МИКРОПРОЦЕССОРНЫЙ ПРОГРАММИРУЕМЫЙ РЕГУЛЯТОР RE15

ПОСЛЕДОВАТЕЛЬНЫЙ ИНТЕРФЕЙС С ПРОТОКОЛОМ MODBUS

Руководство по эксплуатации

Содержание

1. ВВЕДЕНИЕ

Микропроцессорный программируемый регулятор RE15 снабжен последовательным интерфейсом RS-485 для коммуникаций с другими устройствами.
Асинхронный коммуникационный протокол MODBUS использует для передачи данных последовательный интерфейс RS-485.
Конфигурирование параметров последовательного интерфейса описано в Руководстве по эксплуатации регулятора RE15.

Параметры линии последовательной связи регулятора RE15:

- адрес регулятора - 1…247
- скорость передачи данных - 2400, 4800, 9600 бит/с
- рабочий режим - ASCII, RTU
- информационный пакет - ASCII: 8N1, 7E1, 7O1;
  RTU: 8N2, 8E1, 8O1, 8N1
- максимальное время отклика - 1 с

Значение некоторых аббревиатур:

ASCII	=	American Standard Code for Information Interchange	Американский стандартный код обмена информацией
RTU	=	Remote Terminal Unit	Дистанционный терминал
LRC	=	Longitudinal Redundancy Check	Продольный контроль избыточным кодом
CRC	=	Cyclic Redundancy Check	Контроль циклическим избыточным кодом
CR	=	Carriage Return	Возврат каретки
LF	=	Line Feed (Character)	Перевод строки
MSB	=	Most Significant Bit	Старший [двоичный] разряд
Checksum	=	Control Sum	Контрольная сумма

2. ОПИСАНИЕ ПРОТОКОЛА MODBUS

Протокол MODBUS – стандарт, принятый производителями промышленных регуляторов для асинхронного обмена информацией между различными устройствами в системах измерения и управления. При использовании протокола MODBUS на последовательном интерфейсе RS-485 вводится понятие шина MODBUS.
Характеристики протокола MODBUS:

- ❖ Простые правила соединения по принципу “**master-slave**” **(ведущий-ведомый),**
- ❖ Защита передаваемых данных от ошибок,
- ❖ Подтверждение выполнения команды и оповещение об ошибках,
- ❖ Эффективные механизмы обеспечения безопасности системы,
- ❖ Асинхронная передача данных,
- ❖ Полудуплексный обмен данными.

Устройства на шине MODBUS могут взаимодействовать друг с другом по принципу **master-slave**, в котором только одно устройство (**master** – **ведущий**) может инициировать транзакции (передавать запросы), а другие устройства (**slaves** – **ведомые**) отвечают передачей запрашиваемых ведущим данных. Транзакция состоит из передаваемой ведущим устройством команды ведомому (ведомым) и ответной передачи данных от ведомого (ведомых) ведущему. Ответная передача включает в себя данные, запрошенные ведущим, или подтверждение об исполнении команды. Ведущий может запрашивать индивидуально отдельного ведомого, либо инициировать передачу широковещательного сообщения для всех ведомых в системе. При получении широковещательного запроса ответное сообщение не формируется.

ВАЖНО: на шине MODBUS регулятор RE15 может быть ТОЛЬКО ведомым **(slave)**!

Формат передаваемой информации:

- ❖**master = > slave:** адрес ведомого, код функции, данные передачи, контрольное слово для защиты передаваемого сообщения,
- ❖**slave = > master** адрес ведомого – отправителя, подтверждение исполнения команды, запрошенные данные, контрольное слово для защиты ответного сообщения от ошибок.

Если ведомое устройство обнаруживает ошибку при приеме сообщения или не может выполнить команду, оно генерирует специальное сообщение об ошибке и передает это сообщение ведущему в качестве ответа.

Для протокола MODBUS существуют два формата передачи: **ASCII и RTU**. Пользователь выбирает требуемый формат вместе с параметрами последовательного порта (скорость передачи данных, информационный пакет) при конфигурировании любого устройства.

По протоколу **MODBUS** передаваемые сообщения помещаются во фреймы, которые не связаны с последовательной передачей данных. Фреймы имеют определенное начало и окончание, что позволяет приемному устройству распознавать и отклонять неполные фреймы и сигнализировать об ошибке передачи.

Т.к. передача данных может вестись в одном из двух различных режимов (ASCII или RTU), различают два типа фреймов.

2.1. ASCII фреймы

В формате ASCII каждый байт информации передается как два ASCII символа.

Отличительная особенность этого формата состоит в том, что он позволяет делать большие интервалы времени между передачей символов в пределах одного сообщения (до 1 секунды) без возникновения ошибок при передаче.

Типичный формат фрейма сообщения приведен ниже:

Маркер начала сообщения	**Адрес**	**Функция**	**Данные**	**LRC контроль**	**Маркер окончания**
1 символ “:”	2 символа	2 символа	n символов	2 символа	2 символа CR LF

В формате ASCII сообщения начинаются с символа двоеточие (“:” – ASCII 3Ah) и оканчиваются символами “возврат каретки, перевод строки” (CR LF). Информационная часть фрейма защищена LRC кодом (Longitudinal Redundancy Check = Продольный контроль избыточным кодом).

2.2. RTU фреймы

В формате RTU сообщения начинаются и оканчиваются интервалом тишины, равным времени передачи минимум 3.5 х (время передачи одного символа).

Простейший способ осуществления указанного интервала – время передачи не менее 3.5 символов при данной скорости в сети.

Формат фрейма приведен ниже:

Маркер начала сообщения	**Адрес**	**Функция**	**Данные**	**CRC контроль**	**Маркер окончания**
T1-T2-T3-T4	8 бит	8 бит	n x 8 бит	16 бит	T1-T2-T3-T4

Маркер начала и конца сообщения обозначен условно как интервал четырех равных длин символа (информационного пакета). Код проверки составляет 16 бит и является результатом CRC обработки (Cyclic Redundancy Check = Контроль циклическим избыточным кодом) содержимого фрейма.

2.3. Характеристика полей фрейма

Адресное поле

Адресное поле фрейма сообщения содержит два символа (ASCII) или восемь бит (RTU).
Доступные адреса ведомых находятся в интервале 0 - 247. Ведущий обращается к ведомым, помещая их адреса в адресное поле фрейма. При ответе на запрос ведущего ведомый в адресное поле фрейма вставляет свой собственный адрес, что позволяет ведущему идентифицировать отвечающего ведомого. Адрес 0 используется для широковещательной передачи, его распознает любой ведомый, подключенный к шине.

Поле функции

Поле кода функции во фрейме сообщения содержит два символа (ASCII) или восемь бит (RTU). Диапазон кода функции 1 - 127. При отправке сообщения от ведущего к ведомому по полю кода функции ведомый понимает, какую операцию необходимо произвести. При ответе ведомого ведущему поле кода функции используется либо для индикации и подтверждения нормального (безошибочного) ответа или для фиксации ошибки, при которой реализация команды невозможна.
При формальном ответе ведомый просто повторяет оригинальный код функции. При подтверждении ошибки ведомый возвращает специальный код, эквивалентный оригинальному коду функции с установленным в 1 старшим битом. Код ошибки помещается в поле данных ответного фрейма.

Поле данных

Поле данных составлено из набора двух шестнадцатеричных чисел: 00-FF. Это может быть пара ASCII символов или один RTU символ в соответствии с режимом протокола. Поле данных в сообщении от ведущего к ведомому содержит дополнительную информацию, которая необходима ведомому для выполнения указанной функции. Оно может содержать адреса выходов или регистров, их количество, счетчик передаваемых байтов данных и т.п. Поле данных может не существовать (иметь нулевую длину) в определенных типах сообщений. Подобное происходит во всех случаях, когда указанная функция не содержит параметров.

Поле обнаружения ошибок

В протоколе MODBUS используются два метода контроля ошибок передачи. Содержание поля обнаружения ошибок зависит от выбранного формата передачи данных.

При использовании формата **ASCII** поле обнаружения ошибок содержит два ASCII символа. Контрольная сумма является результатом вычисления Longitudinal Redundancy Check (LRC) – продольного контроля избыточным кодом – сделанного над содержанием сообщения (без начала “:” и окончания CRLF). Контрольная сумма добавляется к сообщению как последнее поле фрейма, предшествующее маркеру окончания (CR, LF)

При использовании формата **RTU** поле контрольной суммы содержит 16-битовую величину, представленную как две 8-битовые. Контрольная сумма является результатом вычисления Cyclic Redundancy Check (CRC) - контроля циклическим избыточным кодом – сделанного над содержанием сообщения. Контрольная сумма добавляется к сообщению как последнее поле фрейма младшим байтом вперед.

2.4. Контрольная сумма LRC

Контрольная сумма LRC вычисляется путем сложения последовательно всех 8-битных байтов сообщения, складывая их так, чтобы перенос отбрасывался, и в конце дополняя бинарный результат. Сложение происходит исключая стартовый символ “:” и конечные CR, LF. 8-битная величина LRC помещается в конец фрейма сообщения как два ASCII символа: старший символ будет передан первым, а младший – вторым.

2.5. Контрольная сумма CRC

Процедура генерации контрольной суммы CRC реализуется согласно следующему алгоритму:

1. Загрузить 16-битный регистр числом FFFFh. Использовать далее как CRC регистр.

2. Первый байт из блока данных складывается по исключающему ИЛИ с младшим байтом CRC регистра. Результат помещается в CRC регистр.

3. Содержимое CRC регистра сдвигается на один бит вправо (в направлении младшего бита - LSB), старший бит заполняется 0 (MSB = 0).

4. Проверить состояние младшего бита (LSB), извлеченного из CRC регистра на предыдущем шаге. Если младший бит равен 0, повторяется шаг 3 (новый сдвиг).

   Если младший бит равен 1, делается операция исключающее ИЛИ регистра CRC и полиномиального числа A001.

5. Повторить шаг 3 и 4 восемь раз. По окончании 8-битный байт будет полностью обработан.

6. Повторить шаги со второго по пятый для следующего 8-битного байта сообщения. Это повторяется до тех пор, пока все байты сообщения не будут обработаны.

7. Финальное содержание регистра CRC и есть искомая контрольная сумма CRC.

8. Затем контрольная сумма CRC помещается в сообщение: сначала передается младший байт, а затем старший, как описано ниже.

2.6. Формат передачи символов

В протоколе **MODBUS** передача символов идет младшим битом вперед.

Формат единицы информации в режиме ASCII:

- 1 старт бит,
- 7 бит данных,
- 1 бит контроля четности (нечетности) или нет бита контроля четности
- 1 стоп бит при контроле четности или 2 стоп бита при отсутствии контроля четности.

Формат единицы информации в режиме RTU:

- 1 старт бит,
- 8 бит данных,
- 1 бит контроля четности (нечетности) или нет бита контроля четности
- 1 стоп бит при контроле четности или 2 стоп бита при отсутствии контроля четности.

2.7. Прекращение передачи данных

Для ведущего пользователь задает важный параметр – “максимальное время ответа на фрейм запроса”. При превышении значения данного параметра передача данных прерывается. Указанный интервал времени выбирается таким, чтобы каждый ведомый в сети (даже самый медленный) успевал ответить на запрос ведущего. Превышение данного интервала времени свидетельствует об ошибке передачи и соответственно воспринимается ведущим.
Если ведомый обнаруживает ошибку передачи данных, он не выполняет команду и не формирует ответ ведущему. Это приводит к превышению интервала ожидания ответа и прекращению передачи данных.

ВАЖНО: Для регулятора RE15 “максимальное время ответа на фрейм запроса” составляет 1 секунду.

3. ОПИСАНИЕ ФУНКЦИЙ

В регуляторе RE15 реализованы следующие функции протокола:

Код	Значение
03	Чтение из n-регистров
06	Запись в индивидуальный регистр
16	Запись в n-регистров
17	Идентификация ведомого

3.1. Чтение из n-регистров

Запрос:
Функция позволяет считывать значения, хранящиеся в регистрах опрашиваемого ведомого. **Регистры - это 16- или 32-битные ячейки памяти, которые могут содержать численные значения, связанные с меняющимися процессами и т.п.** Фрейм запроса содержит 16-битный стартовый адрес и число регистров, подлежащих считыванию.

ВАЖНО: Для регулятора RE15 максимальное число считываемых регистров за одно обращение равно 128.

Значения регистров с адресными данными может быть различным для различных типов устройств. Данная функция недоступна в широковещательном режиме.

***Пример**:* чтение из 3х регистров, начиная с адресного регистра 6Bh.

Адрес	Функция	Адресный регистр Hi	Адресный Регистр Lo	Число регистров Hi	Число регистров Lo	Контрольная сумма	
11	**03**	**00**	**6B**	**00**	**03**	**7E**	**LRC**

Ответ:

Данные регистров в ответе передаются, начиная с наименьших адресов: первый байт содержит старшие биты, затем второй байт – младшие биты.

Пример: ответный фрейм

Адрес	Функция	Число бит	Значение в рег.107 Hi	Значение в рег.107 Lo	Значение в рег.108 Hi	Значение в рег.108 Lo	Значение в рег.109 Hi	Значение в рег.109 Lo	Контр. Сумма	
11	03	06	02	2B	00	00	00	64	55	LRC

3.2. Запись значения в регистр (код 06)

Запрос:
Функция позволяет изменять содержание регистров. Доступна в широковещательном режиме.

Пример:

Адрес	Функция	Адресный регистр Hi	Адресный регистр Lo	Значение Hi	Значение Lo	Контрольная сумма	
11	06	00	87	03	9E	C1	LRC

Ответ:

Правильный ответ на запрос записи значения в регистр – передача ответного сообщения после выполнения операции.

Пример:

Адрес	Функция	Адресный регистр Hi	Адресный регистр Lo	Значение Hi	Значение Lo	Контрольная сумма	
11	06	00	87	03	9E	C1	LRC

3.3. Запись в n-регистров (код 16)

Запрос:
Функция доступна в широковещательном режиме. Она позволяет изменять содержание регистров. Для регулятора RE15 максимальное число регистров, записываемых за одно обращение, равно 128.

Пример: запись двух регистров, начиная с адресного регистра 136

Адрес	Функ-ция	Адресн. регистр Hi	Адресн. регистр Lo	Число регист-ров Hi	Число регист-ров Lo	Число байт	Дан-ные Hi	Дан-ные Lo	Дан-ные Hi	Дан-ные Lo	Контр. Сумма	
11	10	00	87	00	02	04	00	0A	01	02	45	LRC

Ответ:

Правильный ответ включает адрес ведомого, код функции, стартовый адрес и число записанных регистров.

Пример:

Адрес	Функция	Адресный регистр Hi	Адресный регистр Lo	Число регистров Hi	Число регистров Lo	Контрольная сумма	
11	10	00	87	00	02	56	LRC

3.4. Отчет об идентификации устройства (код 17)

Запрос:
Данная функция позволяет пользователю получить информацию о типе устройства, его статусе и конфигурации.

Пример:

Адрес	Функция	Контрольная Сумма	
11	11	DE	LRC

Ответ:
Поле “Идентификатор устройства” в ответном фрейме содержит уникальный идентификатор данного класса устройств, а остальные поля содержат параметры в зависимости от типа устройства.

ВАЖНО: Регулятор RE15 предоставляет информацию, связанную с дополнительными входами и выходами.

Пример для регулятора RE15

Адрес ведомого	Функция	Число Байтов	Идентификатор Устройства	Тип доп.входа	Тип выхода	Номер исполнения[2]	Контрольная сумма
11	**11**	**4**	**68**	**xx**[1)]	**yy**[2)]	**0**[3)]	

[1)]XX - значение из кода заказа регулятора – связано с дополнительным входом

[2)]YY - 0 - четыре дискретных выхода,
1 – один аналоговый выход + три дискретных выхода
2 – два аналоговых выхода + два дискретных выхода

[3)]0 - стандартное исполнение,
отличное от нуля значение – для исполнений на заказ

4. КОДЫ ОШИБОК

Когда ведущий передает запрос ведомым, кроме сообщений в широковещательном режиме, он ожидает ответного сообщения. При отправке запроса ведущим могут реализоваться следующие четыре ситуации:

- ❖ Ведомый получает запрос без ошибки передачи информации, правильно отрабатывает его и отправляет ответ ведущему;

- ❖ Ведомый не получает запроса, соответственно – не отправляет ответа. Условия таймаута для запроса выполняются в программе ведущего.

- ❖ Ведомый получает запрос с ошибкой передачи информации (ошибка контроля четности, контрольной суммы LRC или CRC) – ответ не отправляется. Условия таймаута для запроса выполняются в программе ведущего.

- ❖ Ведомый получает запрос без ошибки передачи информации, но не может корректно выполнить его (например, если в запросе содержится требование считать несуществующий регистр или выходной бит). При этом ведомый отправляет ответ с кодом ошибки, информируя ведущего о причине ошибки.

- ❖ Сообщение с ошибочным ответом содержит два поля в отличие от правильного ответа.

Поле кода функции:

В правильном ответе ведомый пересылает код функции из запроса в поле ответного кода функции. У всех кодов функций старший бит (MSB) равен нулю (значения кодов меньше 80h). В ответе на неправильный запрос с верным адресом ведомый передает в качестве старшего бита (MSB) – 1. Таким образом, значение кода функции в неправильном ответе оказывается на 80h больше, чем должно быть в правильном ответе.

На основе кода функции с заданием старшего бита (MSB), полученного от ведомого, программа ведущего идентифицирует неправильный запрос.

Поле данных:

В ответе на правильный запрос ведомый возвращает данные в поле данных (информация, которая была запрошена ведущим). В ответе на неправильный запрос ведомый помещает в поле данных код ошибки. Данный код определяет условия ведомого, которые привели к ошибке.

Пример неправильного запроса ведущего: чтение из регистра 4520 (11A8h) - и ответ ведомого: отсутствующий адрес данных, т.к. максимальный регистровый адрес регулятора RE15 – 4519.

Данные представлены в шестнадцатеричной форме.

Пример: запрос

Адрес ведомого	Функция	Адрес переменной Hi	Адрес переменной Lo	Число переменных Hi	Число переменных Lo	Контрольная сумма	
0A	**03**	**12**	**BO**	**00**	**01**	**39**	**LRC**

Пример: ответ

Адрес ведомого	Функция	Ошибка	Контрольная сумма	
0A	**83**	**02**	**71**	**LRC**

Возможные коды ошибок и их значение представлены в таблице ниже.

Код	Значение
01	Запрещенная функция
02	Запрещенный адрес данных
03	Запрещенное значение данных

5. ТАБЛИЦЫ РЕГИСТРОВ РЕГУЛЯТОРА RE15

В регуляторе RE15 данные хранятся в 16-битных регистрах.
Биты в регистре пронумерованы от младшего к старшему биту (младшим битом вперед) (b0-b15).
Список регистров представлен в таблице 1.
Символы R, W в колонке “Опции” означают допустимые действия над данными регулятора: R – чтение, W – запись.
Подробное описание данных в соответствии с приведенными в таблице 1 символами смотри «МИКРОПРОЦЕССОРНЫЙ ПРОГРАММИРУЕМЫЙ РЕГУЛЯТОР RE15. Руководство по эксплуатации.»

Таблица 1. Содержание 16-битных регистров

Адрес регистра	Опция	Символ	Диапазон	Описание
4000	R, W	inPu	0…16	Тип главного входа: 0-PT100, 1-PT1000, 2-Ni100, 3-Cu100, 4-J, 5-T, 6-K, 7-S, 8-R, 9-B, 10-E, 11-N, 12-хромель-копель, 13-сопротивление 0...400 W, 14-0...20 mA, 15-4...20 mA, 16-0...10V, 17- 0...5V
4001	R, W	t.L1	0, 1	0 – 2х-проводная схема, 1 – 3х-проводная схема
4002	R, W	r.L1	0…200	Сопротивление соединительного кабеля * 10
4003	R, W	ConP	-1…501	Компенсация температуры холодного спая термопары* 10; значения < 0 или > 500 = автоматическая компенсация
4004	R, W	L.cPP	0, 1, 2	Количество знаков после десятичной точки (для аналоговых входов – 2 знака)
4005	R, W	Sh1 F	-999…999	Смещение для главного входа[1)]
4006	R, W	SPLL	-999… SPLH	Нижний диапазон уставки или измеряемого значения на главном входе[1)]
4007	R, W	SPLH	SPLL...9999	Верхний диапазон уставки или измеряемого значения на главном входе[1)]
4008	R, W	i.2Lo	-999…i.2H1	Нижний измерительный диапазон на дополнительном входе[1)]
4009	R, W	i.2H1	i.2Lo…9999	Верхний измерительный диапазон на дополнительном входе[1)]
4010	R, W	i.inP	0…5	Измерительный диапазон дополнительного входа: 0-0...20 mA; 1-4...20 mA; 2-0...10 V ; 3-0...5 V; 4-0...100 W; 5-0...1000 W;
4011	R, W	F.1 n	0…4	Функция дополнительного аналогового входа: 0 – уставка (rsp = inp2); 1-дополнительное информационное измерение; 2- сумма сигналов на обоих аналоговых входах; 3 – разность сигналов на главном и дополнительном входах; 4-среднее арифметическое сигналов на аналоговых входах
4012	R, W	F.1 nb	0…3	Функция дискретного входа: 0 - вход не используется; 1- остановить регулирова ние (управляющий сигнал=0); 2- переклю чение на ручное регулирование, 3- конец программы; 4- остановка программы на последней вычисленной уставке

4013	R, W	oUC	0…3	Диапазон аналогового выхода**1**: 0-0-20 mA - выходы тока; 1-4-20 mA ; 2-0-10 V ; 3-0-5 V – выходы напряжения
4014	R, W	oUC	0…3	Диапазон аналогового выхода **2**: см.выше
4015	R, W	out	0…18	Функция аналогового выхода **1**: 0 – не используется, 1-Y1, 2-Y2-C, 3-Y2-S, 4-rH., 5-rLo, 6-dbH., 7-dbLo, 8-dbHL, 9-db.in, 10-rH.2, 11-rLo2, 12-E.out, 13-E.oP, 14-Err1, 15- Err2, 16-tr.1, 17-tr.2, 18-trSP
4016	R, W	out	0…18	Функция аналогового выхода **2**: см.выше
4017	R, W	out	0…18	Функция аналогового выхода **3**: см.выше
4018	R, W	out	0…18	Функция аналогового выхода **4**: см.выше
4019	R, W	bAr	0…4	Функция барграфа **1**: 0-сигнал управления Y1 0…100%; 1-сигнал управления Y2 0…100%; 2-измеряемое значение на главном входе SPLL…SPLH; 3-измеряемое значение на дополнительном входе .2Lo… .2H.; 4-уставка SPLL…SPLH
4020	R, W	bAr	0…4	Функция барграфа **2**: см.выше
4021	R, W	rSP	0, 1, 2	Тип задания: 0-постоянное, 1-программируемое, 2-ввод с дополнительного аналогового входа
4022	R, W	SP	Зависит от входа	Постоянная уставка[1)]
4023	R, W	nArc	0…999	Скорость изменения уставки, плавный пуск*10
4024	R, W	ncPr	1…15	Номер выполняемой программы при программном регулировании
4025	W R		0, 1, 2 0,1	Контроль работы программы: 0-стоп, 1-продолжение, 2-старт сначала
4026	R, W	Pb	0…9999	Пропорциональный коэффициент для контура I *10
4027	R, W	t.I	0…3600	Интегральный коэффициент для контура I
4028	R, W	t.d	0…1000	Дифференциальный коэффициент для контура I
4029	R, W	t.o	1…250	Время выборки для контура I
4030	R, W	H	0…999	Зона нечувствительности для ВКЛ/ВЫКЛ регулирования для контура I
4031	R, W	Pb-C	0…9999	Пропорциональный коэффициент для контура II *10
4032	R, W	t.I -C	0…3600	Интегральный коэффициент для контураII
4033	R, W	t.d-C	0…1000	Дифференциальный коэффициент для контура II
4034	R, W	to-C	1…250	Время выборки для контура II
4035	R, W	H1 -C	0…999	Зона нечувствительности для ВКЛ/ВЫКЛ регулирования для контура II[1)]
4036	R, W	Hn	0…999	Мертвая зона для регулирования с тремя состояниями[1)]
4037	R, W	tyPr	0, 1	Вид регулирования: 0-обратный, 1-прямой
4038	R, W	y-oF	0…1000	Смещение выхода PID-регулятора*10 (для интегрального коэффициента ti=0)

4039	R, W	1ASP	как для SP	Уставка для аварийного выхода 1[1)]
4040	R, W	1AH,	0…999	Зона нечувствительности аварийного выхода 1[1)]
4041	R, W	1APA	0, 1	Хранение аварии 1: 0-выкл, 1-вкл
4042	R, W	2ASP	как для SP	Уставка для аварийного выхода 2[1)]
4043	R, W	2AH,	0…999	Зона нечувствительности аварийного выхода 2[1)]
4044	R, W	2APA	0, 1	Хранение аварии 2: 0-выкл, 1-вкл
4045	R, W	3ASP	как для SP	Уставка для аварийного выхода 3[1)]
4046	R, W	3AH,	0…999	Зона нечувствительности аварийного выхода 3[1)]
4047	R, W	3APA	0, 1	Хранение аварии 3: 0-выкл, 1-вкл
4048	R, W	4ASP	как для SP	Уставка для аварийного выхода 4[1)]
4049	R, W	4AH,	0…999	Зона нечувствительности аварийного выхода 4[1)]
4050	R, W	4APA	0, 1	Хранение аварии 4: 0-выкл, 1-вкл
4052	R, W	cont	0, 1	Индикатор продолжения выполнения программы с постоянной уставкой после отключения питания: 0- регулирование выкл, 1- регулирование вкл.
4053	R, W	Auto	0, 1, 2	Алгоритм самонастройки: 0-без самонастройки, 1-метод идентификации, 2-метод осцилляции
4054… 4773 адреса отдельных регистров	R, W	LcYc	1…99	Число повторов программы p Номер регистра r для программы p равен: r = (p - 1) * 48 + 4054
	R, W	bloh	0…999	Величина блокировки в программе p[1)] Номер регистра r для программы p равен: r = (p - 1) * 48 + 4055
	R, W	Cont	0, 1	Продолжение программы p после отключения питания: 0-стоп, 1-продолжение, номер регистра r для программы p равен: r = (p - 1) * 48 + 4056
	R, W	nAXX	0…999	Скорость изменения уставки в сегменте xx*10 Номер регистра r для сегмента x программы p равен: r = (p - 1) * 48 + (x-1)*3 + 4057
	R, W	SPxx или tixx	диапазон в зависимости от входа 0…999	Значение уставки в конце сегмента, если скорость изменения уставки > 0, или время удержания, если скорость изменения уставки = 0; Номер регистра r для сегмента x программы p равен: r = (p - 1) * 48 + (x-1)*3 + 4058
	R, W	EoXX blXX	0, 1 0, 1	Выход y в сегменте xx (бит y-1): 1-выход вкл, 0-выход выкл. Индекс блокировки в сегменте xx (бит 4) xxx1xxx-1, 1-вкл, 0-выкл. Номер регистра r для сегмента x программы p равен: r = (p - 1) * 48 + (x-1)*3 + 4059

4774	R			Состояние устройства: бит 0: 1-измеряемое значение на главном входе меньше нижнего входного диапазона или вход замкнут бит 1: 1-измеряемое значение на главном входе больше верхнего входного диапазона или вход разомкнут бит 2: 1-измеряемое значение на дополнительном входе меньше нижнего входного диапазона бит 3: 1-измеряемое значение на дополнительном входе больше верхнего входного диапазона бит 4: состояние выхода 1: 0-выкл, 1-вкл бит 5: состояние выхода 2: 0-выкл, 1-вкл бит 6: состояние выхода 2: 0-выкл, 1-вкл бит 7: состояние выхода 2: 0-выкл, 1-вкл бит 8: 1-ручное регулирование, 0-автоматическое регулирование бит 9: 1-изменение уставки, т.н. плавный пуск бит 10: 1-программное регулирование, 0-регулирование с постоянной уставкой бит 11: 1-выполнение программы, 0-программа не выполняется бит 12: 1-блокировка программы вследствие слишком большого значения отклонения бит 13: состояние дискретного входа: 0-разомкнут, 1- замкнут бит 14 и 15: положение десятичной точки: 00-без десятичной точки, 01-десятичная точка в положении 1, 10-десятичная точка в положении 2 (см.ссылку [1)])
4775	R			Измеряемое значение на выходе 1[1)]
4776	R			Измеряемое значение на выходе 2[1)]
4777	R			Измеряемое значение[1)]
4778	R			Уставка (текущая) [1)]
4779	R	h	0…1000	Измеряемый сигнал контура I * 10
4780	R	c	0…1000	Измеряемый сигнал контура II * 10
4781	R	n	0…15	Число пройденных сегментов при программном управлении
4782	R	t		Длительность сегмента
4783	R	L.		Число оставшихся повторов программы

[1)] значение умножается на коэффициент в зависимости от значения параметра LcPP(положение десятичной точки):
если LcPP= 0, коэффициент = 1,
если LcPP = 1, коэффициент = 10,
если LcPP = 2, коэффициент = 100.

В таблице 2 представлены адреса регистров для 15-ти программ. К каждому регистровому адресу нужно добавить 4000.

Таблица 2. Адреса регистров для программ управления уставкой

Параметр		Программа														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
LCYC		54	102	150	198	246	294	342	390	438	486	534	582	630	678	726
Bloh		55	103	151	199	247	295	343	391	439	487	535	583	631	679	727
cont		56	104	152	200	248	296	344	392	440	488	536	584	632	680	728
nA 1		57	105	153	201	249	297	345	393	441	489	537	585	633	681	729
SP2 or ti1		58	106	154	202	250	298	346	394	442	490	538	586	634	682	730
Eout+blok		59	107	155	203	251	299	347	395	443	491	539	587	635	683	731
nA 2		60	108	156	204	252	300	348	396	444	492	540	588	636	684	732
SP2 or ti2		61	109	157	205	253	301	349	397	445	493	541	589	637	685	733
Eout+blok		62	110	158	206	254	302	350	398	446	494	542	590	638	686	734
nA 3		63	111	159	207	255	303	351	399	447	495	543	591	639	687	735
SP3 or ti3		64	112	160	208	256	304	352	400	448	496	544	592	640	688	736
Eout+blok		65	113	161	209	257	305	353	401	449	497	545	593	641	689	737
nA 4		66	114	162	210	258	306	354	402	450	498	546	594	642	690	738
SP4 or ti4		67	115	163	211	259	307	355	403	451	499	547	595	643	691	739
Eout+blok		68	116	164	212	260	308	356	404	452	500	548	596	644	692	740
nA 5		69	117	165	213	261	309	357	405	453	501	549	597	645	693	741
SP5 or ti5		70	118	166	214	262	310	358	406	454	502	550	598	646	694	742
Eout+blok		71	119	167	215	263	311	359	407	455	503	551	599	647	695	743
nA 6		72	120	168	216	264	312	360	408	456	504	552	600	648	696	744
SP6 or ti6		73	121	169	217	265	313	361	409	457	505	553	601	649	697	745
Eout+blok		74	122	170	218	266	314	362	410	458	506	554	602	650	698	746
nA 7		75	123	171	219	267	315	363	411	459	507	555	603	651	699	747
SP7 or ti7		76	124	172	220	268	316	364	412	460	508	556	604	652	700	748
Eout+blok		77	125	173	221	269	317	365	413	461	509	557	605	653	701	749
nA 8		78	126	174	222	270	318	366	414	462	510	558	606	654	702	750
SP8 or ti8		79	127	175	223	271	319	367	415	463	511	559	607	655	703	751
Eout+blok		80	128	176	224	272	320	368	416	464	512	560	608	656	704	752

номера регистров

nA 9	Номера регистров	81	129	177	225	273	321	369	417	465	513	561	609	657	705	753
SP9 or ti9		82	130	178	226	274	322	370	418	466	514	562	610	658	706	754
Eout+blok		83	131	179	227	275	323	371	419	467	515	563	611	659	707	755
nA10		84	132	180	228	276	324	372	420	468	516	564	612	660	708	756
SP10 or ti10		85	133	181	229	277	325	373	421	469	517	565	613	661	709	757
Eout+blok		86	134	182	230	278	326	374	422	470	518	566	614	662	710	758
nA11		87	135	183	231	279	327	375	423	471	519	567	615	663	711	759
SP11 or ti11		88	136	184	232	280	328	376	424	472	520	568	616	664	712	760
Eout+blok		89	137	185	233	281	329	377	425	473	521	569	617	665	713	761
NA12		90	138	186	234	282	330	378	426	474	522	570	618	666	714	762
SP12 or ti12		91	139	187	235	283	331	379	427	475	523	571	619	667	715	763
Eout+blok		92	140	188	236	284	332	380	428	476	524	572	620	668	716	764
nA13		93	141	189	237	285	333	381	429	477	525	573	621	669	717	765
SP13 or ti13		94	142	190	238	286	334	382	430	478	526	574	622	670	718	766
Eout+blok		95	143	191	239	287	335	383	431	479	527	575	623	671	719	767
nA14		96	144	192	240	288	336	384	432	480	528	576	624	672	720	768
SP14 or ti14		97	145	193	241	289	337	385	433	481	529	577	625	673	721	769
Eout+blok		98	146	194	242	290	338	386	434	482	530	578	626	674	722	770
nA15		99	147	195	243	291	339	387	435	483	531	579	627	675	723	771
SP15 or ti15		100	148	196	244	292	340	388	436	484	532	580	628	676	724	772
Eout+blok		101	149	197	245	293	341	389	437	485	533	581	629	677	725	773

ДОПОЛНЕНИЕ А
ВЫЧИСЛЕНИЕ КОНТРОЛЬНОЙ СУММЫ

В данном дополнении представлены примеры вычисления контрольной суммы LRC для формата ASCII и контрольной суммы CRC для формата RTU на языке Си.

Функция расчета LRC имеет два аргумента:

*unsigned char *outMsg;*	Указатель коммуникационного буфера, содержащего двоичные данные, по которым ведется расчет LRC.
unsigned short usDataLen;	Длина коммуникационного буфера в байтах.

Функция возвращает код LRC в виде беззнакового байта.

```
static unsigned char LRC(outMsg, usDataLen)
unsigned char *outMsg; /* буфер для расчета LRC */
unsigned short usDataLen; /* длина буфера в байтах */
{
unsigned char uchLRC = 0; /* инициализация LRC */
while (usDataLen- -)
uchLRC += *outMsg++; /* увеличить LRC на байт из указателя */
return ((unsigned char)(-(char uchLRC))); /* вернуть LRC */
}
```

Ниже представлен пример вычисления контрольной суммы CRC на языке Си. Все возможные значения контрольной суммы CRC помещены в две таблицы.
В первой таблице представлен старший байт всех 256 возможных значений 16-битного поля CRC, во второй таблице представлен соответственно младший байт.

Задание контрольной суммы CRC в виде таблицы значительно эффективнее, чем расчет нового значения CRC для каждого символа коммуникационного буфера.

Замечание: данная функция (см.ниже) представляет старший/младший байты контрольной суммы CRC и таким образом значение контрольной суммы, возвращаемое функцией, может быть напрямую помещено в коммуникационный буфер.

Функция расчета CRC имеет два аргумента:

*unsigned char *outMsg;* Указатель коммуникационного буфера, содержащего двоичные данные, по которым ведется расчет CRC.

unsigned short usDataLen; Длина коммуникационного буфера в байтах.

Функция возвращает код CRC в виде беззнакового целого.

```
unsigned short CRC16(puchMsg, usDataLen)
unsigned char *puchMsg; /* буфер для расчета CRC */
unsigned short usDataLen; /* длина буфера в байтах */
{
    unsigned char uchCRChi = 0xFF; /* инициализация
                                      старшего байта CRC */
    unsigned char uchCRClo = 0xFF; /* инициализация
                                      младшего байта CRC */
    while (usDataLen- -)
    {   uIndex =uchCRChi ^ *puchMsg++; /* расчет CRC */
        uchCRChi = uchCRClo ^ crc_hi[uIndex];
        uchCRClo = crc_lo[uIndex];
    }
    return(uchCRChi<<8 | uchCRClo);
}
```

```
// таблица старшего байта CRC
const unsigned char crc_hi[ ]={
  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, 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, 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, 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, 0x00, 0xC1, 0x81, 0x40, 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, 0x00, 0xC1, 0x81, 0x40, 0x01,
  0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
  0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
  0x40
};
```

```
// таблица младшего байта CRC
const unsigned char crc_lo[ ]={
  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,
  0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32, 0x36, 0xF6, 0xF7,
  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,
  0x2E, 0x2F, 0xEF, 0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
  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,
  0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C,
  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
};
```

Lubuskie Zakłady Aparatów Elektrycznych - LUMEL S.A.
ul. Sulechowska 1, 65-022 Zielona Góra, Poland

Tel.: (48-68) 32 95 100 (exchange)
Fax: (48-68) 32 95 101
www.lumel.com.pl
e-mail:lumel@lumel.com.pl

Export Department:
Tel.: (48-68) 329 53 02 or 53 04
Fax: (48-68) 325 40 91
e-mail: export@lumel.com.pl