ООО «НПП Электромеханика»

Преобразователь измерительный цифровой многофункциональный ПЦ6806-03. Описание протокола обмена данными Modbus RTU

разработано на основе Modbus Application Protocol Specification V1.1b, Modbus over Serial Line Specification and Implementation Guide V1.02 http://www.Modbus-IDA.org

19 сентября 2013 г.

ОГЛАВЛЕНИЕ

ОБЩИЕ ПРИНЦИПЫ ПЕРЕДАЧИ ДАННЫХ В ПРОТОКОЛЕ MODBUS RTU

ПЕРЕДАЧА В СЕТИ MGTODBUS

Контроллеры соединяются, используя технологию главный-подчиненный, при которой только одно устройство (главный) может инициировать передачу (сделать запрос). Другие устройства (подчиненные) передают запрашиваемые главным устройством данные, или производят запрашиваемые действия. Типичное главное устройство включает в себя ведущий (HOST) процессор и панели программирования. Типичное подчиненное устройство - программируемый контроллер.

Главный может адресоваться к индивидуальному подчиненному или может инициировать широкую передачу сообщения на все подчиненные устройства. Подчиненное устройство возвращает сообщение в ответ на запрос, адресуемый именно ему. Ответы не возвращаются при широковещательном запросе от главного.

ЦИКЛ ЗАПРОС – ОТВЕТ

Запрос от главного	Ответ подчинённого
Адрес	Адрес
Код функции	Код функции
8 битные байты данных	8 битные байты данных
Контрольная сумма	Контрольная сумма

Запрос: Код функции в запросе говорит подчиненному устройству, какое действие необходимо провести. Байты данных содержат информацию, необходимую для выполнения запрошенной функции. Например, код функции 4 подразумевает запрос на чтение содержимого регистров подчиненного.

Ответ: Если подчиненный дает ответ, код функции в ответе повторяет код функции в запросе. В байтах данных содержится затребованная информация. Если имеет место ошибка, то код функции модифицируется, и в байтах данных передается причина ошибки.

ПЦ6806-03 начинает отвечать через временной интервал, равный времени передачи 3,5 символов, после последнего байта запроса.

СОДЕРЖАНИЕ СООБЩЕНИЯ MODBUS

Режим ASCII не используется.

RTU ФРЕЙМ

В RTU режиме сообщение начинается с интервала тишины продолжительностью более 3,5 символа при данной скорости передачи в сети. Первым байтом затем передается адрес устройства.

Вслед за последним передаваемым символом также следует интервал тишины продолжительностью не менее 3,5 символов. Новое сообщение может начинаться после этого интервала.

Фрейм сообщения передается непрерывно. Интервал тишины продолжительностью более 1,5 символа во время передачи фрейма, воспринимается устройством как ошибка.

Типичный фрейм сообщения показан ниже.

Старт	Адрес	Функция	Данные	CRC	Конец

T1-T2-T3-T4	8 бит	8 бит	N x 8 бит	16 бит	T1-T2-T3-T4

СОДЕРЖАНИЕ АДРЕСНОГО ПОЛЯ

Адресное поле фрейма содержит 8 бит. Допустимый адрес передачи находится в диапазоне 0 - 247. Каждому подчиненному устройству присваивается адрес в пределах от 1 до 247.

Адрес 0 используется для широковещательной передачи, его распознает каждое устройство.

СОДЕРЖАНИЕ ПОЛЯ ФУНКЦИИ

Поле функции фрейма содержит 8 бит. Диапазон числа 1–255. Имеющийся набор функций описан в разделе «Функции контроля и обработки данных».

Когда подчиненный отвечает главному, он использует поле кода функции для фиксации ошибки. В случае нормального ответа подчиненный повторяет оригинальный код функции. Если имеет место ошибка, возвращается код функции с установленным в 1 старшим битом.

Например, сообщение от главного подчиненному прочитать группу регистров имеет следующий код функции:

0000 0011 (03h) Если подчиненный выполнил затребованное действие без ошибки, он возвращает такой же код. Если имеет место ошибка, то он возвращает:

1000 0011 (83h) В добавление к изменению кода функции, подчиненный размещает в поле данных уникальный код, который говорит главному, какая именно ошибка произошла или причину ошибки.

СОДЕРЖАНИЕ ПОЛЯ ДАННЫХ

Поле данных в сообщении от главного к подчиненному содержит дополнительную информацию, которая необходима подчиненному для выполнения указанной функции. Оно может содержать адреса регистров или выходов, их количество, счетчик передаваемых байтов данных.

Например, если главный запрашивает у подчиненного прочитать группу регистров (код функции 04h), поле данных содержит адрес начального регистра и количество регистров. Если главный хочет записать группу регистров (код функции 10h), поле данных содержит адрес начального регистра, количество регистров, счетчик количества байтов данных и данные для записи в регистры.

Поле данных может не существовать (иметь нулевую длину) в определенных типах сообщений.

СОДЕРЖАНИЕ ПОЛЯ КОНТРОЛЬНОЙ СУММЫ

Когда используется RTU-режим, поле контрольной суммы содержит 16-ти битовую величину. Контрольная сумма является результатом вычисления Cyclical Redundancy Check сделанного над содержанием сообщения. CRC добавляется к сообщению последним полем младшим байтом вперед.

ФОРМАТ ПЕРЕДАЧИ СИМВОЛОВ

Передача символов идет младшим битом вперед.

ФОРМАТ КАЖДОГО БАЙТА В RTU-РЕЖИМЕ

старт	1	2	3	4	5	6	7	8	паритет	стоп

Система кодировки: 8-ми битовая двоичная, шестнадцатеричная

0-9, A-F

Две шестнадцатеричные цифры содержатся в

каждом 8-ми битовом байте сообщения.

Назначение битов: 1 стартовый бит

8 бит данных, младшим значащим разрядом вперед

1 бит паритета с контролем четности

1 стоповый бит

МЕТОДЫ КОНТРОЛЯ ОШИБОК

Контроль паритета и контрольная сумма. Главное и подчинённое устройства проверяют каждый байт и всё сообщение в процессе приема. Если подчиненный обнаружил ошибку передачи, то он не формирует ответ главному.

КОНТРОЛЬ ПАРИТЕТА

Устройства используют чётный (Even) паритет. Например, 8 бит RTU фрейма содержат биты 1100 0101. Общее количество единиц – 4. Бит паритета будет равен 0, так чтобы общее количество единиц вместе с битом паритета было чётным числом.

КОНТРОЛЬНАЯ СУММА CRC

Контрольная сумма CRC состоит из двух байт. Контрольная сумма вычисляется передающим устройством и добавляется в конец сообщения. Принимающее устройство вычисляет контрольную сумму в процессе приема и сравнивает ее с полем CRC принятого сообщения.

Счетчик контрольной суммы предварительно инициализируется числом FFh. Только восемь бит данных используются для вычисления контрольной суммы CRC. Старт и стоп биты, бит паритета, если он используется, не учитываются в контрольной сумме.

Во время генерации CRC каждый байт сообщения складывается по исключающему ИЛИ с текущим содержимым регистра контрольной суммы. Результат сдвигается в направлении младшего бита, с заполнением нулем старшего бита. Если младший бит равен 1, то производится исключающее ИЛИ содержимого регистра контрольной суммы и определенного числа. Если младший бит равен 0, то исключающее ИЛИ не делается.

Процесс сдвига повторяется восемь раз. После последнего (восьмого) сдвига, следующий байт складывается с текущей величиной регистра контрольной суммы, и процесс сдвига повторяется восемь раз как описано выше. Конечное содержание регистра и есть контрольная сумма CRC.

СИСТЕМА КОМАНД ИЗМЕРИТЕЛЬНОГО ЦИФРОВОГО ПРЕОБРАЗОВАТЕЛЯ ПЦ6806-03

СПИСОК КОМАНД ПЦ6806-03

Код функции	Описание
01h	Чтение выходов ТУ
02h	Чтение входов ТС
03h	Чтение фиксированных регистров
04h	Чтение регистров
05h	Установка выходов ТУ
06h	Запись в единичный регистр
07h	Чтение регистра статуса
08h	Функция диагностики
10h	Запись нескольких регистров
2Bh/0Eh	Запрос идентификации

01H ЧТЕНИЕ ВЫХОДОВ ТУ

Функция «Чтение выходов ТУ» используется для считывания состояния выходов телеуправления.

Запрос содержит адрес начального выхода и количество выходов для чтения. Выходы ТУ адресуются с нуля: выходы 1-16 адресуются как 0-15. Адреса 8-15 соответствуют выходам ТУ, зафиксированным функцией 06h.

Запрос

Код функции	1 байт	0x01
Стартовый адрес	2 байта	0x0000 – 0x000F
Количество битов	2 байта	1-16

Ответ

Код функции	1 байт	0x01
Количество байт	1 байт	1 – 2*
Значения битов	1 – 2 байт*	

* Количество запрашиваемых битов/8 округляется до большего целого

Сообщение об ошибке

Код ошибки	1 байт	0x81
Причина ошибки	1 байт	02 или 03

Состояние выходов ТУ передаётся, как один выход ТУ на бит.

Пример. Запрос для чтения выхода ТУ1.

Запрос		Ответ	
Поле	Hex	Поле	Hex
Адрес	01	Адрес	01
Код функции	01	Код функции	01
Начальный адрес ст.	00	Кол-во байт	01

Продолжение **примера**

Запрос		Ответ	
Поле	Hex	Поле	Hex
Начальный адрес мл.	00	Значение регистра ст.	01
Кол-во битов ст.	00	CRC ст.	90
Кол-во битов мл.	01	CRC мл.	48
CRC ст.	FD		
CRC мл.	CA		

ТУ1 включено.

Если возвращаемое количество битов не кратно 8, то оставшиеся биты в последнем байте сообщения будут установлены в 0. Счётчик байт содержит количество байт, передаваемых в поле данных.

02H ЧТЕНИЕ ВХОДОВ ТС

Функция «Чтение входов ТС» используется для считывания состояния входов телесигнализации. Запрос содержит адрес начального входа и количество входов для чтения. Входы ТС адресуются с нуля: входы 1-8 адресуются как 0-7. По адресам 8-15 расположены те же ТС, только зафиксированные по команде 06h.

Запрос

Код функции	1 байт	0x02
Стартовый адрес	2 байта	0x0000 – 0x000F
Количество битов	2 байта	1 – 16

Ответ

Код функции	1 байт	0x02
Количество байт	1 байт	1 – 2*
Значения битов	1 – 2 байт*	

* Количество запрашиваемых битов / 8, округлённое до целого вверх

Сообщение об ошибке

Код ошибки	1 байт	0x82
Причина ошибки	1 байт	02 или 03

Состояние выходов ТС передаётся, как один выход ТС на бит.

Пример. Запрос для чтения 16 дискретных входов. Первые 8 текущие ТС, остальные фиксированные.

Запрос		Ответ	
Поле	Hex	Поле	Hex
Адрес	01	Адрес	01
Код функции	02	Код функции	02
Начальный адрес ст.	00	Кол-во байт	02
Начальный адрес мл.	00	Данные	00
Кол-во битов ст.	00	Данные	00
Кол-во битов мл.	10	CRC ст.	B9
CRC ст.	79	CRC мл.	B8
CRC мл.	C6		

03H ЧТЕНИЕ ФИКСИРОВАННЫХ РЕГИСТРОВ

Функция «Чтение фиксированных регистров» используется для считывания регистров ПЦ, зафиксированных функцией 06h. Адреса регистров приведены в таблице 3.

Запрос содержит номер начального регистра и количество регистров для чтения.

Данные регистров в ответе передаются как два байта на регистр. Для каждого регистра первый байт содержит старшие биты, второй байт содержит младшие биты.

Запрос

Код функции	1 байт	0x03
Стартовый адрес	2 байта	0x0000 – 0xFFFF
Количество регистров	2 байта	1 – 125 (0x7D)

Ответ

Код функции	1 байт	0x03
Количество байт	1 байт	2 x N*
Значения регистров	N* x 2 байта	

*N = количество регистров

Сообщение об ошибке

Код ошибки	1 байт	0x83
Причина ошибки	1 байт	02 или 03 или 04

Пример. Запрос для чтения 3 регистров мгновенных фиксированных значений активной мощности фаз A, B, C.

Запрос		Ответ	
Поле	Hex	Поле	Hex
Адрес	01	Адрес	01
Код функции	03	Код функции	03
Начальный адрес ст.	00	Кол-во байт	06
Начальный адрес мл.	07	Значение регистра ст.	00
Кол-во регистров ст.	00	Значение регистра мл.	65
Кол-во регистров мл.	03	Значение регистра ст.	00
CRC ст.	E5	Значение регистра мл.	66
CRC мл.	CA	Значение регистра ст.	00
		Значение регистра мл.	00
		CRC ст.	8D
		CRC мл.	62

Содержимое регистра 0x0007 - Мгновенная активная мощность фазы A (P1a) - имеет значение 0x0065 (101)

Содержимое регистра 0x0008 - Мгновенная активная мощность фазы B (P1b) - имеет значение 0x0066 (102)

Содержимое регистра 0x0009 - Мгновенная активная мощность фазы C (P1c) - имеет значение 0x0000 (0)

04H ЧТЕНИЕ РЕГИСТРОВ

Функция «Чтение регистров» используется для считывания регистров ПЦ, содержащих значения параметров, указанных в таблице 3.

Запрос содержит номер начального регистра и количество регистров для чтения.

Данные регистров в ответе передаются как два байта на регистр. Для каждого регистра первый байт содержит старшие биты, второй байт содержит младшие биты.

Запрос

Код функции	1 байт	0x04
Стартовый адрес	2 байта	0x0000 – 0xFFFF
Количество регистров	2 байта	1 – 125 (0x7D)

Ответ

Код функции	1 байт	0x04
Количество байт	1 байт	2 x N*
Значения регистров	N* x 2 байта	

*N = количество регистров

Сообщение об ошибке

Код ошибки	1 байт	0x84
Причина ошибки	1 байт	02 или 03 или 04

Пример

Запрос для чтения 1 регистра «Мгновенное действующее значение напряжения фазы A (U1a)»

Запрос		Ответ	
Поле	Hex	Поле	Hex
Адрес	01	Адрес	01
Код функции	04	Код функции	04
Начальный адрес ст.	02	Кол-во байт	02
Начальный адрес мл.	00	Значение регистра ст.	00
Кол-во регистров ст.	00	Значение регистра мл.	02
Кол-во регистров мл.	01	CRC ст.	38
CRC ст.	30	CRC мл.	F1
CRC мл.	72		

Содержимое регистра 0x0200 имеет значение 0x0002.

05H УСТАНОВКА ЕДИНИЧНОГО ВЫХОДА ТУ

Функция используется для установки единичного выхода телеуправления в состояние включено/выключено. При широковещательной передаче функция устанавливает все выходы с данным адресом во всех подчинённых устройствах.

Запрос содержит номер выхода ТУ для установки. Выходы ТУ адресуются с нуля: выходы 1-8 адресуется как 0-7.

Состояние, в которое необходимо установить выход описывается в поле данных. Значение 0xFF00 соответствует включению. Значение 0x0000 соответствует выключению.

Запрос

Код функции	1 байт	0x05
Адрес выхода	2 байта	0x0000-0x0007
Данные	2 байта	0x0000 или 0xFF00

Ответ

Код функции	1 байт	0x05
Адрес выхода	1 байт	0x0000-0x0007
Данные	1 байт	0x0000 или 0xFF00

Сообщение об ошибке

Код ошибки	1 байт	0x85
Причина ошибки	1 байт	02 или 03 или 04

Пример. Включить ТУ1.

Запрос		Ответ	
Поле	Hex	Поле	Hex
Адрес	01	Адрес	01
Код функции	05	Код функции	05
Адрес ТУ ст.	00	Адрес ТУ ст.	00
Адрес ТУ мл.	00	Адрес ТУ мл.	00
Данные ст.	FF	Данные ст.	FF
Данные мл.	00	Данные мл.	00
CRC ст.		CRC ст.	
CRC мл.		CRC мл.	

ТУ1 включено.

Нормальный ответ повторяет запрос.

06H ЗАПИСЬ В ЕДИНИЧНЫЙ РЕГИСТР

Эта функция используется для записи одного регистра в удалённом устройстве. Запрос содержит адрес регистра и значение, которое должно быть записано. Нормальный ответ – эхо запроса - возвращается после того как содержимое регистра в удалённом контроллере перезаписано.

Запрос

Код функции	1 байт	0x06
Адрес регистра	2 байта	0xXXXX*
Значение регистра	2 байта	0xXXXX*

Ответ

Код функции	1 байт	0x06
Адрес регистра	2 байта	0xXXXX*
Значение регистра	2 байта	0xXXXX*

Сообщение об ошибке

Код ошибки	1 байт	0x86
Причина ошибки	1 байт	02 или 03 или 04

*Допустимые адреса и значения регистров для записи

Адрес	Действие	Возможные значения
0x0100 – 0x01FF	карта регистров	0x0200 – 0x228
0x1000 – 0x103F	конфигурация уставок	структура UST_CONFIG
0x1040 – 0x104F	конфигурация ТУ	структура TU_MASK
0x7FD0 – 0x7FD1	пароль на уставки	0x00000000 – 0xFFFFFFFF
0x7FD2 – 0x7FD3	пароль на счётчики и журналы	0x00000000 – 0xFFFFFFFF
0x7FE0	установка ТУ	0x0000 – 0x0007[1]
0x7FE1	защитный регистр для установки ТУ	0x0039
0x7FE2	время удержания ТУ1, с	0 – 255
0x7FE3	время удержания ТУ2, с	0 – 255
0x7FE4	время удержания ТУ3, с	0 – 255
0x7FE5	время удержания ТУ4, с	0 – 255
0x8000	фиксация данных	0x000F
0x8001	сброс регистра состояния	0x000F
0x8002	сброс регистра-защёлки ТУ	0x000F
0x8010	разрешение установки параметра, сбрасывается после записи любого параметра или чтения данных[2]	0x0000 (адрес) 0x0001 (скорость) 0x0002 (уставка) 0x0004 (протокол) 0x0005 (очистка счётчиков)
0x8011	установка адреса устройства	0x0001 – 0x00F7
0x8012	установка скорости текущего канала	0x0001 (19200 бит/с) 0x0002 (9600 бит/с) 0x0003 (4800 бит/с) 0x0004 (2400 бит/с) 0x0005 (1200 бит/с) 0x0011 (38400 бит/с) 0x0012 (57600 бит/с) 0x0013 (115200 бит/с)
0x8013	установка протокола интерфейса текущего канала[3]	0x0001 (FT3) 0x0002 (MODBUS)
0x8024	сброс счётчиков энергии[4]	0x000F
0x8025	сброс счётчиков ТС[4]	0x000F

[1]Запись 1 в соответствующий бит регистра включает ТУ, запись 0 - выключает.

[2]Непосредственно перед изменением параметра (например, скорости) необходимо предварительно записать константу (например, 1 для скорости) в регистр 0x8010.

[3]Выполняется с нажатой кнопкой  .

[4]См. «Очистка счётчиков энергии, счётчиков ТС».

Пример. Выполнение фиксации всех регистров таблицы 3.

Запрос		Ответ	
Поле	Hex	Поле	Hex
Адрес	01	Адрес	01
Код функции	06	Код функции	06
Адрес регистра ст.	80	Адрес регистра ст.	80

Продолжение примера

Запрос		Ответ	
Поле	Hex	Поле	Hex
Адрес регистра мл.	00	Адрес регистра мл.	00
Значение регистра ст.	00	Значение регистра ст.	00
Значение регистра мл.	0F	Значение регистра мл.	0F
CRC ст.	E0	CRC ст.	E0
CRC мл.	0E	CRC мл.	0E

0x8000 – Адрес регистра фиксации данных

0x000F – Маска фиксируемых данных

07H ЧТЕНИЕ РЕГИСТРА СТАТУСА

Эта функция используется для чтения восьми статусных битов в удалённом устройстве. Нормальный ответ содержит 1 байт статусного регистра.

Запрос

Код функции	1 байт	0x07

Ответ

Код функции	1 байт	0x07
Данные регистра	1 байт	0x00 – 0xFF

Сообщение об ошибке

Код ошибки	1 байт	0x87
Причина ошибки	1 байт	04

Байт регистра статуса равен младшему байту в структуре SENSORSTATE

Пример запроса на чтение статусного регистра

Запрос		Ответ	
Поле	Hex	Поле	Hex
Адрес	01	Адрес	01
Код функции	07	Код функции	07
CRC ст.	41	Данные регистра	C1
CRC мл.	E2	CRC ст.	E3
		CRC мл.	A0

0xC1 - данные регистра состояния.

08H ЧТЕНИЕ РЕГИСТРОВ ДИАГНОСТИКИ

Эта функция обеспечивает проверку коммуникации между главным и подчинённым и выявляет внутренние ошибки в подчинённом. Широкое вещание не поддерживается.

Функция использует два байта кода подфункции в запросе для определения типа диагностики. Подчинённый возвращает оба кода функции и подфункции в ответе.

Запрос

Код функции	1 байт	0x08
Код подфункции	2 байта	0x00XX*
Поле данных	2 байта	

Ответ

Код функции	1 байт	0x08
Код подфункции	2 байта	0x00XX*
Поле данных	2 байта	

*см. в таблице диагностических подфункций функции 08

Сообщение об ошибке

Код ошибки	1 байт	0x88
Причина ошибки	1 байт	01 или 03 или 04

Таблица диагностических подфункций функции 08

Код подфункции (Dec)	Применение
00	эхо, возвращает принятые данные
01	инициализация UART
02	возвращает регистр диагностики
04	переводит в режим прослушивания сети
10	очистка счетчиков и регистра диагностики
11	возвращает счетчик принятых сообщений
12	возвращает счетчик ошибок CRC и паритета
13	возвращает счетчик неправильных запросов
14	возвращает счетчик запросов устройства и широковещательных запросов
15	возвращает счетчик запросов без ответа
16	возвращает счетчик NAK
17	возвращает счетчик ответов «занят»
18	возвращает счетчик ошибок переполнения буфера
20	очищает счетчик и флаг ошибок переполнения буфера

00 Возвращает принятые данные. Ответ идентичен запросу.

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 00	Любые 2 байта	Повтор данных запроса

01 Инициализируется UART, все коммуникационные счётчики очищаются. Если порт находится в режиме прослушивания сети, то ответ не возвращается, но порт переходит в обычный режим. Если поле данных содержит FF00h, то происходит очистка коммуникационного журнала событий.

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 01	00 00	Повтор данных запроса
00 01	FF 00	Повтор данных запроса

02 Возвращает содержимое регистра диагностики.

Младший байт регистра диагностики равен байту регистра статуса устройства. Старший байт содержит дополнительные флаги состояния устройства.

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 02	00 00	Содержимое регистра статуса

04 Устанавливает устройство в режим прослушивания сети. Все сообщения, адресуемые подчинённому или широковещательно, отслеживаются, но не выполняется никаких действий и ответы не возвращаются. Только одна функция может быть выполнена – рестарт UART, что вернёт устройство в нормальный режим.

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 04	00 00	не возвращается

10 Очистка всех счётчиков и регистра диагностики

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 0A	00 00	Эхо данных запроса

11 Возвращает счётчик принятых сообщений после последнего рестарта UART, операции очистки счётчиков или включения питания.

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 0B	00 00	Счётчик принятых сообщений

12 Возвращает счётчик ошибок CRC после последнего рестарта, операции очистки счётчиков или включения питания.

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 0C	00 00	Счётчик ошибок CRC

13 Возвращает счётчик сообщений об ошибках, насчитанных после последнего рестарта, операции очистки счётчиков или включения питания.

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 0D	00 00	Счётчик неправильных запросов

14 Возвращает счетчик запросов, адресованных устройству и широковещательных запросов, насчитанных после последнего рестарта, операции очистки счётчиков или включения питания.

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 0E	00 00	Счётчик запросов

15 Возвращает счетчик запросов подчинённому, которые остались без ответа, насчитанных после последнего рестарта, операции очистки счётчиков или включения питания.

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 0F	00 00	Счётчик запросов без ответа

16 Возвращает счетчик сообщений, адресованных устройству, для которых был возвращён ответ с сообщением об ошибке типа Negative Acknowledge (NAK)

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 10	00 00	Счётчик NAK

17 Возвращает счетчик ответов «занят»

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 11	00 00	Счётчик ответов «занят»

18 Возвращает счетчик ошибок переполнения буфера

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 12	00 00	Счётчик переполнения

20 Очищает счётчик переполнений и флаг ошибок переполнения буфера

Подфункция	Поле данных (запрос)	Поле данных (ответ)
00 14	00 00	00 00

0FH УСТАНОВКА НЕСКОЛЬКИХ ВЫХОДОВ ТУ

Функция устанавливает каждый выход ТУ в одно из состояний включено/выключено. При широковещательной передаче функция устанавливает подобные выходы во всех подчинённых устройствах.

Запрос содержит стартовый выход ТУ для установки и поле данных. Выходы ТУ адресуются с нуля: выходы 1-8 адресуется как 0-7. Состояние, в которое необходимо установить выходы, начиная со стартового, передаётся в поле данных. ‘1’ в соответствующей битовой позиции означает включение ТУ. ‘0’ – выключение ТУ.

Запрос

Код функции	1 байт	0x0F
Стартовый адрес	2 байта	0x0000-0x0007
Количество выходов	2 байта	1 – 8
Количество байт	1 байт	1 – 2*
Состояние выходов	1 – 2*	

* Количество выходов / 8, округлённое до целого вверх

Ответ

Код функции	1 байт	0x0F
Стартовый адрес	2 байта	0x0000-0x0007
Количество выходов	2 байта	1 – 8

Сообщение об ошибке

Код ошибки	1 байт	0x8F
Причина ошибки	1 байт	02 или 03 или 04

Пример. ТУ1, ТУ2 – включить, ТУ3, ТУ4 – выключить.

Запрос		Ответ	
Поле	Hex	Поле	Hex
Адрес	01	Адрес	01
Код функции	0F	Код функции	0F
Адрес ТУ ст.	00	Адрес ТУ ст.	00
Адрес ТУ мл.	00	Адрес ТУ мл.	00
Количество ст.	00	Количество ст.	00
Количество мл.	04	Количество мл.	04
Количество байт.	01	CRC ст.	54
Состояние выходов	03	CRC мл.	08
CRC ст.	7E		
CRC мл.	97		

Нормальный ответ возвращает адрес удалённого устройства, код функции, начальный адрес и количество установленных выходов.

10H ЗАПИСЬ НЕСКОЛЬКИХ РЕГИСТРОВ

Эта функция используется для записи непрерывного блока регистров (от 1 до 123 регистров) в удалённом устройстве. Запрос содержит стартовый адрес регистра, количество записываемых регистров, число передаваемых байт и данные регистров. Нормальный ответ возвращает код функции, стартовый адрес и количество записанных регистров.

Запрос

Код функции	1 байт	0x10
Стартовый адрес	2 байта	**
Количество регистров	2 байта	0x0001 – 0x007B
Количество байт	1 байт	2 x N*
Значения регистров	N* x 2 байта	

*N – число регистров

Ответ

Код функции	1 байт	0x10
Стартовый адрес	2 байта	**
Количество регистров	2 байта	0x0001 – 0x007B

Сообщение об ошибке

Код ошибки	1 байт	0x90
Причина ошибки	1 байт	02 или 03 или 04

**Допустимые адреса и значения регистров для записи см. в таблице, указанной в описании функции 06.

ИЗМЕНЕНИЕ ПАРОЛЕЙ

Для изменения пароля доступа к уставкам или пароля доступа к счётчикам и журналам необходимо выполнить последовательно три запроса к устройству.

1. Разрешение установки пароля путём записи константы в регистр функцией 06h.
2. Запись старого пароля в 2 регистра пароля, используя функцию 10h. См. в таблице, указанной в описании функции 06h.
3. Запись нового пароля в те же регистры пароля, используя функцию 10h.

При несовпадении отправленного старого пароля с хранящимся в ПЦ, либо неверной последовательности действий на третий запрос будет выдано сообщение об ошибке с причиной ошибки 04. Пароль, естественно, в случае любых ошибок останется прежний.

ЗАПИСЬ УСТАВОК

Запись уставок производится в следующей последовательности.

1. Разрешить доступ к уставкам путём записи константы в регистр функцией 06h.
2. Записать пароль доступа к уставкам (2 регистра), используя функцию 10h.
3. Записать новую конфигурацию уставок, см. таблицу, указанную в описании функции 06h.

При несовпадении отправленного пароля с хранящимся в ПЦ, либо неверной последовательности действий на третий запрос будет выдано сообщение об ошибке с причиной ошибки 04.

ОЧИСТКА СЧЁТЧИКОВ ЭНЕРГИИ, СЧЁТЧИКОВ ТС.

Производится в следующей последовательности.

1. Разрешить доступ к счётчикам и журналам путём записи константы 0x0005 в регистр 0x8010 функцией 06h.
2. Записать пароль доступа к счётчикам и журналам (2 регистра), используя функцию 10h.
3. Записать константу в соответствующий нужному действию регистр, см. таблицу, указанную в описании функции 06h.

При несовпадении отправленного пароля с хранящимся в ПЦ, либо неверной последовательности действий на третий запрос будет выдано сообщение об ошибке с причиной ошибки 04.

2BH/0EH (43/14) ЧТЕНИЕ ИДЕНТИФИКАЦИИ УСТРОЙСТВА

Эта функция используется для чтения идентификационной и дополнительной информации конфигурации устройства.

Запрос

Код функции	1 байт	0x43
Считывание идентификации	1 байт	0x0E
Тип считывания	1 байт	0x01 или 0x02
Id объекта	1 байт	0x00

Ответ

Код функции	1 байт	0x43
Считывание идентификации	1 байт	0x0E
Тип считывания	1 байт	0x01 или 0x02
Уровень соответствия	1 байт	0x02
Кадр последовательности	1 байт	0x00
Резерв	1 байт	0x00
Количество объектов	1 байт	
номер объекта	1 байт	0x00
длина объекта	1 байт	
ASCII строка	длина объекта	зависит от ID объекта
…	…	…
номер n объекта	1 байт	n
длина n объекта	1 байт	
ASCII строка n объекта	длина объекта	зависит от ID n объекта

В случае ошибки при обработке запроса выдается специальный исключительный ответ

Код ошибки	1 байт	0xAB (0x2B + 0x80)
Считывание идентификации	1 байт	0x0E
Причина ошибки	1 байт	01

В ответе может содержаться либо упрощённая идентификация, либо полная. Упрощённый ответ содержит только первые три базовых объекта с 0x00 по 0x02. Полный ответ содержит все семь объектов идентификации устройства с 0x00 по 0x06.

Номер объекта	Название объекта	строка ASCII	Категория объекта
0x00	торговое наименование	“ООО “НПП Электромеханика”	базовый
0x01	код изделия	“06”	
0x02	версия ПО	"35"	
0x03	электронный адрес производителя	"www.npp-em.ru"	стандарт
0x04	наименование изделия	"ПЦ6806-03/31"	
0x05	наименование модели	"Преобразователь измерительный цифровой"	
0x06	серийный номер	"0001000052"	

РЕГИСТРЫ ПЦ6806-03

Адреса регистров, приведены в таблице 3.

Физически регистры измеренных параметров расположены в области с адреса 0x0200. Начиная с адреса 0x8000, размещены регистры конфигурации и управления. Для удобства и совместимости с другими устройствами в MODBUS сети, пространство 0x0000–0x00FF используется для отображения физических регистров. То есть, физическим регистрам можно назначать адреса-синонимы из диапазона 0x0000–0x00FF. Для этого необходимо в карту регистров (0x0100–0x01FF) записать адреса физических регистров, в любой последовательности, как это удобно для конкретной задачи. В карту регистров нельзя записать адреса вне диапазона физических регистров. Например, если записать в регистр 0x0100 значение 0x0200, то мгновенное действующее значение напряжения фазы A (Ua) можно читать и по адресу 0x0000, и, конечно, по неизменному 0x0200. В карте регистров значение, записанное по адресу 0x0100, соответствует отображаемому регистру по адресу 0x0000. Значение, записанное по адресу 0x0101, соответствует отображаемому регистру по адресу 0x0001, и так далее.

Регистры, фиксированные функцией 06h, и читаемые функцией 03h, доступны по тем же адресам, что и текущие. Их содержимое можно получить и из физической области (начиная с адреса 0x0200), и из области отображаемых регистров (0x0000–0x00FF).

Карта регистров (0x0100–0x01FF) едина и для фиксированных, и для текущих регистров.

Таблица 3

Имя регистра	Адрес	Чтение/ Запись
Отображаемые регистры в соответствии с картой регистров	0x0000	+/-
	…	+/-
	0x00FF	+/-
Карта регистров	0x0100	+/+
	…	+/+
	0x01FF	+/+
Мгновенное действующее значение напряжения фазы A (Ua)	0x0200	+/-
Мгновенное действующее значение напряжения фазы B (Ub)	0x0201	+/-
Мгновенное действующее значение напряжения фазы C (Uc)	0x0202	+/-
Мгновенное действующее значение тока фазы A (Ia)	0x0203	+/-
Мгновенное действующее значение тока фазы B (Ib)	0x0204	+/-
Мгновенное действующее значение тока фазы C (Ic)	0x0205	+/-
Мгновенная активная мощность трехфазной системы (P_) - младшее слово	0x0206	+/-
Мгновенная активная мощность трехфазной системы (P_) - старшее слово	0x0207	+/-
Мгновенная активная мощность фазы A (Pa)	0x0208	+/-
Мгновенная активная мощность фазы B (Pb)	0x0209	+/-
Мгновенная активная мощность фазы C (Pc)	0x020A	+/-
Мгновенная реактивная мощность трехфазной системы (Q_) - мл. слово	0x020B	+/-
Мгновенная реактивная мощность трехфазной системы (Q_) - ст. слово	0x020C	+/-
Мгновенная реактивная мощность фазы A (Qa)	0x020D	+/-
Мгновенная реактивная мощность фазы B (Qb)	0x020E	+/-
Мгновенная реактивная мощность фазы C (Qc)	0x020F	+/-
Мгновенная полная мощность трехфазной системы (S_) - младшее слово	0x0210	+/-

Имя регистра	Адрес	Чтение/ Запись
Мгновенная полная мощность трехфазной системы (S_) - старшее слово	0x0211	+/-
Мгновенная полная мощность фазы A (Sa)	0x0212	+/-
Мгновенная полная мощность фазы B (Sb)	0x0213	+/-
Мгновенная полная мощность фазы C (Sc)	0x0214	+/-
Мгновенное действующее значение межфазного напряжения (Uab)	0x0215	+/-
Мгновенное действующее значение межфазного напряжения (Ubc)	0x0216	+/-
Мгновенное действующее значение межфазного напряжения (Uac)	0x0217	+/-
Мгновенное действующее напряжение нулевой последовательности (3U0)	0x0218	+/-
Мгновенное действующее значение тока нулевой последовательности (3I0)	0x0219	+/-
Мгновенное среднее значение напряжения (U)	0x021A	+/-
Мгновенное среднее значение тока (I)	0x021B	+/-
Интегрированное действующее значение напряжения фазы A (Ura)	0x021C	+/-
Интегрированное действующее значение напряжения фазы B (Urb)	0x021D	+/-
Интегрированное действующее значение напряжения фазы C (Urc)	0x021E	+/-
Интегрированное действующее значение тока фазы A (Ira)	0x021F	+/-
Интегрированное действующее значение тока фазы B (Irb)	0x0220	+/-
Интегрированное действующее значение тока фазы C (Irc)	0x0221	+/-
Интегрированная активная мощность трехфазной системы (Pr_) – мл. слово	0x0222	+/-
Интегрированная активная мощность трехфазной системы (Pr_) – ст. слово	0x0223	+/-
Интегрированная активная мощность фазы A (Pra)	0x0224	+/-
Интегрированная активная мощность фазы B (Prb)	0x0225	+/-
Интегрированная активная мощность фазы C (Prc)	0x0226	+/-
Интегрированная реактивная мощность трехфазной системы (Qr_) - мл. слово	0x0227	+/-
Интегрированная реактивная мощность трехфазной системы (Qr_) - ст. слово	0x0228	+/-
Интегрированная реактивная мощность фазы A (Qra)	0x0229	+/-
Интегрированная реактивная мощность фазы B (Qrb)	0x022A	+/-
Интегрированная реактивная мощность фазы C (Qrc)	0x022B	+/-
Интегрированная полная мощность трехфазной системы (Sr_) – мл. слово	0x022C	+/-
Интегрированная полная мощность трехфазной системы (Sr_) – ст. слово	0x022D	+/-
Интегрированная полная мощность фазы A (Sra)	0x022E	+/-
Интегрированная полная мощность фазы B (Srb)	0x022F	+/-
Интегрированная полная мощность фазы C (Src)	0x0230	+/-
Интегрированное действующее значение межфазного напряжения (Urab)	0x0231	+/-
Интегрированное действующее значение межфазного напряжения (Urbc)	0x0232	+/-
Интегрированное действующее значение межфазного напряжения (Urac)	0x0233	+/-
Интегрированное действующее напряжение нулевой последовательности (3Ur0)	0x0234	+/-
Интегрированное действующее значение тока нулевой последовательности (3Ir0)	0x0235	+/-
Интегрированное среднее значение напряжения (Ur)	0x0236	+/-
Интегрированное среднее значение тока (Ir)	0x0237	+/-
Частота напряжения фазы A (F)	0x0238	+/-
Температура	0x0239	+/-
Энергия активная потреблённая (Ep+) - мл. слово	0x023A	+/-
Энергия активная потреблённая (Ep+) - ст. слово	0x023B	+/-
Энергия активная возвращённая (Ep-) - мл. слово	0x023C	+/-
Энергия активная возвращённая (Ep-) - ст. слово	0x023D	+/-

Имя регистра	Адрес	Чтение/ Запись
Энергия реактивная потреблённая (ErL-) - мл. слово	0x023E	+/-
Энергия реактивная потреблённая (ErL-) - ст. слово	0x023F	+/-
Энергия реактивная потреблённая (ErC) - мл. слово	0x0240	+/-
Энергия реактивная потреблённая (ErC) - ст. слово	0x0241	+/-
Счётчик ТС1, мл. слово	0x0242	+/-
Счётчик ТС1, ст. слово	0x0243	+/-
Счётчик ТС2, мл. слово	0x0244	+/-
Счётчик ТС2, ст. слово	0x0245	+/-
Время в формате CP56 (занимает 4 слова)[1]	0x0246	+/-
	0x0247	+/-
	0x0248	+/-
	0x0249	+/-
Активные уставки, 16 бит	0x024A	+/-
Регистр состояния, 16 бит	0x024B	+/-
Регистр-защёлка ТУ, 16 бит	0x024C	+/-
Конфигурация уставок		
Конфигурация уставки 1 (структура UST_CONFIG)	0x1000	+/+
	0x1001	+/+
	0x1002	+/+
	0x1003	+/+
Конфигурация уставки 2 (структура UST_CONFIG)	0x1004	+/+
	0x1005	+/+
	0x1006	+/+
	0x1007	+/+
Конфигурация уставки 3 (структура UST_CONFIG)	0x1008	+/+
	0x1009	+/+
	0x100A	+/+
	0x100B	+/+
Конфигурация уставки 4 (структура UST_CONFIG)	0x100C	+/+
	0x100D	+/+
	0x100E	+/+
	0x100F	+/+
Конфигурация уставки 5 (структура UST_CONFIG)	0x1010	+/+
	0x1011	+/+
	0x1012	+/+
	0x1013	+/+
Конфигурация уставки 6 (структура UST_CONFIG)	0x1014	+/+
	0x1015	+/+
	0x1016	+/+
	0x1017	+/+
Конфигурация уставки 7 (структура UST_CONFIG)	0x1018	+/+
	0x1019	+/+
	0x101A	+/+
	0x101B	+/+
Конфигурация уставки 8 (структура UST_CONFIG)	0x101C	+/+

Имя регистра	Адрес	Чтение/ Запись
	0x101D	+/+
	0x101E	+/+
	0x101F	+/+
Конфигурация уставки 9 (структура UST_CONFIG)	0x1020	+/+
	0x1021	+/+
	0x1022	+/+
	0x1023	+/+
Конфигурация уставки 10 (структура UST_CONFIG)	0x1024	+/+
	0x1025	+/+
	0x1026	+/+
	0x1026	+/+
Конфигурация уставки 11 (структура UST_CONFIG)	0x1028	+/+
	0x1029	+/+
	0x102A	+/+
	0x102B	+/+
Конфигурация уставки 12 (структура UST_CONFIG)	0x102C	+/+
	0x102D	+/+
	0x102E	+/+
	0x102F	+/+
Конфигурация уставки 13 (структура UST_CONFIG)	0x1030	+/+
	0x1031	+/+
	0x1032	+/+
	0x1033	+/+
Конфигурация уставки 14 (структура UST_CONFIG)	0x1034	+/+
	0x1035	+/+
	0x1036	+/+
	0x1037	+/+
Конфигурация уставки 15 (структура UST_CONFIG)	0x1038	+/+
	0x1039	+/+
	0x103A	+/+
	0x103B	+/+
Конфигурация уставки 16 (структура UST_CONFIG)	0x103C	+/+
	0x103D	+/+
	0x103E	+/+
	0x103F	+/+
Конфигурация ТУ		
Конфигурация ТУ1 (структура TU_MASK)	0x1040	+/+
	0x1041	+/+
	0x1042	+/+
	0x1043	+/+
Конфигурация ТУ2 (структура TU_MASK)	0x1044	+/+
	0x1045	+/+
	0x1046	+/+
	0x1047	+/+
Конфигурация ТУ3 (структура TU_MASK)	0x1048	+/+

Имя регистра	Адрес	Чтение/ Запись
	0x1049	+/+
	0x104A	+/+
	0x104B	+/+
Конфигурация ТУ4 (структура TU_MASK)	0x104C	+/+
	0x104D	+/+
	0x104E	+/+
	0x104F	+/+
логика вкл-выкл ТУ5[1]	0x1050	+/+
	0x1051	+/+
	0x1052	+/+
	0x1053	+/+
логика вкл-выкл ТУ6[1]	0x1054	+/+
	0x1055	+/+
	0x1056	+/+
	0x1057	+/+
логика вкл-выкл ТУ7[1]	0x1058	+/+
	0x1059	+/+
	0x105A	+/+
	0x105B	+/+
логика вкл-выкл ТУ8[1]	0x105C	+/+
	0x105D	+/+
	0x105E	+/+
	0x105F	+/+
Информация об устройстве (структура Sns_TYPE)		
Модель = 0x0668	0x2000	+/-
Номер модификации и ревизии Mod_N_R = 0xXX04	0x2001	+/-
Биты исполнения и версия программы M_Pr_V	0x2002	+/-
Серийный номер, мл. слово	0x2003	+/-
Серийный номер, ст. слово	0x2004	+/-
Контрольная сумма ПО общая, мл. слово	0x2005	+/-
Контрольная сумма ПО общая, ст. слово	0x2006	+/-
Контрольная сумма метрологически значимой части ПО, мл. слово	0x2007	+/-
Контрольная сумма метрологически значимой части ПО, ст. слово	0x2008	+/-
Пароль на уставки, мл. слово	0x7FD0	-/+
Пароль на уставки, ст. слово	0x7FD1	-/+
Пароль на счётчики и журналы, мл. слово	0x7FD2	-/+
Пароль на счётчики и журналы, ст. слово	0x7FD3	-/+
Установка ТУ	0x7FE0	-/+
Защитный регистр для установки ТУ	0x7FE1	-/+
Время удержания ТУ1, с	0x7FE2	-/+
Время удержания ТУ2, с	0x7FE3	-/+
Время удержания ТУ3, с	0x7FE4	-/+
Время удержания ТУ4, с	0x7FE5	-/+

Имя регистра	Адрес	Чтение/ Запись
Защёлка для фиксации данных	0x8000	-/+
Сброс регистра состояния	0x8001	-/+
Сброс регистра-защёлки ТУ	0x8002	-/+
Разрешение установки параметра	0x8010	-/+
Адрес устройства	0x8011	+/+
Скорость интерфейса	0x8012	+/+
Протокол связи канала	0x8013	+/+
Установка времени[1]	0x8020–0x8023	-/+
Сброс счётчиков энергии	0x8024	-/+
Сброс счётчиков ТС	0x8025	-/+
Очистка журнала[1]	0x8026	-/+

[1]В данном изделии не реализовано. Зарезервировано для совместимости с другими устройствами.

ПРИМЕРЫ И ИНТЕРПРЕТАЦИЯ. ТИПЫ ДАННЫХ.

Правила перевода значений регистров в единицы физических величин

Физическая величина	тип данных регистра	формула перевода
Ток, А	unsigned short	Значение_регистра/1000.0
Напряжение, В	unsigned short	Значение_регистра/10.0
Мощность активная, Вт	short	Значение_регистра/10.0
Мощность реактивная, вар	short	Значение_регистра/10.0
Мощность полная, Вт	unsigned short	Значение_регистра/10.0
Частота, Гц	unsigned short	2457600.0/Значение_регистра
Энергия, Вт*час	unsigned long	Значение_регистра
Температура, °С	short	Значение_регистра/32.0
Мощность трехфазной системы, Вт	long	Значение_регистра/100.0

Примеры

Регистр	адрес	Значение HEX(dec)	Интерпретация
Мгновенное действующее значение тока фазы А (Ia)	0x0203	0x03E8 (1000)	1.000 А
Мгновенное действующее значение напряжения фазы А (Ua)	0x0200	0x0241 (577)	57.7 В
Мгновенная активная мощность фазы В (Pb)	0x0209	0xFC15 (-1003)	-100.3 Вт
Частота	0x0238	0xC000 (49152)	2457600.0/49152=50.0 Гц
Температура	0x0239	0x03D0 (976)	976/32=30.5 °С

НЕКОТОРЫЕ ТИПЫ ДАННЫХ

SENSORSTATE

```
// Регистр состояния
typedef struct _SENSORSTATE
{
 unsigned char ProcReset1     :1;   // Сброс процессора
 unsigned char ErrWriteAddrB  :1;   // Ошибка КС адреса, скорости
 unsigned char ErrWriteData   :1;   // Ошибка индикатора
 unsigned char ErrCS_K        :1;   // Ошибка КС коэффициентов
 unsigned char ErrCS_U        :1;   // Ошибка КС уставок
 unsigned char ErrCS_C        :1;   // Ошибка КС счетчиков
 unsigned char ErrFrame       :1;   // Ошибка кадровой синхронизации
 unsigned char ErrCRC         :1;   // Ошибка CRC пакета

 unsigned char ProcReset2     :1;   // АЦП был переинициализирован
 unsigned char ErrStatusCRC   :1;   // Резерв
 unsigned char ErrProcExchange:1;   // Резерв
 unsigned char Reserv1        :1;   // Резерв
 unsigned char ErrProcAnswer  :1;   // Резерв
 unsigned char ErrTemperatureDevice:1;// Ошибка датчика температуры
 unsigned char Reserv3        :1;   // Резерв
 unsigned char Reserv4        :1;   // Резерв
}SENSORSTATE;
```

SNS_TYPE

```
// Информация об устройстве
typedef struct
{
      unsigned char     Mod_L;          // Модель датчика     const=0x68
      unsigned char     Mod_H;          // Модель датчика     const=0x06
      unsigned char     Mod_Numb;       // Номер модификации  const=0x04
      unsigned char  PowerVType:4;  // Тип питания
      unsigned char  InputVType:4;  // Тип входного напряжения
      unsigned char     Mod_Ver;        // Модификация модели
      unsigned char     Prog_Ver;       // Версия программы
      unsigned long     Ser_Numb;   // Серийный номер
} Sns_TYPE;
/*
Расшифровка поля InputVType
InputVType     Расшифровка
1     Количество фаз = 3  Входное напряжение =  60V  Входной ток = 1A
2     Количество фаз = 2  Входное напряжение = 100V  Входной ток = 1A
3     Количество фаз = 3  Входное напряжение =  60V  Входной ток = 5A
4     Количество фаз = 2  Входное напряжение = 100V  Входной ток = 5A
5     Количество фаз = 3  Входное напряжение = 220V  Входной ток = 5A

Расшифровка поля PowerVType
PowerVType  Расшифровка
1     ~80…260 В, =100…300 В;
2     Питание от измерительной цепи
*/
```

UST_CONFIG

```
typedef struct
{  //
   unsigned char Type;              // Тип уставки
   unsigned char OnOffTU;           // 1-вкл; 0-выкл ТУ
   unsigned short Value;            // Значение для данного типа уставки
   unsigned short TimeTo;           // Время с момента возникновения условия на
                                    // срабатывание уставки до ее фактического
                                    // срабатывания(1/256 с)
   unsigned short ReturnValue;      // Резерв
} UST_CONFIG;

//Типы уставок
#define UST_NOTYPE                  0 // Уставка отсутствует
#define UST_MAX_CURRENT             1 // Уставка по макс. току
#define UST_MIN_CURRENT             2 // Уставка по мин. току
#define UST_MAX_VOLTAGE             3 // Уставка по макс. напряжению
#define UST_MIN_VOLTAGE             4 // Уставка по мин. напряжению
#define UST_MAX_ACTIVE_POWER        5 // Уставка по макс. активной мощности
#define UST_MIN_ACTIVE_POWER        6 // Уставка по мин. активной мощности
#define UST_MAX_REACTIVE_POWER      7 // Уставка по макс. реактивной
#define UST_MIN_REACTIVE_POWER      8 // Уставка по мин. реактивной
#define UST_MAX_FREQUENCY           9 // Уставка по макс. частоте
#define UST_MIN_FREQUENCY          10 // Уставка по мин. частоте
#define UST_MAX_NULL_CURRENT       11 // Уставка по макс. току нулевой
                                      // последовательности
#define UST_MIN_NULL_CURRENT       12 // Уставка по мин. току нулевой
                                      // последовательности
#define UST_MAX_NULL_VOLTAGE       13 // Уставка по макс. напряжению нулевой
                                      // последовательности
#define UST_MIN_NULL_VOLTAGE       14 // Уставка по мин. напряжению нулевой
                                      // последовательности
#define UST_MAX_TEMP               15// Уставка по макс. внутренней температуре
#define UST_MIN_TEMP               16// Уставка по мин. внутренней температуре
#define UST_REMOTE_SIGNALING       128//Уставки по ТС
```

TU_MASK

```
typedef struct
{
  unsigned short no;          // маска no
  unsigned short and;         // маска and
  unsigned short or;          // маска or
  unsigned short TimeAfter;   //Время удержания ТУ (0 – бесконечное)
}                                                                  TU_MASK;
```

ПРИЛОЖЕНИЕ А. СООБЩЕНИЯ ОБ ОШИБКАХ

Одна из четырех ситуаций может иметь место при запросе главного к подчиненному:

- Если подчиненное устройство приняло запрос без коммуникационных ошибок, и может нормально распознать запрос, оно возвращает нормальный ответ.
- Если подчиненное устройство не приняло запрос, ответ не возвращается. Главный ожидает ответа на запрос в течение определенного таймаута.
- Если подчиненный принял запрос, но обнаружил коммуникационную ошибку (паритет, ошибка контрольной суммы), то ответ не возвращается. Главный ожидает ответа на запрос в течение определенного таймаута.
- Если подчиненный принял запрос без коммуникационной ошибки, но не может выполнить затребованную функцию(например, чтение несуществующих выходов или регистров), подчиненный возвращает сообщение об ошибке и ее причинах.

Сообщение об ошибке имеет два поля, которые отличаются от полей нормального ответа:

ПОЛЕ КОДА ФУНКЦИИ: В нормальном ответе, подчиненный повторяет код функции содержащийся в поле кода функции запроса. Во всех кодах функций старший значащий бит установлен в 0. При возврате сообщения об ошибке подчиненный устанавливает этот бит в 1.

По установленному старшему биту в коде функции главный распознает сообщение об ошибке, и может проанализировать поле данных сообщения.

ПОЛЕ ДАННЫХ: В нормальном ответе, подчиненный может возвращать данные или статистику в поле данных (любую информацию, которая затребована в запросе). В сообщении об ошибке, подчиненный возвращает код ошибки в поле данных.

Ниже показан пример запроса главного и сообщения об ошибке подчиненного:

Запрос		Ответ	
Поле	Hex	Поле	Hex
Адрес	01	Адрес	01
Код функции	04	Код ошибки	84
Начальный адрес ст.	00	Причина ошибки	02
Начальный адрес мл.	2E	CRC ст.	C2
Кол-во регистров ст.	00	CRC мл.	C1
Кол-во регистров мл.	01		
CRC ст.	51		
CRC мл.	C3		

В данном примере главный требует прочитать несуществующий регистр с адресом 0x002E. Подчиненный возвращает сообщение об ошибке с кодом ошибки (02). Этот код специфицирует несуществующий адрес данных в подчиненном.

Список кодов ошибок представлен ниже.

Код	Название	Описание
01	ILLEGAL FUNCTION	Принятый код функции не поддерживается на подчиненном.
02	ILLEGAL DATA ADDRESS	Адрес данных указанный в запросе не доступен данному подчиненному.
03	ILLEGAL DATA VALUE	Величина, содержащаяся в поле данных запроса, является недопустимой величиной для подчиненного.
04	SLAVE DEVICE FAILURE	Невосстанавливаемая ошибка имела место, пока подчиненный пытался выполнить затребованное действие.

ПРИЛОЖЕНИЕ В. ГЕНЕРАЦИЯ CRC

CRC это 16-ти разрядная величина, т.е. два байта. CRC вычисляется передающим устройством и добавляется к сообщению. Принимающее устройство также вычисляет CRC в процессе приема и сравнивает вычисленную величину с полем контрольной суммы пришедшего сообщения. Если суммы не совпали - то имеет место ошибка.

16-ти битовый регистр CRC предварительно загружается числом 0xFFFF. Процесс начинается с добавления байтов сообщения к текущему содержимому регистра. Для генерации CRC используются только 8 бит данных. Старт и стоп биты, бит паритета, если он используется, не учитываются в CRC.

В процессе генерации CRC, каждый 8-ми битовый символ складывается по ИСКЛЮЧАЮЩЕМУ ИЛИ с содержимым регистра. Результат сдвигается в направлении младшего бита, с заполнением 0 старшего бита. Младший бит извлекается и проверяется. Если младший бит равен 1, то содержимое регистра складывается с определенной ранее, фиксированной величиной, по ИСКЛЮЧАЮЩЕМУ ИЛИ. Если младший бит равен 0, то ИСКЛЮЧАЮЩЕЕ ИЛИ не делается.

Этот процесс повторяется, пока не будет сделано 8 сдвигов. После последнего (восьмого) сдвига, следующий байт складывается с содержимым регистра и процесс повторяется снова. Финальное содержание регистра, после обработки всех байтов сообщения и есть контрольная сумма CRC.

Алгоритм генерации CRC

1. 16-ти битовый регистр загружается числом 0xFFFF, и используется далее как регистр CRC.

2. Первый байт сообщения складывается по ИСКЛЮЧАЮЩЕМУ ИЛИ с содержимым регистра CRC. Результат помещается в регистр CRC.

3. Регистр CRC сдвигается вправо(в направлении младшего бита) на 1 бит, старший бит заполняется 0.

4. (Если младший бит 0): Повторяется шаг 3 (сдвиг)

(Если младший бит 1): Делается операция ИСКЛЮЧАЮЩЕЕ ИЛИ регистра CRC и полиномиального числа A001 hex.

5. Шаги 3 и 4 повторяются восемь раз.

6. Повторяются шаги со 2 по 5 для следующего сообщения. Это повторяется до тех пор пока все байты сообщения не будут обработаны.

7. Финальное содержание регистра CRC и есть контрольная сумма.

РАЗМЕЩЕНИЕ CRC В СООБЩЕНИИ

При передаче 16 бит контрольной суммы CRC в сообщении, сначала передается младший байт, затем старший. Например, если CRC имеет значение 1241 hex :

Адрес	Функция	Счетчик байт	Данные	Данные	Данные	Данные	CRC Мл.	CRC Ст.
							41	12

ПРИМЕР

Пример функции на языке С реализующей генерацию CRC приведен ниже. Все возможные величины CRC загружены в два массива. Один массив содержит все 256 возможных комбинаций CRC для старшего байта поля CRC, другой массив содержит данные для младшего байта. Идексация CRC в этом случая обеспечивает быстрое выполнение вычислений новой величины CRC для каждого нового байта из буфера сообщения.

```
const char auchCRCHi[256] = {
    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, 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, 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};
const char auchCRCLo[256] = {
    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};

unsigned short CRC16MOD(puchMsg, usDataLen)
char *puchMsg; /* Указатель на буфер */
unsigned short usDataLen; /* Количество байтов в буфере */
{
  struct {
          char uchCRCHi;  // здесь специально такой порядок байт,
          char uchCRCLo;  // т.к. в MODBUS сначала передаётся младший байт CRC
         } uchCRC;
  unsigned uIndex;

  *(word*)&uchCRC=0xFFFF;
  while (usDataLen--)
      {
        uIndex = uchCRC.uchCRCHi^ *puchMsg++;
        uIndex&=0x00FF;
        uchCRC.uchCRCHi = uchCRC.uchCRCLo^auchCRCHi[uIndex];
        uchCRC.uchCRCLo = auchCRCLo[uIndex];
      }
  return *(word*)&uchCRC;
}
```