ТЕРМОТРОНИК

Описание протокола обмена преобразователя измерительного «АДИ» с системой верхнего уровня

Редакция 1.08

ООО «ТЕРМОТРОНИК»
193318, Россия, Санкт-Петербург, ул. Ворошилова, д.2
Телефон, факс: +7 (812) 326-10-50
Сайт ООО «ТЕРМОТРОНИК»: **www.termotronic.ru**
Служба технической поддержки: **support@termotronic.ru**
тел. 8-800-333-10-34

Содержание

1 Введение 4
1.1 Общие сведения 4
1.2 Уровни протокола обмена 4
1.2.1 Логический уровень 4
1.2.2 Коммуникационный уровень 4
ADU «Modbus ASCII» 4
ADU «Modbus RTU» 5
ADU «Modbus TCP» 6
2 Реализованные функции протокола 7
2.1 Функция Modbus 03 (0x03 hex) (Read Holding Registers) и 04 (0x04 hex) (Read Input Registers) 7
2.2 Функция Modbus 06 (0x06 hex) (Write Single Register) 7
2.3 Функция Modbus 16 (0x10 hex) (Write Multiple Registers) 8
2.4 Функция Modbus 20 (0x14 hex) (Read File Record) 8
3 Организация доступа к данным 10
3.1 Используемые типы данных и условные обозначения 10
3.2 Регистровый доступ и характеристики параметров 10
3.3 Порядок хранения и передачи байт данных 10
3.4 Чтение/запись текущих и настроечных параметров 11
3.5 Организация архивов и считывание архивных данных 11
3.5.1 Интерпретация часового, суточного и месячного архивов 12
3.5.2 Интерпретация архива событий 13
4 Карта переменных 17
Приложение 1. Функция расчета контрольной суммы LRC 20
Приложение 2. Функции преобразования в ASCII и обратно 21
Приложение 3. Коды ошибок, возвращаемые прибором 22
Приложение 4. Функция расчета контрольной суммы Crc32 23
Приложение 5. Функции преобразования в BCD и обратно 24
Приложение 6. Оптимизация алгоритма считывания архивов 25
Приложение 7. Функция расчета контрольной суммы Cyclical Redundancy Checking (CRC16) 26

История редактирования

- 29.04.2014 создана редакция 1.00;
- 21.08.2014 создана редакция 1.01;
- 03.06.2019 создана редакция 1.02:
  - В карту переменных внесено описание переменных с адресами 358, 360, 362, 364.
  - Добавлено описание режима Modbus RTU и упоминание об автоматическом распознавании протокола обмена при получении запросов.
  - Добавлено приложение 7 – описание расчета Crc16.
- 12.11.2019 создана редакция 1.03:
  - В карту переменных внесено описание переменных с адресами 366, 370 – установка начальных значений интегралов.
- 10.08.2020 создана редакция 1.04;
  - исправлены реквизиты организации;
  - добавлена информация о порядке нумерации файлов при чтении архива;
  - добавлено описание команд включения/выключения дискретного выходы;
  - добавлено описание настроек дискретного выхода.
- 19.04.2021 создана редакция 1.05;
  - добавлено примечание о физической структуре организации данных в архивах;
- 03.03.2022 создана редакция 1.06:
  - добавлено описание по работе с интерфейсами RS-485 и Ethernet;
- 13.07.2022 создана редакция 1.07:
  - добавлено описание исключительной ситуации чтения данных функцией 0x14 (см. п. 2.4).
- 11.11.2026 создана редакция 1.08:
  - изменено описание параметра с адресом 345;
  - изменено описание параметра с адресом 577;
  - изменено описание параметра с адресом 92;

1 Введение

1.1 Общие сведения

Преобразователь измерительный «АДИ» (далее Прибор) позволяет получать текущие и архивные параметры, а также предоставляет доступ к чтению и изменению настроечных параметров через коммуникационный интерфейс. Физический уровень интерфейса соответствует стандартам RS-232C/RS-485, скорость обмена 9600/19200 бит/сек., 8 бит данных, 1 стоповый, контроль четности отсутствует, либо стандарту Ethernet.

1.2 Уровни протокола обмена

Этот раздел содержит краткие сведения из стандарта «Modbus». Подробную информацию можно подучить из документов, размещенных на сайте www.modbus.org:

- Modbus Application Protocol;
- Modbus Over Serial Line;

1.2.1 Логический уровень

Логический уровень протокола отвечает за способ доступа к данным. Протокол «Modbus» определяет понятие PDU (Protocol Data Unit), независимое от используемого коммуникационного протокола. PDU содержит 2 поля: код функции (длина 1 байт) и данные (длина не более 252 байт). Подробное описание логического уровня приведено в разделе Реализованные функции протокола.



Рисунок 1: Состав PDU

1.2.2 Коммуникационный уровень

Коммуникационный уровень протокола отвечает за доставку передаваемой информации между двумя совместимыми «Modbus». В различных коммуникационных протоколах PDU дополняется полями сетевой адрес, контрольная сумма, заголовок и т.д., образуя при этом, понятие ADU (Application Data Unit). Дополнительные поля требуются для адресации, идентификации и контроля целостности данных.

Независимо от коммуникационного протокола прибор работает только в режиме «ведомый». Это означает, что прибор может выполнить посылку только в ответ на запрос системы верхнего уровня. Время ответа прибора на запрос «чтение» не превышает 500 мс, на запрос «запись» не превышает 15000 мс. В случае отсутствия ответа от прибора система верхнего уровня должна выполнить повтор запроса.

ADU «Modbus ASCII»

В случае использования коммуникационного протокола «Modbus ASCII» PDU дополняется полями сетевой адрес и контрольная сумма.

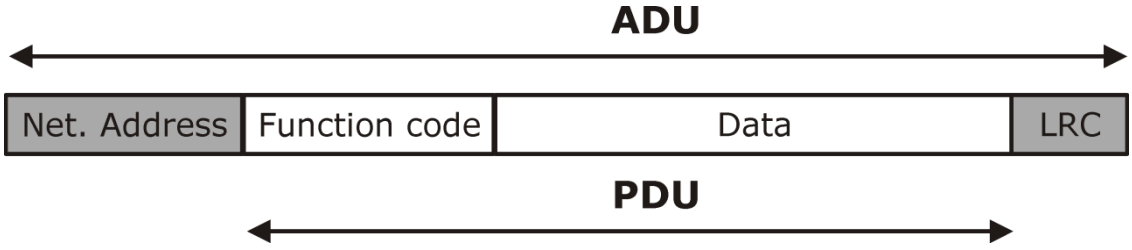


Рисунок 2: ADU для «Modbus RTU/ASCII»

Сетевой адрес служит для адресации прибора в сети. Контрольная сумма служит для проверки целостности данных. Передающее устройство вычисляет контрольную сумму над всеми полями посылки, и затем результат вычисления добавляет в конец посылки. Принимающее устройство, получив всю посылку, вычисляет контрольную сумму кадра для всех байтов сообщения, исключая байты контрольной суммы. В случае если принятая и вычисленная контрольные суммы равны, принимается решение о достоверности принятого кадра. В противном случае кадр считается недостоверным. Если прибор получает недостоверный кадр, он его игнорирует и не посылает каких-либо ответных сообщений. Это означает, что система верхнего уровня не получит ответа в течение ожидаемого времени и должна сделать повтор запроса. Если же факт получения недостоверной посылки обнаружен системой верхнего уровня, то она должна выполнить повтор запроса.

При передаче исходные двоичные данные кодируются. Начало и конец сообщения помечены специальными маркерами. Началом сообщения всегда является символ двоеточия ‘:’ (0x3A в шестнадцатеричном представлении). Концом сообщения всегда является пара символов «возврат каретки» (CR) и «перевод строки» (LF) (0x0D и 0x0A соответственно в шестнадцатеричном представлении). Каждый байт двоичного исходного сообщения передается в виде пары символов. Например, значение 27 (0x1B в шестнадцатеричном представлении) будет представлено как пара символов ‘1’ (0x31 - символьное представление старших 4-х битов) и ‘B’ (0x42 - символьное представление младших 4-х битов). Допустимые символы для передачи - это шестнадцатеричные символы 0-9, A-F. В качестве функции расчета контрольной суммы используется Longitudinal Redundancy Checking (LRC). Пример функции расчета LRC приведен в Приложении 1, а описание генерации контрольной суммы может быть найдено в документации на сайте www.modbus.org. Примеры функций перекодировки из двоичного представления в ASCII и из ASCII в двоичное представление приведены в Приложении 2. Над двоичным содержимым буфера передачи сначала выполняется расчет контрольной суммы. Затем двоичные данные вместе с полем контрольной суммы подвергаются преобразованию в ASCII и затем результат дополняется символами начала и конца кадра.

ADU «Modbus RTU»

В случае использования коммуникационного протокола «Modbus RTU» PDU дополняется полями сетевой адрес и контрольная сумма.

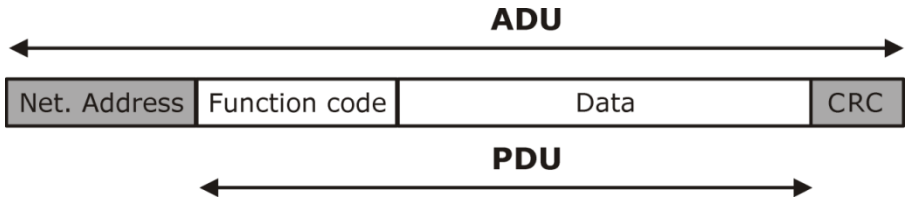


Рисунок 3: ADU для «Modbus RTU»

Сетевой адрес служит для адресации прибора в сети. Контрольная сумма служит для проверки целостности данных. Передающее устройство вычисляет контрольную сумму над всеми полями посылки, и затем результат вычисления добавляет в конец посылки. Принимающее устройство, получив всю посылку, вычисляет контрольную сумму кадра для всех байтов сообщения, исключая байты контрольной суммы. В случае, если принятая и вычисленная контрольные суммы равны, принимается решение о достоверности принятого кадра. В противном случае кадр считается недостоверным. Если прибор получает недостоверный кадр, он его игнорирует и не посылает каких-либо ответных сообщений. Это означает, что система верхнего уровня не получит ответа в течение ожидаемого времени и должна сделать повтор запроса. Если же факт получения недостоверной посылки обнаружен системой верхнего уровня, то она может выполнить повтор запроса.

При передаче используется двоичная система кодирования, начало и конец посылки определяются по длительности «интервала тишины» - времени, в течение которого не происходило передачи. Т.е. кадр сообщения «заключен» между двумя интервалами тишины. Байты сообщения должны передаваться непрерывным потоком. Длительность интервала тишины по стандарту «Modbus RTU» определяется как 3,5 длительности передачи байта на заданной скорости. В качестве функции расчета контрольной суммы в режиме RTU используется Cyclical Redundancy Checking (CRC16). Пример функции расчета CRC16 приведен в Приложении 7, а описание генерации контрольной суммы может быть найдено в документации на сайте www.modbus.org.

ADU «Modbus TCP»

В случае использования коммуникационного протокола «Modbus TCP» PDU дополняется заголовком MBAP (MODBUS Application Protocol), служащим для идентификации.

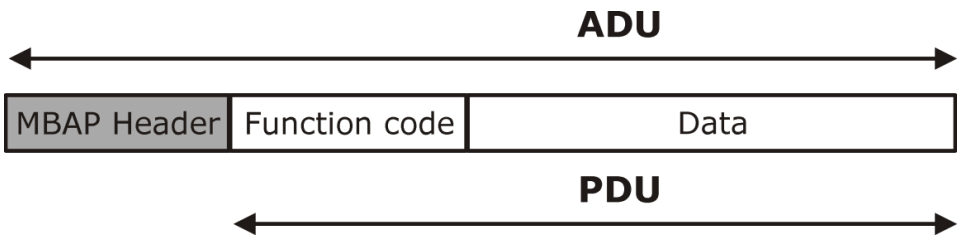


Рисунок 4: ADU для «Modbus TCP»

MBAP Header содержит следующие поля:

- Идентификатор запроса (длина 2 байта). Заполняется системой верхнего уровня. Копируется прибором из запроса при выполнении ответа;
- Идентификатор протокола (длина 2 байта). Должен быть равен 0;
- Длина сообщения в байтах (длина 2 байта). Должна быть равна длине PDU + 1;
- Сетевой адрес прибора (длина 1 байт). Заполняется системой верхнего уровня. При выполнении ответа прибором заполняется в соответствии с его настройкой.

Двухбайтные поля заголовка передаются в порядке сначала старший, затем младший байт.

ВНИМАНИЕ!	Приборы, имеющие версию встроенного ПО менее 1.02 не реализуют протокол Modbus RTU. В остальных приборах определение используемого протокола обмена выполняется автоматически в момент получения запроса. В связи с этим сетевой адрес 58 является недопустимым для приборов с версией ПО 1.02 и выше.

ВНИМАНИЕ!	Сетевой адрес 240 является широковещательным, т.е. на запрос с таким адресом отвечает любой прибор. При объединении нескольких приборов в сеть каждому из них должен быть присвоен уникальный в пределах сети сетевой адрес, отличный от широковещательного.

ВНИМАНИЕ!	При работе по интерфейсу Ethernet прибор может работать в режиме TCP сервера или TCP клиента. В режиме TCP сервера на порту 502 используется протокол Modbus TCP, на порту 5001 – Modbus ASCII (RTU при возможности). В режиме TCP сервера на клиентских портах менее 5000 используется протокол Modbus TCP, на портах 5000 и более – Modbus ASCII (RTU при возможности).

2 Реализованные функции протокола

2.1 Функция Modbus 03 (0x03 hex) (Read Holding Registers) и 04 (0x04 hex) (Read Input Registers)

Функции предназначены для чтения двоичного содержимого шестнадцатиразрядных регистров прибора. Отличие функции 0x04 от 0x03 в том, что она применяется только для чтения параметров, недоступных для записи. В общем виде структура запроса и ответа имеет следующий вид:

PDU запроса:

Функция **0x03/0x04**	Начальный адрес (старший байт)	Начальный адрес (младший байт)	Количество регистров (старший байт)	Количество регистров (младший байт)

Поле Data PDU содержит поля «Начальный адрес», указывающий с какого регистра начинать чтение, и «Количество регистров», указывающее, сколько регистров следует прочитать.

PDU ответа в случае выполнения без ошибок:

Функция **0x03/0x04**	Количество байт данных в ответе	1-ый регистр (старший байт)	1-ый регистр (младший байт)	Байты регистров 2,3…N

В случае успешного выполнения в ответе присутствует содержимое запрошенных регистров. Поле «Количество байт данных в ответе» будет равно количеству запрошенных регистров, умноженному на два. Прочитанное содержимое регистров начинается с байта, следующего за полем «Количество байт данных в ответе».

PDU ответа при возникновении ошибки:

Функция 0x83/0x84 (установлен старший бит)	Код ошибки

Для информирования ведущего о том, что операция не выполнена или выполнена с ошибкой, прибор устанавливает старший бит поля «Функция» в ответе. Байт, следующий за полем «Функция», будет содержать код ошибки (значения кодов ошибок приведены в Приложении 3).

2.2 Функция Modbus 06 (0x06 hex) (Write Single Register)

Функция предназначена для записи двоичного содержимого одного шестнадцатиразрядного регистра прибора.

PDU запроса:

Функция 0x06	Адрес (старший байт)	Адрес (младший байт)	Значение (старший байт)	Значение (младший байт)

Поле Data PDU запроса содержит поля «Адрес», указывающий в какой регистр выполняется запись, и значение записываемого регистра.

PDU ответа в случае выполнения без ошибок:

Функция 0x06	Адрес (старший байт)	Адрес (младший байт)	Значение (старший байт)	Значение (младший байт)

В случае успешного выполнения PDU ответа содержит копию первых пяти байт PDU запроса.

PDU ответа при возникновении ошибки:

Функция 0x86 (установлен старший бит)	Код ошибки

Для информирования ведущего о том, что операция не выполнена или выполнена с ошибкой, прибор устанавливает старший бит поля «Функция» в ответе. Байт, следующий за полем «Функция», будет содержать код ошибки (значения кодов ошибок приведены в Приложении 3).

2.3 Функция Modbus 16 (0x10 hex) (Write Multiple Registers)

Функция предназначена для записи двоичного содержимого шестнадцатиразрядных регистров прибора.

PDU запроса:

Функция 0x10	Нач-ый адрес (старший байт)	Нач-ый адрес (младший байт)	Кол-во рег-ов (старший байт)	Кол-во рег-ов (младший байт)	Кол-во байт для записи	1-ый регистр (старший байт)	1-ый регистр (младший байт)	Байты рег-ов 2,3…N

Поле Data PDU запроса содержит поля «Начальный адрес», указывающий с какого регистра начинать запись, «Количество регистров», указывающее, сколько регистров следует записать, «Количество байт для записи» и непосредственно значения записываемых регистров.

PDU ответа в случае выполнения без ошибок:

Функция 0x10	Нач-ый адрес (старший байт)	Нач-ый адрес (младший байт)	Кол-во регистров (старший байт)	Кол-во регистров (младший байт)

В случае успешного выполнения PDU ответа содержит копию первых пяти байт PDU запроса.

PDU ответ при возникновении ошибки:

Функция 0x90 (установлен старший бит)	Код ошибки

Для информирования ведущего о том, что операция не выполнена или выполнена с ошибкой, прибор устанавливает старший бит поля «Функция» в ответе. Байт, следующий за полем «Функция», будет содержать код ошибки (значения кодов ошибок приведены в Приложении 3).

2.4 Функция Modbus 20 (0x14 hex) (Read File Record)

Функция предназначена для чтения регистров файла. Файл организован как набор записей с номерами от 0000 до 9999. Функция может читать несколько различных групп регистров. Группы могут быть непоследовательными, но регистры внутри группы должны быть последовательными.

PDU запроса:

Функция 0x14	Длина запроса (7÷490)	Группа X, тип (всегда равен 6)	Группа X, номер файла (старш. байт)	Группа X, номер файла (младш. байт)	Группа X, номер записи (старш. байт)	Группа X, номер записи (младш. байт)	Группа X, длина записи (старш. байт)	Группа X, длина записи (младш. байт)	Группа X+1, тип (всегда равен 6)	Группа X+1, номер файла (старш. байт)	Группа X+1, номер файла (младш. байт)	…

Поле Data PDU запроса содержит:

- Поле «длина запроса»= 7 * N, где N – количество групп запрашиваемых регистров;
- Описание групп запрашиваемых регистров, каждое из которых имеет поля:
  - Тип. Всегда равен 6;
  - Номер файла;
  - Номер записи внутри файла;
  - Длина записи. Определяет количество запрашиваемых регистров группы.

PDU ответа в случае выполнения без ошибок:

Функция 0x14	Длина ответа	Группа X, длина (байт)	Группа X, тип (всегда равен 6)	Группа X, регистр 1 (старш. байт)	Группа X, регистр 1 (младш. байт)	Группа X, регистр 2 (старш. байт)	Группа X, регистр 2 (младш. байт)	...	Группа X+1, длина (байт)	Группа X+1, тип (всегда равен 6)	Группа X+1, регистр 1 (старш. байт)	Группа X+1, регистр 1 (младш. байт)	Группа X+1, регистр 2 (старш. байт)	Группа X+1, регистр 2 (младш. байт)	…

Поле Data PDU запроса содержит:

- Поле «длина ответа»=((Группа X, длина) + 2)+((Группа X+1, длина)+2)+…;
- Группы регистров, каждая из которых имеет:
  - Длина группы (байт);
  - Тип. Всегда равен 6;
  - Регистры данных.

PDU ответ при возникновении ошибки:

Функция 0x94 (установлен старший бит)	Код ошибки

Для информирования ведущего о том, что операция не выполнена или выполнена с ошибкой, прибор устанавливает старший бит поля «Функция» в ответе. Байт, следующий за полем «Функция», будет содержать код ошибки (значения кодов ошибок приведены в Приложении 3).

ВНИМАНИЕ!	В отступление от стандарта Modbus прибор может присылать ответ, содержащий только первые 4 байта PDU, и не содержащий данных запрошенной группы приборов. Такие ответы следует интерпретировать как ситуацию, когда прибор не нашел записи файла с достоверными данными.

3 Организация доступа к данным

3.1 Используемые типы данных и условные обозначения

- unsigned char –беззнаковое целое число (8 бит);
- signed char –знаковое целое число (8 бит);
- unsigned short –беззнаковое целое число (16 бит);
- signed short –знаковое целое число (16 бит);
- unsigned long –беззнаковое целое число (32 бита);
- signed long –знаковое целое число (32 бита);
- unsigned __int64 –беззнаковое целое число (64 бита);
- signed __int64 –знаковое целое число (64 бита);
- float – вещественное число одинарной точности (32 бита) с плавающей запятой, соответствующее стандарту «IEEE 754»;
- double – вещественное число двойной точности (64 бита) с плавающей запятой, соответствующее стандарту «IEEE 754»;
- R/O – доступ только на чтение;
- R/W – доступ на чтение и запись (безусловный);
- R/W* – доступ на чтение и запись. Разрешение на запись дается только при условиях, описанных отдельно;
- BUTTON - доступ на чтение, доступ на запись после однократного нажатия кнопки «Доступ» (длительность нажатия не менее 1 сек.);

3.2 Регистровый доступ и характеристики параметров

Доступ к текущим и настроечным параметрам прибора реализован через функции чтения и записи регистров – переменных, имеющих тип шестнадцатиразрядное беззнаковое целое. При организации регистрового доступа делается допущение, что все многообразные структуры данных располагаются в памяти, элементарной ячейкой которой является один шестнадцатиразрядный регистр типа «беззнаковое целое». Физически данные могут находиться в совершенно разных участках памяти прибора и даже в разных типах памяти (оперативная, энергонезависимая и т.д.), но для системы верхнего уровня данные «выглядят» как единое адресное пространство. В этом случае все доступные данные можно представить как массив шестнадцатиразрядных регистров, каждый из которых характеризуется номером в массиве (далее адресом). Каждый параметр прибора может занимать часть регистра, весь регистр целиком или несколько регистров. Таким образом, параметр характеризуется собственным типом и расположением внутри массива регистров.

Доступ к архивам организован через стандартные функции чтения файлов, предусмотренные протоколом «Modbus».

3.3 Порядок хранения и передачи байт данных

Для чтения и записи регистров в стандарте «Modbus» предусмотрены специальные функции, которые оперируют содержимым шестнадцатиразрядных регистров. Эти функции предполагают, что прибор хранит данные только типа шестнадцатиразрядное беззнаковое целое и ничего не «знают» о тех типах данных, которыми действительно представлены параметры прибора. Таким образом, получается, что в приборе данные хранятся в некоем исходном формате, а передаются по сети в виде набора шестнадцатиразрядных регистров. При передаче данных, чей размер в исходном формате превышает 16 бит (long, float, double и т.д.), используются несколько последовательных регистров. При этом младшие слова передаются в первую очередь, старшие - в последнюю. Т.о., для преобразования к порядку байт, естественному для платформы PC, требуется для каждого прочитанного/записываемого регистра изменить порядок байт.

Пример размещения данных для типа **long** (MSB-most significant byte, LSB-least significant byte):

B3	B2	B1	B0
MSB			LSB

Регистр	Регистр A0		Регистр A1	
Порядок передачи	первый			последний
Байт	B1	B0(LSB)	B3	B2(MSB)

Пример размещения данных для типа **float**:

B3	B2	B1	B0
SEEEEEEE	EMMMMMMM	MMMMMMMM	MMMMMMMM

Регистр	Регистр A0		Регистр A1	
Порядок передачи	первый			последний
Байт	B1	B0(LSB)	B3	B2(MSB)

Пример размещения данных для типа **double**:

B7	B6	B5-B1	B0
SEEEEEEE	EEEEMMMM	MMMMMMMM	MMMMMMMM

Регистр	Регистр A0		Регистр A1		Регистр A2		Регистр A3	
Порядок передачи	первый							последний
Байт	B1	B0(LSB)	B3	B2	B5	B4	B7(MSB)	B6

3.4 Чтение/запись текущих и настроечных параметров

Чтение/запись текущих и настроечных параметров производится при помощи функций чтения/записи регистров (функции чтения 0x03, 0x04, функции записи 0x06, 0x10). Параметры, доступные только на чтение, могут быть прочитаны только функцией 0x04. Параметры, доступные как на чтение, так и на запись, могут быть прочитаны обеими функциями по одинаковым адресам. Распределение переменных в адресном пространстве описано в разделе Карта переменных.

ВНИМАНИЕ!	Попытка записи любого параметра с уровнем доступа, отличным от «R/W», должна предваряться записью параметра «Авторизация» (см. *Карта переменных*). Параметр «Авторизация» должен быть записан единым массивом (20 байт).

3.5 Организация архивов и считывание архивных данных

Прибор содержит архивы следующих типов:

- События;
- Часовой архив;
- Суточный архив;
- Месячный архив;

Архивы имеют кольцевую структуру, т.е. при полном заполнении архива новые записи пишутся на место самых старых.

Архивы хранятся в приборе в виде файлов. Каждый файл содержит дескриптор файла (в первой записи), а также записи, представляющей одно событие или один «временной

срез» данных. Для чтения содержимого архивов используется функция чтения файлов (функция 0x14).

Каждый файл в первой записи (номер 0) хранит дескриптор файла. Остальные записи представляют данные файла. Дескриптор файла содержит:

- Длина первой записи (длина дескриптора, unsigned short);
- Тип первой записи (тип дескриптора, unsigned short, всегда равен 1);
- Длина файла в количестве записей (без учета первой записи, unsigned short);
- Длина каждой записи в байтах (кроме первой записи, unsigned short);
- Тип содержимого файла (unsigned short). Может принимать значения: 0 – файл событий, 1 – часовой архив, 2 – суточный архив, 3 – месячный архив;
- Индекс будущей записи (unsigned short). Принимает значение от 0 до «Длина файла в количестве записей – 1». Обнуляется при сбросе архива.
- Тотальный счетчик записей (unsigned long). Обнуляется при сбросе архива.

Алгоритм доступа к файлам:

1. Чтение записи с номером 0 из файла с номером N, где N≥1 (получение дескриптора файла). Если прибор вернул ошибку с кодом 2, значит, файлов больше нет;
2. Анализ дескриптора на предмет длины основных записей и длины файла;
3. Чтение оставшихся записей файла и их интерпретация;
4. Повторение пунктов 1-3 для следующего файла.

Каждая архивная запись, независимо от типа файла, имеет следующую структуру:

- Номер записи. Счетчик, который увеличивается на единицу при формировании новой записи внутри данного файла. Имеет тип unsigned __int64;
- Массив байт длиной «Длина каждой записи в байтах (см. дескриптор файла)» - 13. В нем размещается структура данных архивной записи (см. Интерпретация архивов);
- Служебный байт. Имеет тип unsigned char;
- Контрольная сумма записи. Имеет тип unsigned long и рассчитывается по алгоритму Crc32 (Проверка целостности данных должна выполняться над всем блоком данных длиной «Длина каждой записи в байтах (см. дескриптор файла)» включая контрольную сумму. Функция проверки контрольной суммы в Приложении 4).

ВНИМАНИЕ!	При чтении архивов следует учитывать физическую структуру размещения данных во flash-памяти. Под одну запись файла событий резервируется 64 байта, под записи файлов часового, суточного, месячного архивов – 256 байт. При этом, в приборах, имеющих аппаратную версию менее 4.02 размер стираемого сектора flash-памяти составляет 256 байт, в аппаратной версии 4.02 и выше – 4096 байт. Т.е. один стираемый сектор может содержать более одной архивной записи. Перед записью в начальные байты сектора flash-памяти производится стирание сектора, что приводит к уничтожению всех имеющихся в нем записей.

Алгоритм оптимизированного считывания архива приведен в Приложении 6.

3.5.1 Интерпретация часового, суточного и месячного архивов

Часовой, суточный архивы ведутся только при наличии функции архивирования в приборе. Наличие функции архивирования можно определить по значению переменной «Модель» (см. Карта переменных).

Необходимо прочитать все записи файла, выстроить их по признаку нарастания значения поля «Номер записи» и интерпретировать те записи, у которых сходится контрольная сумма.

Архивная запись содержит следующие поля:

Название	Смещение	Размер (байт)	Тип	Примечание
Дата/время	0	6	unsigned char[6]	сс:мм:чч дд:мм:гг в формате BCD
Среднее давление P1	6	4	float	(МПа)
Среднее давление P2	10	4	float	(МПа)
Минимальное давление P1	14	4	float	(МПа)
Минимальное давление P2	18	4	float	(МПа)
Максимальное давление P1	22	4	float	(МПа)
Максимальное давление P2	26	4	float	(МПа)
Минимальный расход по LIN	30	4	float	(м3/ч)
Максимальный расход по LIN	34	4	float	(м3/ч)
Дельта V+ LIN	38	4	float	Приращ. V+ за период архивирования (м3)
Дельта V- LIN	42	4	float	Приращ. V- за период архивирования (м3)
Интеграл V+ LIN	46	8	double	Интеграл V+ на момент архивирования (м3)
Интеграл V- LIN	54	8	double	Интеграл V- на момент архивирования (м3)
Вес импульса канала V1	62	4	float	(л/имп.)
Вес импульса канала V2	66	4	float	(л/имп.)
Дельта V1	70	4	float	Приращение V1 за период архивирования (м3)
Дельта V2	74	4	float	Приращение V2 за период архивирования (м3)
Интеграл V1	78	8	double	Интеграл V1 на момент архивирования (м3)
Интеграл V2	86	8	double	Интеграл V2 на момент архивирования (м3)
Сост. дискр.входов	94	1	unsigned char	*не используется*
Сост. дискр.выхода	95	1	unsigned char	*не используется*
Ошибки и состояния	96	4	unsigned long	См. ниже
КС калибровок	100	2	unsigned short	Контрольная сумма на момент архивирования
КС настроек	102	2	unsigned short	Контрольная сумма на момент архивирования
Дельта наработки	104	4	unsigned long	Приращ. нараб. за период архивирования (мин.)
Интеграл наработки	108	4	unsigned long	Интеграл нараб. (мин.)
Дельта нараб. без питания	112	4	unsigned long	Приращ. нараб. без пит. за период архивир. (мин.)
Интеграл нараб. без питания	116	4	unsigned long	Интеграл нараб. без пит. (мин.)
Серийный номер Питерфлоу	120	4	unsigned long	

Параметры, имеющие в названии «LIN» получают значения путем чтения из расходомера «Питерфлоу РС», подключенного к прибору по интерфейсу LIN.

Поле «Ошибки и состояния» является битовым полем, в котором каждый из битов кодирует следующие ошибки и состояния:

- 0-калибровки разрешены;
- 1-доступ к изменению параметров настройки разрешен;
- 2-сбой АЦП;
- 3-сбой flash;
- 4-P1 min;
- 5-P1 max;
- 6-P2 min;
- 7-P2 max;
- 8-нет связи по LIN;
- 9-было изменение настроек;
- 10-отключение питания;
- 11-авторизация по ключу;
- 12-факт выполнения калибровок.

3.5.2 Интерпретация архива событий

Архив событий может содержать более одного файла. Поэтому следует прочитать все записи всех файлов архива событий. Затем записи, у которых сходится контрольная

сумма отсортировать по возрастанию поля «Номер события» (см. ниже) и затем интерпретировать.

Архивная запись содержит следующие поля:

Название	Смещение	Размер (байт)	Тип	Примечание
Тип события	0	1	unsigned char	
Время наработки	1	4	unsigned long	
Внешняя временная метка	5	4	unsigned long	
Дата/время	9	6	unsigned char[6]	
Тип значения	15	1	unsigned char	
Тип переменной	16	2	unsigned short	
Старое значение	18	8	unsigned char[8]	
Новое значение	26	8	unsigned char[8]	
Идентификатор ключа доступа	34	4	unsigned char[4]	
Ошибки и состояния	38	4	unsigned long	
Номер события	42	4	unsigned long	

- Тип события. Определяет тип произошедшего события. Может принимать следующие значения:
  - 0x00 - Нет события;
  - 0x02 – Изменение параметра;
  - 0x03 – Перезапуск;
  - 0x04 – Вкл. питания;
  - 0x10 - Ошибка стека;
  - 0x11 – Ошибка измерения;
  - 0x12 – Ошибка часов;
  - 0x13 – Первый запуск;
  - 0x15 – Сброс архива;
  - 0x16 – Стирание архива;
  - 0x20 – Калибровка P;
  - 0x21 – Калибровка OUT;
  - 0x22 – Начало поверки;
  - 0x23 – Окончание поверки;
  - 0x24 – Установка часов;
  - 0x25 – Коррекция часов;
  - 0x26 – Запрет калибровок;
- Время наработки на момент записи (минут);
- Внешняя временная метка в формате BCD. Тип signed long. Представляет количество секунд, прошедшее от 01.01.1970 00:00:00;
- Дата/время. Если прибор не имеет функции архивирования, то анализировать данное поле не следует. Значение представлено в виде массива unsigned char из 6-ти элементов. Каждый байт содержит число в формате BCD (функции преобразования представлены в Приложении 5). Порядок следования: секунда, минута, час, день, месяц, год;
- Тип значения. Тип unsigned char. Определяет тип данных значений, хранимых в полях «Старое значение» и «Новое значение». Может принимать следующие значения:
  - 0 – нет значения;
  - 1 – unsigned char;
  - 2 – signed char;
  - 3 – unsigned short;
  - 4 – signed short;
  - 5 – unsigned long;
  - 6 – signed long;
  - 7 - float
  - 8 – unsigned __int64;
  - 9 – signed __int64;
  - 10 – double;
  - 17 – дата/время. Массив из 6-ти байт в формате BCD (секунда, минута, час, день, месяц, год).

- Тип переменной для события «Изменение параметра». Тип unsigned short. Содержит значение начального адреса регистра «Modbus» измененного параметра (см. Карта переменных);
- Старое значение. Массив из 8-ми байт. Содержит значение в соответствии с типом, указанным в поле «Тип значения»;
- Новое значение. Массив из 8-ми байт. Содержит значение в соответствии с типом, указанным в поле «Тип значения»;
- Идентификатор ключа доступа. Массив типа unsigned char из 4-х элементов;
- Ошибки и состояния. Тип unsigned long. Интерпретация аналогична интерпретации одноименного поля в часовом архиве;
- Номер события. Тип unsigned long. Счетчик, который увеличивается на единицу при формировании новой записи (сквозная нумерация событий, общая для всех файлов архива событий).

4 Карта переменных

Название	Адрес	Размер (байт)	Тип	Доступ	Примечание
Информация о приборе					
Тип устройства	0	2	unsigned short	R/O	0x1705
Аппаратная версия	1	2	unsigned short	R/O	старший байт версия, младший-редакция
Программная версия (ПВ)	2	2	unsigned short	R/O	старший байт – метрологически значимая часть, младший-метрологически незначимая часть
Контр.сумма метрологически значимой части ПО	3	2	unsigned short	R/O	Алгоритм CRC16
Контр.сумма метрологически незначимой части ПО	4	2	unsigned short	R/O	Алгоритм CRC16
Контр.сумма настроек	5	2	unsigned short	R/O	Алгоритм CRC16
Контр.сумма калибровок	6	2	unsigned short	R/O	Алгоритм CRC16
Модель	7	2	unsigned short	R/O	бит0: 0-без ток.выхода, 1-с токовым выходом; бит1: 0-без архива, 1-с архивом
Серийный номер	8	4	unsigned long	R/O	
Сервисная команда					
Команда	32	2	unsigned short	R/W*	(см.ниже)
Дата/время	33	6	unsigned char[6]	R/W*	сек.,мин.,час,день,месяц,год в формате BCD
Эталон	36	4	float	R/W*	(см.ниже)
Общие настройки					
Сетевой адрес	64	2	unsigned short	BUTTON	240-широковещательный
Отчетный час	65	2	unsigned short	BUTTON	0-23
Отчетные сутки	66	2	unsigned short	BUTTON	1-31
Наличие Питерфлоу	67	2	unsigned short	BUTTON	0-нет, 1-есть
Скорость обмена RS-232	68	2	unsigned short	BUTTON	0-9600, 1-19200 (бит/сек.)
Настройки входа V1					
Вес импульса	71	4	float	BUTTON	(л/импульс)
Настройки входа V2					
Вес импульса	76	4	float	BUTTON	(л/импульс)
Настройки входа P1					
Pmax	79	4	float	BUTTON	(Мпа)
Настройки входа P2					
Pmax	81	4	float	BUTTON	(МПа)
Настройки токового выхода					
Gmin	83	4	float	BUTTON	(м3/ч)
Gmax	85	4	float	BUTTON	(м3/ч)
Настройки дискретного выхода					
Режим	87	2	unsigned short	BUTTON	Список значений: 0 – выключен 1 - сраб. дискр.входа 1; 2 - сраб. дискр.входа 2; 3 - P1> уставки max; 4 - P1< уставки min; 5 - P2> уставки max; 6 - P2< уставки min; 7 - G> уставки max; 8 - G< уставки min; 9 - RS-232; 10-(P1< min)или(P1> max); 11-(P1≥min) и (P1≤max); 12-(P2< min)или(P2> max); 13-(P2≥min) и (P2≤max); 14-(G < min)или(G > max); 15-(G ≥min) и (G ≤max); 16-пустая труба; *Режимы 10 и более реализованы в версиях ПВ≥2.00*
Уставка min	88	4	float	BUTTON	Значение нижней уставки
Уставка max	90	4	float	BUTTON	Значение верхней уставки
Флаги	92	4	unsigned long	BUTTON	Бит 0 – выход активен при отсутствии измерений
Калибровки					
Вход P1 (АЦП min)	256	4	float	R/O	
Вход P2 (АЦП min)	258	4	float	R/O	
Вход P1 (АЦП max)	260	4	float	R/O	

Название	Адрес	Размер (байт)	Тип	Доступ	Примечание
Вход P2 (АЦП max)	262	4	float	R/O	
Вход P1 (ток в точке min)	264	4	float	R/O	
Вход P2 (ток в точке min)	266	4	float	R/O	
Вход P1 (ток в точке max)	268	4	float	R/O	
Вход P2 (ток в точке max)	270	4	float	R/O	
Выход OUT (ЦАП min)	272	4	float	R/O	
Выход OUT (ЦАП max)	274	4	float	R/O	
Выход OUT (ток в точке min)	276	4	float	R/O	
Выход OUT (ток в точке max)	278	4	float	R/O	
Текущие					
Дата/время	320	6	unsigned char[6]	R/O	сс:мм:чч дд:мм:гг в формате BCD
Расход LIN	323	4	float	R/O	(м3/ч)
V+ LIN (интеграл)	325	8	double	R/O	(м3)
V- LIN (интеграл)	329	8	double	R/O	(м3)
V1 интегр.	333	8	double	R/O	(м3)
V2 интегр.	337	8	double	R/O	(м3)
Давление P1	341	4	float	R/O	(МПа)
Давление P2	343	4	float	R/O	(МПа)
Дискр.входы V1, V2, Выход OUT	345	2	unsigned short	R/O	бит 0 – замкнут вход 1, бит 1 – замкнут вход 2, бит 8 – замкнут выход
Выходной ток	346	4	float	R/O	(мА)
События и ошибки	348	4	unsigned long	R/O	как в часовом архиве
Коррекция времени	350	4	unsigned long	R/O	разрешенная коррекция времени
Время наработки	352	4	unsigned long	R/O	(сек.)
Время без питания	354	4	unsigned long	R/O	(минут)
Серийный номер Питерфлоу	356	4	unsigned long	R/O	(минут)
V+ LIN (интеграл) *	358	4	signed long	R/O	(м3), целая часть перем-ой 325
V- LIN (интеграл) *	360	4	signed long	R/O	(м3), целая часть перем-ой 329
V1 интегр. *	362	4	signed long	R/O	(м3), целая часть перем-ой 333
V2 интегр. *	364	4	signed long	R/O	(м3), целая часть перем-ой 337
Авторизация					
Авторизация	577	16	unsigned char[16]+ unsigned long	R/W	Байты 0..15: данные авторизации + метка времени. Метка времени в формате количества секунд, прошедших от 01.01.1970 00:00:00
Наладка					
Вход P1 (код АЦП)	619	4	float	R/O	
Вход P1 (ток мА)	621	4	float	R/O	
Вход P2 (код АЦП)	623	4	float	R/O	
Вход P2 (ток мА)	625	4	float	R/O	
Вход V1 (кол-во фронтов имп.)	627	4	unsigned long	R/O	
Вход V2 (кол-во фронтов имп.)	629	4	unsigned long	R/O	
Начальные значения интегралов V1, V2					
Начальное значение V1 после сброса	366	8	double	BUTTON	Только в ПВ 1.05 и более
Начальное значение V2 после сброса	370	8	double	BUTTON	Только в ПВ 1.05 и более

* только для программной версии 1.02 и более (см. modbus адрес 2).

<u>Группа параметров «Сервисная команда»</u>:

При выполнении команды «запись» разрешено изменение только всех трех параметров («Команда», «Дата/время» и «Эталон») одним запросом. Параметр «Дата/время» представлен массивом из 6-ти байт в формате BCD (секунда, минута, час, день, месяц, год).

Параметр «Команда» может принимать следующие значения:

Значение	Название	Доступ	Примечание
7	установка даты/времени	BUTTON	При этом «Дата/время» содержит значение для установки, «Эталон» не анализируется прибором
8	Включить дискр. выход	R/W	Поля «Дата/время» и «Эталон» не анализируются приборов
9	Выключить дискр. выход	R/W	Поля «Дата/время» и «Эталон» не анализируются приборов
10	сброс архива	BUTTON	Параметры «Дата/время» и «Эталон» не анализируются прибором
13	коррекция даты/времени	R/W	Параметр «Эталон» содержит величину коррекции времени (секунд), «Дата/время» не анализируется прибором. Величина допустимой

			коррекции времени может быть прочитана из параметра «Коррекция времени» группы параметров «Текущие» (см. Карта переменных).

Приложение 1. Функция расчета контрольной суммы LRC

Пример функции расчета контрольной суммы кадра на языке СИ:

```
unsigned char Lrc(unsigned char * pSrc, int length)
{
     unsigned char locLrc=0;
     for(int i=0;i<length;i++)
          locLrc += *(pSrc+i);
     return locLrc = ~locLrc + 1;
}
```

где:

- pSrc – указатель на буфер, содержащий сообщение;
- length – количество байт данных, для которых требуется произвести подсчет LRC.

Приложение 2. Функции преобразования в ASCII и обратно

Ниже приведены примеры на языке СИ функций преобразования из ASCII формата в двоичный и обратного преобразования из двоичного в ASCII.

```
const     unsigned char CharToBin[23]={0,1,2,3,4,5,6,7,8,9,
0,0,0,0,0,0,0,
10,11,12,13,14,15};
const     char BinToChar[16]={'0','1','2','3','4','5','6','7',
'8','9','A','B','C','D','E','F'};

char TwoASCIItoBIN(char *cptr, unsigned char *bptr)
{
     char ca,i;
     unsigned char  cb;

     cb=0;
     for(i=0; i<2; i++)
     {
          ca=*cptr++;
          cb<<=4;
          if((ca >= '0') && (ca <= '9') || (ca >= 'A') && (ca <=
'F'))
               cb|=CharToBin[ca-0x30];
          else
               return(0);
     }
     *bptr=cb;
     return(1);
}
```

где:

- cptr – указатель на буфер, содержащий символы ASCII;
- bprt - указатель на буфер, куда записываются двоичные байты.

```
void BINtoTwoASCII(unsigned char *bptr, char *cptr)
{
     unsigned char  cb;

     cb=*bptr;
     *cptr=BinToChar[(cb>>4) & 0x0F];
     cptr++;
     *cptr=BinToChar[cb & 0x0F];
}
```

где:

- bptr – указатель на буфер, содержащий двоичные байты;
- cprt - указатель на буфер, куда записываются символы ASCII.

Приложение 3. Коды ошибок, возвращаемые прибором

0 - нет ошибок;
1 - недопустимая функция;
2 - недопустимый адрес в запросе;
3 - недопустимые значения в поле данных запроса;
4 – непоправимая ошибка возникла во время выполнения операции;
5 – подтверждение выполнения операции (не используется);
6 - запрос не может быть обработан сейчас, необходимо повторить запрос позднее;
129 – доступ запрещен;

Приложение 4. Функция расчета контрольной суммы Crc32

```
#define   CRC32_Pre()    0xFFFFFFFF
#define   CRC32_Pnom()   0xEDB88320
#define   CRC32_XOR()    0xFFFFFFFF
#define   CRC32_Check(_C)     (_C == 0xDEBB20E3)

unsigned char CRC32(void * pBuf, unsigned long len)
{
     unsigned long  j;
     unsigned long _CRC;

     _CRC = CRC32_Pre();
     for (j=0; j < len; j++) {
          _CRC = CRC32_Calc_Byte(*((BYTE*)(pBuf)+j), _CRC);
     }

     if (CRC32_Check(_CRC)) {
          return 1;
     } else {
          return 0;
     }
}

unsigned long CRC32_Calc_Byte(unsigned char _D, unsigned long _C)
{
     unsigned char  _i = 8;
     _C ^= (_D & 0xFF);
     do {
          if (_C & 1) { _C=(_C>>1)^CRC32_Pnom(); } else { _C>>=1; }
     } while (--_i);
     return (_C);
}
```

Приложение 5. Функции преобразования в BCD и обратно

```
unsigned char BinByteToBcdByte(unsigned char Src)
{
     unsigned char res;
     res = (unsigned char)( ((Src/10)<<4) | (Src % 10) );
     return res;
}

unsigned char BcdByteToBinByte(unsigned char Src)
{
     unsigned char res;
     res = (unsigned char)( ((Src>>4) * 10) + (Src & 0x0f) );
     return res;
}
```

Приложение 6. Оптимизация алгоритма считывания архивов

Доступ к архивам прибора реализован через считывание файлов по записям без возможности указания желаемой даты архивной записи. Это предполагает считывание всего файла архива целиком в каждом сеансе связи. При регулярном считывании архива это приводит к дополнительным временным затратам на получение архивных записей, которые были прочитаны во время предыдущих сеансов связи. Для экономии времени считывания алгоритм может быть оптимизирован на стороне программного обеспечения верхнего уровня, если оно имеет возможность сохранять прочитанные данные в каком-нибудь хранилище, например, базе данных.

Алгоритм основан на том, что база данных (БД) хранит как содержимое архивных записей, так и служебную информацию о записях (номер файла и индекс записи, из которых запись была прочитана).

Если для данного файла в БД нет информации, то производится считывание всех имеющихся в приборе записей (количество записей может быть определено из дескриптора файла).

Если БД содержит информацию о предыдущем считывании, то необходимо произвести считывание из прибора, указав номер файла и индекс записи такие же, как в последней записи БД (т.о. выполняется контрольное считывание последней известной записи).

Несовпадение инкрементного номера записи (тип unsigned __int64) или контрольной суммы прочитанной из прибора записи с аналогичными полями записи из БД говорит о том, что архивная запись была переписана с момента последнего считывания (архив «закольцевался»). В этом случае следует произвести считывание данных так, как будто БД не содержала информации об архиве.

В противном случае считывание данных выполняется, начиная со следующей записи. Считывание продолжается до возникновения одного из событий:

- Прочитаны все записи файла;
- Прочитана запись, имеющая значение инкрементного номер меньше максимального известного значения инкрементного номера записи (хранящегося в БД или прочитанного во время текущего сеанса связи). Данная ситуация говорит о том, что прочитана запись, которая уже должна находиться в БД.

Приложение 7. Функция расчета контрольной суммы Cyclical Redundancy Checking (CRC16)

Расчет контрольной суммы кадра RTU может выполняться по следующему алгоритму (текст на языке программирования СИ):

```
WORD Crc16(BYTE *Data, ULONG size) {
   union
   {BYTE b[2]; unsigned short w;} Sum;
   char shift_cnt;
   BYTE *ptrByte;
   ULONG     byte_cnt = size;

   ptrByte=Data;
   Sum.w=0xffffU;
   for(; byte_cnt>0; byte_cnt--) {
        Sum.w=(unsigned short)
   ((Sum.w/256U)*256U+((Sum.w%256U)^(*ptrByte++)));
        for(shift_cnt=0; shift_cnt<8; shift_cnt++) {
             if((Sum.w&0x1)==1)
                  Sum.w=(unsigned short)((Sum.w>>1)^0xa001U);
             else
                  Sum.w>>=1;
        }
   }
   return Sum.w;
}
```

Пусть какое-либо сообщение, имеющего длину N, записано в массиве Data[N+2] типа unsigned char. Тогда для этого сообщения контрольную сумму следует формировать следующим образом:

```
   WORD CheckSumm = Crc16(Data, N);
   Data[N]   = CheckSumm;
   Data[N+1] = CheckSumm>>8;
```