трейлер		CRLF	Нет
	Всего байт	23	11

Функциональные коды, поддерживаемые контроллерами ПВ-310.

Приводимый ниже в таблице 5 список содержит, поддерживаемые контроллерами Modicon. Коды даны в десятичных цифрах.

Таблица 5 **Функциональные коды**

Код	Имя
01	Считать информацию о состоянии обмотки
02	Считать информацию о состоянии на входе
03	Считать значения регистров временного хранения информации
05	Форсировать единичную обмотку
06	Установить единичный регистр
15	Форсировать множественные обмотки
16	Установить множественные регистры
20	Прочитать общую ссылку
21	Записать общую ссылку
25	Терминальная команда
27	Терминальная команда
26	Перезапуск устройства

Команды, реализованные в протоколе Modbus

01 Считать информацию о состоянии обмотки

Описание: Считывает состояние ON/OFF дискретных выходов (0X ссылки, обмотки) в slave. Широковещательная передача не поддерживается.

В контроллере ПВ-310, в зависимости от конфигурации, может быть 4 или 8 выходов, т.е. их адреса находятся в диапазоне 0...7. Если запрашиваются состояния выходов за пределами диапазона, то возвращается сообщение об ошибке. Фактически читается переменная uprdat, которая хранит состояние выходов.

02 Считать информацию о состоянии на входе

Описание: Считывает состояние ON/OFF дискретных входов (ссылки 1X) slave. Широковещательная передача не поддерживается.

В контроллере не более 8-ми входов, т.е. их адреса находятся в диапазоне 0...7. Если запрашиваются состояния входов за пределами диапазона, то возвращается сообщение об ошибке.

03 Считывать значения регистров временного хранения информации

Описание: Считывает двоичные значения регистров временного хранения информации (ссылки 4X) устройства slave. Широковещательная передача не поддерживается.

Данная команда позволяет читать параметры, приведенные в таблице 6.

Таблица 6

№ рег.	Параметр	Ед. изм.	Длина, байты	Доступ	Примечание
0	Тип устройства		4	Чт	
1	Номер версии ПО		4	Чт	
2	Итог общий	Тонна	4	Чт	
3	Итог общий	Грамм	4	Чт	
4	Итог сменный	Тонна	4	Чт	
5	Итог сменный	Грамм	4	Чт	
6	Тип устройства, старшее слово		2	Чт	
7	Тип устройства, младшее слово		2	Чт	
8	Номер версии ПО, старшее слово		2	Чт	
9	Номер версии ПО, младшее слово		2	Чт	
10	Общий итог тонны, старшее слово		2	Чт	
11	Общий итог тонны, младшее слово		2	Чт	
12	Общий итог граммы, старшее слово		2	Чт	
13	Общий итог граммы, младшее слово		2	Чт	
14	Сменный итог тонны, старшее слово		2	Чт	
15	Сменный итог тонны, младшее слово		2	Чт	
16	Сменный итог граммы, старшее слово		2	Чт	
17	Сменный итог граммы, младшее слово		2	Чт	
18	Текущее значение веса	Грамм	4	Чт	
19	Текущая производительность	0.01 т/час	4	Чт	
20	Плановая производительность	0.01 т/час	4	Чт Зп	
21	Погонный вес ленты	0.01 кг/м,	4	Чт	
22	Порог погонного веса ленты	0.01 кг/м	4	Чт Зп	
23	Максимальная производительность	Процен	4	Чт	

	при регулировке по отгруженному весу	т			
24	Доза	0.01 т	4	Чт Зп	
25	Отгружено от заданной дозы	0.01 т	4	Чт	
26	Длина ленты	см	4	Чт	
27	Коэффициент коррекции	0.001 раза	2	Чт Зп	
28	Коэффициент усиления при регулировке	0.01 раза	2	Чт Зп	
29	Коэффициент интегрирования ЦАП	0.01 раза	2	Чт Зп	
30	Вес на 1 импульс внешнего счетчика	Тонна	2	Чт Зп	
31	Введенная скорость	см/сек	2	Чт Зп	
32	Лимит времени выдачи предельного кода ЦАП для достижения заданной производительности	0.1 сек	2	Чт Зп	
33	Лимит времени выдачи предельного кода ЦАП для достижения заданной производительности	0.1 сек	2	Чт Зп	
34	Состояние блока	Число	1	Чт	0 – рабочий режим и режим меню 1 – измерение длины и веса ленты 2 – измерение веса ленты 3 – калибровка 4 – ожидание 5 – тест ЦАП 6 – тест ввода-вывода 7 – тест сохранения 8 – дистанционное измерение веса ленты 9 – состояние ошибки 10 – большое кол-во

					импульсов ДСЛ
35	Состояние индикации	Число	1	Чт	0 – норма 2 – нет готовности оборудования (запрет1) 3 – нет готовности оборудования (запрет2) 4 – авария преобразователя 5 – смещение ленты 7 – авария АЦП 8 – превышение тары 9 – произв-ть не достигла плановой 10 – произв-ть превышает плановую
36	Байт дистанционного управления	Число	1	Чт Зп	0x40 – разрешение пуска/стоп преобразователя (если бит=1, то пуск) 0x20 - пуск измерения веса ленты.
38	Нечувствительность	Число	1	Чт Зп	
39	Задание скорости	Число	1	Чт Зп	0- измеряется, 1-вводится
40	Регулирование по производительности или по весу	Число	1	Чт Зп	0 – произв-ть 1 – отгр. вес
41	Скорость по RS485	Число	1	Чт Зп	0 – 1200, 1 – 2400, 2 – 4800, 3 – 9600, 4 – 19200, 5 – 38400

42	Положение запятой в производительности	Число	1	Чт Зп	Кол-во знаков после запятой при индикации производительности
43	Пуск дозирования	Число	1*	Чт Зп	1 – пуск 0 - стоп
44	Текущее значение веса, старшее слово		2	Чт	
45	Текущее значение веса, младшее слово		2	Чт	
46	Текущая производительность, старшее слово	0.01 т/час	2	Чт	
47	Текущая производительность, младшее слово	0.01 т/час	2	Чт	
48	Плановая производительность, старшее слово	0.01 т/час	2	Чт Зп	
49	Плановая производительность, младшее слово	0.01 т/час	2	Чт Зп	
50	Погонный вес ленты, старшее слово	0.01 кг/м	2	Чт	
51	Погонный вес ленты, младшее слово	0.01 кг/м	2	Чт	
52	Порог погонного веса ленты, старшее слово	0.01 кг/м	2	Чт Зп	
53	Порог погонного веса ленты, младшее слово	0.01 кг/м	2	Чт Зп	
54	Максимальная производительность при регулировании по отгруженному весу, старшее слово	0.01 кг/м	2	Чт Зп	
55	Максимальная производительность при регулировании по отгруженному весу, младшее слово	0.01 кг/м	2	Чт Зп	
56	Доза, старшее слово	0.01 т	2	Чт Зп	
57	Доза, младшее слово	0.01 т	2	Чт Зп	
58	Отгружено от заданной дозы, старшее слово	0.01 т	2	Чт Зп	
59	Отгружено от заданной дозы, младшее слово	0.01 т	2	Чт Зп	
60	Текущее значение скорости, если нет регулирования по производительности	% макс	2	Чт Зп	
61	Начальное значение скорости	% макс	2	Чт Зп	

62	Длина ленты, старшее слово	см	2	Чт	
63	Длина ленты, младшее слово	см	2	Чт	

* - однобайтные величины читаются и записываются как двухбайтные, но их значение не должно превосходить 256.

** - при установленном параметре (кол-во знаков после запятой) выполнить умножение на соответствующий коэффициент.

05 Форсировать единичную обмотку

Описание: Форсирует единичную обмотку (ссылка 0X) на ON или OFF. При широковещательном сообщении, данная функция форсирует эту же ссылку обмотки во всех slave.

Примечание: Данная функция заменит защиту памяти контроллера и блокированное состояние обмотки. Форсированное состояние останется в силе, пока логика контроллера не разрешит обмотку. Обмотка останется форсированной, если она не запрограммирована в логике контроллера.

Адрес не более 7-ми. Если адрес выхода больше 7, то возвращается сообщение об ошибке.

06 Установить единичный регистр

Описание: Устанавливает значение в единичном регистре хранения информации (ссылка 4X). При широковещательном сообщении, данная функция форсирует эту же ссылку регистра во всех присоединенных устройствах slave.

Примечание: Данная функция заменит защиту памяти контроллера. Установленный уровень останется в силе, пока логика контроллера не разрешит содержание регистра. Значение регистра останется без изменений, если оно не запрограммировано в логике контроллера.

Смотри таблицу 5.

15 (0F Hex) Форсировать множественные обмотки

Описание: Форсирует каждую обмотку (ссылка 0X) в последовательности на ON или OFF. При широковещательном сообщении, данная функция форсирует эти же ссылки обмоток во всех

присоединенных устройствах slave.

Примечание: Данная функция заменит защиту памяти контроллера и блокированное состояние обмотки. Форсированное состояние останется в силе, пока логика контроллера не разрешит обмотку. Обмотка останется форсированной, если она не запрограммирована в логике контроллера.

Позволяет установить значения сразу нескольких выходов. Максимальный адрес не должен превосходить 7, в противном случае возвращается ошибка.

16 (10 Hex) Предварительно установить множественные регистры

Описание: Задает значения последовательности регистров хранения информации (ссылки 4X). При широковещании, функция устанавливает те же ссылки регистров во всех slave.

Примечание: Данная функция заменит защиту памяти контроллера. Установленный уровень останется в силе, пока логика контроллера не разрешит содержание регистра. Значение регистра останется без изменений, если оно не запрограммировано в логике контроллера.

См. таблицу 5. Позволяет одной командой записать несколько регистров.

20 (14 Hex) Прочитать общую ссылку

Описание: Возвращает содержимое регистров в ссылках файла (6XXXXX) расширенной памяти. Широковещательная передача не поддерживается.

Функция может читать множественные группы ссылок. Группы могут быть отдельными (не смежными), но ссылки внутри каждой группы должны быть последовательными.

В протоколе MODBUS команда 20 позволяет читать файлы расширенной памяти контроллеров фирмы MODICON, при этом указывается номер файла расширенной памяти. В нашем случае номер файла расширенной памяти трактуется как тип памяти.

№файла=1, читается внутреннее ОЗУ,

№файла=2, читается внешнее ОЗУ,

№файла=3, читается флешь.
Единицей памяти в контроллере ПВ-310 является байт. Для совместимости с протоколом MODBUS каждый байт передается как 16-битное слово, в котором старший байт равен 0.

Команда 16 позволяет читать несколько "разбросанных" в памяти переменных в одном пакете.

21 (15 Hex) Записать общую ссылку

Описание: Производит запись всех регистров в ссылки файла расширенной памяти (6XXXXX).

Широковещательная передача не поддерживается.

Функция способна записывать множественные группы ссылок. Группы могут быть отдельными (не смежными), но ссылки внутри каждой группы должны быть последовательными.

В протоколе MODBUS команда 21 позволяет производить запись в файл расширенной памяти.

Для контроллера ПВ-310 номер файла расширенной памяти трактуется аналогично команде 20.

25 (19 Hex) Терминальная команда

Описание: Позволяет передавать код нажатой кнопки и получать в ответ состояние индикатора Широковещательная передача поддерживается. Формат смотри в таблицах 7 и 8.

Коды кнопок :
01 – Меню/Ввод
02 – Режим/Выбор
03 – Тара/ +1
04 – Контр/Выход

Таблица 7

ЗАПРОС			
Название поля	Пример (Hex)	Знаки ASCII	8-битовое поле RTU
Заголовок		: (двоеточие)	нет
Адрес slave	06	0 6	0000 0110
Функция	19	1 9	0001 1001
Код кнопки	01	0 1	0000 0001
Проверка на ошибку		LRC (2 знака)	CRC (16 бит)
Трейлер		CRLF	Нет
Всего байт		11	5

Таблица 8

ОТВЕТ			
Название поля	Пример (Hex)	Знаки ASCII	8-битовое поле RTU
Заголовок		: (двоеточие)	нет
Адрес slave	06	0 6	0000 0110
Функция	19	1 9	0001 1001
Подсчет байт	11	1 1	0001 0001
Положение курсора	02	0 2	0000 0010
Состояние 1–го символа индикатора	2B	2 B	0010 1011
Состояние 2–го символа индикатора	2B	2 B	0010 1011
Состояние 3–го символа индикатора	2B	2 B	0010 1011
Состояние 4–го символа индикатора	2B	2 B	0010 1011
Состояние 5–го символа индикатора	2B	2 B	0010 1011
Состояние 6–го символа индикатора	2B	2 B	0010 1011
Состояние 7–го символа индикатора	2B	2 B	0010 1011
Состояние 8–го символа индикатора	2B	2 B	0010 1011
Состояние 9–го символа индикатора	2B	2 B	0010 1011
Состояние 10–го символа индикатора	2B	2 B	0010 1011
Состояние 11–го символа индикатора	2B	2 B	0010 1011
Состояние 12–го символа индикатора	2B	2 B	0010 1011
Проверка на ошибку		LRC (2 знака)	CRC(16 бит)
Трейлер		CRLF	Нет
Всего байт		37	18

27 (1B Hex) Терминальная команда

Описание: Позволяет передавать код нажатой кнопки и получать в ответ состояние индикатора. Широковещательная передача не поддерживается.

Формат аналогичен команде 25.

26 (1A Hex) Перезагрузка

Описание: Позволяет выполнить перезапуск устройства. Формат смотри в таблице 9.

Таблица 9

ЗАПРОС			
Название поля	Пример (Hex)	Знаки ASCII	8-битовое поле RTU
Заголовок		: (двоеточие)	нет
Адрес slave	06	0 6	0000 0110
Функция	1A	1 A	0001 1010
Проверка на ошибку		LRC (2 знака)	CRC (16 бит)
Трейлер		CRLF	Нет
	Всего байт	9	4

Генерация LRC/CRC

- Запуск LRC
- Запуск CRC

Генерация LRC

Поле проверки продольного резервирования (LRC) является однобайтовым полем, содержащим 8-мибитовое бинарное значение. Значение LRC вычисляется передающим устройством, которое прибавляется к сообщению LRC. Принимающее устройство снова вычисляет LRC во время приема сообщения и сравнивает вычисленное значение с полученым. При несовпадении этих двух значений определяется ошибка.

Значение LRC вычисляется путем сложения последовательных 8-ми битовых байтов в сообщении, отбрасывая все цифры переноса, а затем дополнительные коды результата. LRC представляет собой 8-ми битовое поле, следовательно, каждое новое прибавление знака, которое приведет к результату, превышающему 255 десятичных значений просто «перекрывает» значение поля через ноль. Т.к. девятый бит отсутствует, цифра переноса автоматически отбрасывается.

Для вычисления LRC необходимо:

1. Сложить все байты в сообщении, за исключением начального двоеточия и окончания CRLF в 8-ми битовое поле, исключая перенос.
2. Вычесть получившееся значение поля из FF hex (все 1) для получения дополнения до единицы.
3. Прибавить единицу для получения дополнения до двух.

Помещение LRC в сообщение

При передаче 8-ми битовой LRC (2 символа ASCII) сначала происходит передача знаков старшего, затем младшего разряда. Например, при значении LRC равном 61 hex (0110 0001):

Двое-точие	Ад-рес	Функ-ция	Кол-во данных	Данные	Данные	Данные	Дан-ные	LRC Hi	LRC Lo	CR	LF
								6	1		

Рис. 1. Последовательность символов LRC

Пример:

Ниже приведен пример функции языка С, осуществляющий запуск LRC. Функция использует два аргумента:

unsigned char *auchMsg - указатель на буфер сообщения, содержащий двоичные данные, используемые для запуска LRC

unsigned short usDataLen - количество байт в буфере сообщения.

Функция возвращает LRC как тип unsigned char.

Функция запуска LRC

```
static unsigned char LRC(auchMsg, usDataLen)
unsigned char *auchMsg;   /* message to calculate LRC upon  */
unsigned short usDataLen;   /* quantity of bytes in message  */

{ unsigned char uchLRC = 0 ;   /* LRC char initialized   */
 while (usDataLen—)   /* pass through message buffer  */
 uchLRC += *auchMsg++ ;   /* add buffer byte without carry  */
 return ((unsigned char)(–((char)uchLRC))) ;   /* return twos complement  */
 }
```

Генерация CRC

Поле проверки циклического резервирования (CRC) является двухбайтовым полем, содержащим 16-ти битовое двоичное значение. Значение CRC вычисляется передающим устройством, которое прибавляет CRC к сообщению. Приемное устройство производит подсчет CRC во время принятия сообщения, и сравнивает вычисленное значение с полученным. При несовпадении значений определяется ошибка.

Генерация CRC начинается с предварительной установки значений 16-ти битового регистра в единицы. Затем начинается процесс прибавления последовательных 8-мибитовых байтов сообщения к содержимому регистра. Только 8 бит данных в каждом знаке используются для вычисления CRC. Стартовые и стоповые биты, разряд четности не используются.

При подсчете CRC каждый 8-мибитовый знак является исключающим OR по отношению к содержанию регистра. Затем результат сдвигается в направлении младших разрядов (LSB), а в самый старший (MSB) ставится ноль. LSB выделяется и сравнивается. Если LSB равнялся единице, регистр является исключающим OR с заданным установленным значением. Если LSB равнялся нулю, исключающее OR отсутствует.

Данный процесс продолжается, пока не будут выполнены все 8 сдвигов. После последнего (8-го) сдвига, следующие 8-мибитовые знаки являются исключающими OR по отношению к текущему значению регистра, и следующие восемь сдвигов происходят таким же образом, как было описано выше. Конечное содержание регистра, после сложения всех знаков сообщения, и является значением CRC.

Процедура генерации CRC следующая:

1) Загрузите 16-ти битовый регистр FFFF (все 1). Назовем это регистром CRC.

2) Исключающее OR первый 8-ми битовый байт сообщения с битом младшего разряда 16-битового разряда регистра CRC. Поместить результат в регистр CRC.

3) Переместите регистр CRC на один бит вправо (в направлении LSB), заполняя MSB нулями.

Выделите и проверьте LSB

4) (Если LSB имел значение 0): повторите пункт 3 (следующий сдвиг);

(Если LSB имел значение 1): исключающее OR регистра CRC припомощи полиномиального значения A001 hex (1010 0000 0000 0001).

5) Повторяйте пункты 3 и 4 до тех пор, пока не будут выполнены 8 сдвигов регистра. После этого происходит обработка полного 8-мибитового байта.

6) Повторите пункты 2 – 5 для следующего 8-ми битового байта в сообщении.

7) Полученное содержание регистра является значением CRC.

8) При помещении CRC в сообщение, его младший и старший байт должны быть переставлены в соответствии с данным выше описанием.

Помещение CRC в сообщение:

При передаче 16-ти бит CRC (два 8-ми битовых байта), первым происходит передача младшего байта, затем старшего байта. Например, если значение CRC равно 1241 hex (0001 0010 0100 0001):

Двоеточие	Адрес	Функция	Кол-во данных	Данные	Данные	Данные	Данные	LRC Hi	LRC Lo
								41	12

Рис. 2 Последовательность байтов CRC

Пример:

На последующих страницах приведен пример функции языка С, выполняющей генерацию CRC. Все возможные значения CRC вводятся в две таблицы, которые нумеруются как приращение функции через буфер сообщения. Одна таблица содержит все возможные 256 значения старшего байта 16-ти битового поля CRC, другая таблица содержит все значения младшего байта.

Данная нумерация CRC позволяет быстрее определить CRC в сравнении с тем временем, которое было бы затрачено на вычисление значения CRC каждого нового знака из буфера сообщения.

Таким образом, значение CRC после выполнения функции, можно сразу же помещать в сообщение для передачи.

Функция использует два аргумента:

unsigned char *puchMsg; - указатель на буфер сообщения, содержащий двоичные данные, используемые для запуска CRC;

unsigned short usDataLen; - количество байт в буфере сообщения.

Функция возвращает CRC как тип unsigned short.

Функция запуска CRC

```
unsigned short CRC16(puchMsg, usDataLen)

unsigned shar *puchMsg;        /* message to calculate CRC upon  */
unsigned short usDataLen;      /* quantity of bytes in message  */

{
unsigned char uchCRCHi = 0xFF;        /* high byte of initialized   */
unsigned char uchCRCLo = 0xFF;    /* low byte of CRC initialized */
unsigned uIndex ;                     /* will index into CRC lookup table  */
```

```
        while (usDataLen—)              /* pass through message buffer  */
        { uIndex = uchCRCHi ^ *puchMsgg++ ;       /* calculate the CRC      */
          uchCRCHi = uchCRCLo ^ auchCRCHi[uIndex} ;
          uchCRCLo = auchCRCLo[uIndex] ;
        }
        return (uchCRCHi << 8 I uchCRCLo) ;
        }
```

Таблица старших байтов

```
        /* Таблица значений CRC для старших байтов */
static unsigned char auchCRCHi[ ] = {
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,
0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80,
0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81,
0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1,
0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80,
0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81,
0x40
};
```


Таблица младших байтов

```
        /* Таблица значений CRC для младших байтов */
static char auchCRCLo[ ] = {
0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7,
0x05, 0xC5, 0xC4,
0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB,
0x0B, 0xC9, 0x09,
0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE,
0xDF, 0x1F, 0xDD,
0x1D, 0x1C, 0xDC, 0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2,
0x12, 0x13, 0xD3,
0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E,
0xFE, 0xFA, 0x3A,
0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B,
0x2A, 0xEA, 0xEE,
0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1,
0x63, 0xA3, 0xA2,
0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD,
0x6D, 0xAF, 0x6F,
0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8,
0xB9, 0x79, 0xBB,
0x7B, 0x7A, 0xBA, 0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4,
0x74, 0x75, 0xB5,
0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0,
0x50, 0x90, 0x91,
0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59,
0x58, 0x98, 0x88,
0x48, 0x49, 0x89, 0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D,
0x4D, 0x4C, 0x8C,
0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83,
0x41, 0x81, 0x80,
0x40
};
```

Руководство по эксплуатации содержит информацию о принципе действия, устройстве, установке и использовании табло - терминала ТВ-330-16 (далее по тексту – табло). Перед установкой и использованием внимательно с ним ознакомьтесь и сохраните для последующего обращения как к справочнику.

1 НАЗНАЧЕНИЕ

Табло предназначено для удаленного управления в терминальном режиме процессором весовым ПВ-310 по стандартному интерфейсу RS-485, и подключению к локальной компьютерной сети предприятия по дополнительному интерфейсу RS-485.

2 ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

2.1 Основные технические характеристики табло ТВ-330-16 указаны в таблице 2.1.

2.2 Климатическое исполнение УХЛ 3.1 по ГОСТ 15150-69, но при температуре от минус 10 до +50 °С.

Таблица 2.1

Характеристика	Значение
Питание	от сети переменного тока напряжением $\left(220^{+22}_{-33}\right)$ В, частотой (50±1) Гц
Потребляемая мощность, ВА не более	10
Габаритные размеры, мм:	182 x 110 x 90
Масса, кг, не более:	1,25
Диапазон температур, ° С	от – 10 до + 50
Дисплей:ТВ-330-16	жидкокристаллический знакосинтезирующий с подсветкой, 16 знакомест, высота знакоместа 9,66 мм
Скорость обмена, бод	1200, 2400, 4800, 9600, 19200, 38400
Кнопки управления, шт.	4
Среднее время наработки на отказ, часов	10000
Средний срок службы, лет	10
Интерфейс RS-485, шт.	2
Длина кабеля RS-485, не более м	1200

2.3 Класс защиты табло от поражения электрическим током по ГОСТ 12.2.007.0-75 0

2.4 Изоляция силовых электрических цепей, замкнутых между собой, относительно корпуса платы контроллера ТВ-330-16 выдерживает в течение 1 мин. действие испытательного напряжения 1500 В практически синусоидальной формы, частотой от 45 до 65 Гц во всем диапазоне рабочих температур.

2.5 Электрическое сопротивление изоляции цепей питания, замкнутых между собой, относительно корпуса платы контроллера ТВ-330 не менее 40 МОм во всем диапазоне рабочих температур.

2.6 Степень защиты от воздействия окружающей среды IP65 по ГОСТ 14254-96.

2.7 Допустимая вибрация в диапазоне частот 10…55 Гц, амплитудой до 0,35 мм.

3 КОМПЛЕКТНОСТЬ

Табло выносное ТВ-330	1 шт.
Кронштейн с винтами крепления	1 шт.
Руководство по эксплуатации	1 шт.

4 УСТРОЙСТВО И РАБОТА

Табло-терминал является микропроцессорным устройством, которое по заданной программе осуществляет обмен данными с внешними устройствами по стандартным интерфейсам RS-485.

Интерфейс №1 используется для удаленного управления в терминальном режиме процессором весовым ПВ-310. Табло ТВ 330-16 передает процессору весовому ПВ-310 код нажатия кнопок и отображает состояние его индикатора.

Интерфейс №2 используется для интеграции в локальную компьютерную сеть предприятия. При этом табло выполняет роль ретранслятора информационных пакетов данных при обращениях компьютера к процессору весовому ПВ-310. Обмен данными между компьютером и процессором весовым ПВ-310 через табло осуществляется точно так же, как в случае прямого подключения компьютера к весовому устройству. Т.е. в обоих случаях компьютер обращается к весовому устройству по одному и тому же адресу и читает или записывает содержимое тех же ячеек, что и при отсутствии табло (см. протокол обмена и карту памяти на соответствующее весовое устройство).

Шкаф управления

A1

Плата контролера

X3

Конт.	Цепь
1	RS485A-1
2	RS485B-1
3	GND

X2

Конт.	Цепь
1	RS485A-2
2	RS485B-2

X1

Конт.	Цепь
3	~12В
4	~12В

Источник питания

X2

X3

X1

9	~220В
10	~220В

1 2 ~220В

A2

Плата контролера

Экран

X1

Конт.	Цепь
1	Ex-
2	Ex-
3	Sg1-
4	Sg1+
5	Ex+
6	Ex+
7	Sg2+
8	Sg2-
9	Sns+
10	Sns-

X4

Конт.	Цепь
1	RS 1A
2	RS 1B/Tx
3	Rx
4	2RS485A
5	2RS485B
6	GND485
7	Iout
8	GND
9	+24V
10	0V

X5

Конт.	Цепь
1	+
2	-
3	-1.1
4	+1.2
5	-2.1
6	+2.2
7	-3.1
8	+3.2
9	-4.1
10	+4.2

X2

Конт.	Цепь
1	+
2	-
3	-1.1
4	+1.2

A B 1 3

1 2 3 4 Экран 1 2 3 Экран 1 2 1 2 1

X1 1 2 3 4 5 6 7 8 9 10 11 12 13

1 2 3 4 Э 1 2 3 Э

Транспортер ленточный

A4

Зел. Бел. Красн. Черн. Экран

X4.2

Зел. 15 1
Бел. 14 2
Красн. 13 3
Черн. 12 4
Экран G Э
11
Экран
Син. 10 1
Черн. 9 2
Корич. 8 3

B1 B2 B3

"Дист" (3') (2') (1')

"Мест." (3) (2) (1) S3 R1

"Пуск" (3) (2) (1) S4

"Стоп" (3') (2') (1')

X4.1

1 2 3 4 5 6 7

2 1 3 1 2 3 2 1

M1 M ~3