АО «ГМС Ливгидромаш»
303851 Россия, Орловская область, г. Ливны, ул. Мира, 231
тел. : +7(48677) 7-80-00, 7-80-03
факс : +7(48677) 7-80-80, 7-80-98
Техническая поддержка: +7(48677) 7-92-21, 7-92-12
e-mail : info@hms-livgidromash.ru
www.hms-livgidromash.ru www.grouphms.ru

УСТРОЙСТВО УПРАВЛЕНИЯ И ЗАЩИТЫ «L3.1» 012.96.00.00.00

Описание протокола MODBUS

(версия 1.02)

г. Ливны, 2025 г.

СОДЕРЖАНИЕ:

1. О документе **3**
2. Описание реализации **3**
2.1 Интерфейс **3**
2.2 Адреса устройств Modbus **3**
2.3 Команды **3**
2.4 Адресация **3**
2.5 Режим передачи **4**
3. Описание регистров **5**
3.1 Чтение состояния дискретных выходов (Read Coils). **5**
3.2 Чтение состояния дискретных входов (Read Discrete Inputs). **6**
3.3 Регистры задания параметров объекта (Holding Register). **7**
3.4 Регистры чтения данных объекта (Input Register). **12**
4. Исключительные ситуации **15**
5. Задержки между пакетами **16**
6 Генерация контрольной суммы **17**
6.1 Контрольная сумма CRC **17**

1. О документе

Документ описывает реализацию протокола RTU в устройствах управления и защиты трехфазных асинхронных электродвигателей серии L3 (далее по тексту - контроллер) производства АО «ГМС Ливгидромаш». Содержится вся необходимая информация для программистов при подключении контроллеров к SCADA системам или при создании распределенных систем автоматики.

Данный документ ссылается на следующие документы:

1. Modbus Application Protocol Specification v1.1b (www.modbus.org)
2. Modbus messaging on TCP/IP implementation Guide Rev 1.0 (www.modbus.org)
3. Modbus over Serial Line Specification & Implementation guide V1.0 (www.modbus.org)

Термины и сокращения:

RS-232 – стандарт EIA/TIA-232;
RS-485 – стандарт EIA/TIA-485 Standard.

2. Описание реализации

2.1 Интерфейс

Контроллер имеет последовательные интерфейсы RS-485 и/или RS-232. Для организации сети из двух и более приборов можно использовать преобразователь интерфейсов 232/485. Интерфейс RS-485 позволяет объединить в сеть до 128 устройств на линии длиной до 1200 м. Контроллер является ведомым (slave) устройством, отвечающим на команды с соответствующим адресом в пакете протокола.

По последовательным интерфейсам поддерживается протокол верхнего уровня Modbus с форматом пакета RTU в полном соответствии с документом «Modbus over Serial Line Specification & Implementation guide V1.0». Поддерживаются скорости передачи от 2400 бит/с до 128000 бит/с с контролем четности, 8 или 9 бит данных, 1 или 2 стоп-бита. Физический интерфейс, скорость соединения и сетевой адрес задаются при программировании контроллера. Максимальное время ожидания ответа составляет не более 100 мс.

2.2 Адреса устройств Modbus

Все устройства серии поддерживают команды Modbus в полном соответствии с синтаксисом запроса и ответа, определенным в документе «Modbus Application Protocol Specification v1.1b». Поддерживаются запросы к конкретным устройствам по их адресам, широковещательный режим не поддерживается. Адрес устройства может быть от 01h до F7h. Диапазон адресов F8h-FFh зарезервирован в стандарте Modbus.

2.3 Команды

Микроконтроллер поддерживает следующие команды:

- **01h** Чтение состояния дискретных выходов (Read Coils)
- **02h** Чтение состояния дискретных входов (Read Discrete Inputs)
- **03h** Чтение регистров настройки (Read Holding Registers)
- **04h** Чтение входных регистров (Read Input Register)
- **05h** Установка единичного дискретного выхода (Write Single Coil)
- **06h** Запись регистра настройки (Write Single Register)
- **10h** Запись нескольких регистров настройки (Write Multiple Registers)

Наличие команд 07h, 08h, 0Bh, 0Ch, 0Fh, 11h, 14h, 15h, 16h, 17h, 18h не обязательна. Поддержка конкретным устройством команды из приведенного выше списка отражается отдельно.

2.4 Адресация

Адреса запрашиваемых регистров и битов по протоколу Modbus и адреса в памяти устройства имеют однозначное табличное соответствие, но не совпадают. Таблица соответствия адресов задается программистом и должна быть отражена в документации к устройству. По одному интерфейсу может быть доступно не более 32767 байт адресуемых регистров переменных, 32767 байт регистров входов и 8192 байт, доступных через битовых переменные.

Основным способом передачи данных по протоколу Modbus является чтение или запись регистров.
Реализация протокола поддерживает как побайтную адресацию, так и пословную.
Адресация битовых массивов данных полностью соответствует стандарту Modbus. Максимально возможное количество битов, передаваемых в одном пакете, не может быть более 256.

2.5 Режим передачи

В протоколе Modbus существуют два режима передачи. Это ASCII (American Standard Code for Information Interchange) и RTU (Remote Terminal Unit). Режим выбирается пользователем, в зависимости от используемого в сети оборудования. Для каждой сети Modbus должен использоваться только один режим. Использование смешанных режимов в одной сети не допускается.

Структура кадра сообщения в режиме ASCII

Семибитовый код ASCII был разработан как универсальный код для отображения символов английского языка для телетайпов и является принятым в США стандартом для представления символов английского языка и управляющих символов, например, CR (возврат каретки) и LF (перевод строки). Наименования этих символов сохранились со времен телетайпов и сейчас просто указывает на конец кадра.

Преимуществом данного режима является то, что если в качестве ведомого устройства включить монитор, то можно увидеть на экране понятный человеку отформатированный код, который послан ведущим устройством на экран монитора. На рисунке 1 показана структура сообщения Modbus в режиме ASCII. Его начало обозначается символом «:»(3Ah), а конец - последовательностью CR/LF (два символа ASCII - 0Dh, 0Ah).

Начало	Адрес устройства	Код функции	Данные	Контрольная сумма (LRC)	Конец
1 символ (:)	2 символа	2 символа	N символов	2 символа	2 символа (CR+LF)

Рисунок 1 Структура кадра сообщения Modbus ASCII

Любой символ ASCII представляется 7 битами. Символы должны быть либо цифрами от 0 до 9, либо буквами от A до F, так как предполагается, что данные представляются в шестнадцатеричном формате, но отображаются в виде символов ASCII. Например, код функции 03 будет отображаться двумя ASCII-символами: «0»(30h) и «3»(33h). То же самое относится и к содержимому поля данных.

Одним из преимуществ режима ASCII является то, что он не предъявляет жестких требований к синхронизации. Допускается временной промежуток между символами до 1 секунды - только по истечении его генерируется сообщение о превышении лимита времени.

Структура кадра сообщения в режиме RTU

При работе в режиме RTU синхронизация имеет более важное значение, чем в режиме ASCII. В этом варианте специальный начальный символ отсутствует. Вместо этого кадр сообщения начинается с маркерного интервала, длительность которого равна времени передачи четырех символов. После истечения этого интервала передается адрес устройства, затем код функции и собственно данные. Имеются и другие отличия от кадра сообщения в режиме ASCII, как это показано на рисунке 2.

Начало	Адрес устройства	Код функции	Данные	Контрольная сумма CRC	Конец
Интервал, равный времени передачи 4-х символов	8 бит	8 бит	N*8 бит	16 бит	Интервал, равный времени передачи 4-х символов

Рисунок 2 Структура кадра сообщения Modbus RTU

Вместо контрольной суммы LRC (Longitudinal Redundancy Check - продольный контроль по избыточности) в режиме RTU используется контрольная сумма CRC (Cyclic Redundancy Check - циклический контроль по избыточности). Конец кадра отмечается маркерным интервалом, равным времени передачи четырех символов.

RTU-сообщения должны посылаться в виде непрерывного потока, и появление значительного временного интервала между смежными символами рассматривается как окончание сообщения. Сообщения в этом режиме весьма компактны и более эффективны, чем ASCII, с точки зрения их передачи. Поэтому режим RTU является более популярным.

3. Описание регистров.

3.1 Чтение состояния дискретных выходов (Read Coils).

Чтение дискретных выходов производится командой **01h (Read Coils).**

Функция позволяет пользователю получить статус (1/0) логических ячеек. Широковещательный режим не поддерживается. Помимо полей адреса и функции, сообщение требует, чтобы информационное поле содержало логический адрес первой ячейки и число ячеек, статус которых необходимо получить.

Если возвращаемое количество выходов не кратно восьми, то оставшиеся старшие биты в последнем байте сообщения будут установлены в «0».

Адрес (dec)	**Описание**	**Состояние**
000	Реле включения двигателя K1	1: Включено 0: Отключено
001	Реле универсальное K2	
002	Реле универсальное K3	
003	Сброс аварии	1 (Включено): команда сброса 0 (Отключено): не влияет
004	Сброс (перезагрузка) контроллера	
005-049	Резерв	

Примечание - При чтении выходов по адресам 004 – 005 их значение всегда будет нулевым.

Чтение состояния 5 выходов с адреса 000.

Запрос RTU		**Ответ RTU**	
Название поля	*(Hex)*	*Название поля*	*(Hex)*
Адрес	01	Адрес	01
Функция	01	Функция	01
Начальный адрес старший	00	Счетчик байт	01
Начальный адрес младший	00	Данные выхода 004 - 000	04
Количество выходов старший	00	CRC	50 4B
Количество выходов младший	05		
CRC	FC 09		

Статус выходов 004 - 000 = 0x04h = 00000100.
Читая слева направо, видим, что выход 002 (Реле K3) в состоянии «Включено», остальные в состоянии «Отключено».
Чтение состояние одного выхода по адресу 001 (Реле K2).

Запрос RTU		**Ответ RTU**	
Название поля	*(Hex)*	*Название поля*	*(Hex)*
Адрес	01	Адрес	01
Функция	01	Функция	01
Начальный адрес старший	00	Счетчик байт	01
Начальный адрес младший	00	Данные выхода 001	00
Количество выходов старший	00	CRC	51 88
Количество выходов младший	01		
CRC	FD CA		

В младшем разряде – состояние бита: 0 (реле K2 отключено).
При запросе любого единичного выхода в младшем разряде будет его состояние.

Установка единичного дискретного выхода производится командой **05h** (Write Single Coil).

Запрос содержит содержит адрес устройства, номер функции, адрес выхода (2 байта) и состояние, в которое его необходимо установить (2 байта).

Значение FF 00 (hex) – состояние «**Включено**», значение 00 00 (hex) – состояние «**Отключено**». Любое другое значение неверно и не влияет на выход.

Запись состояния «Включено» дискретного выхода по адресу 003 (Сброс аварии). Нормальный ответ повторяет запрос.

Запрос RTU		**Ответ RTU**	
Название поля	*(Hex)*	*Название поля*	*(Hex)*
Адрес	01	Адрес	01
Функция	05	Функция	05
Начальный адрес старший	00	Начальный адрес старший	00
Начальный адрес младший	03	Начальный адрес младший	03
Значение выхода старший	FF	Значение выхода старший	FF
Значение выхода младший	00	Значение выхода младший	00
CRC	7C 3A	CRC	7C 3A

Примечание - Изменение состояния выхода по адресу 000 «Реле включения двигателя К1» возможно только при режиме работы - «Управление по RS-485». В других режимах попытки записи значений будут игнорироваться.

Изменение состояния выходов по адресам 001…002 «Реле универсальное К2…К3» возможно только при установке функций каждого реле - «Управляется по RS-485».

Сброс аварии и перезагрузка контроллера осуществляется записью состояния «Включено» в соответствующие регистры. Значение этих выходов при чтении всегда нулевое.

3.2 Чтение состояния дискретных входов (Read Discrete Inputs).

Чтение дискретных входов производится командой **02h (Read Discrete Inputs)**.

Данная функция позволяет пользователю получить состояние («Замкнут» или «Разомкнут») дискретных входов и аварийных битовых сигналов адресуемого контроллера. Широковещательный запрос не поддерживается.

Запрос содержит адрес устройства, номер функции, начальный адрес (2 байта) и количество требуемых входов (2 байта). Статус входов в ответном сообщении передается как один выход на бит.

Если возвращаемое количество входов не кратно восьми, то оставшиеся старшие биты в последнем байте сообщения будут установлены в 0. Счетчик байт содержит количество байт, передаваемых в поле данных.

Адрес (dec)	**Описание**	**Состояние**
050	Дискретный вход DI.1	0: вход разомкнут 1: вход замкнут
051	Дискретный вход DI.2	
052	Дискретный вход DI.3	
053	Дискретный вход DI.4	
054	Дискретный вход DI.5	
055	Дискретный вход DI.6	
056	-	
057	-	
058	Сигнал повышения напряжения	0: нет аварийного сигнала 1: есть аварийный сигнал
059	Сигнал понижения напряжения	
060	Сигнал перекоса фаз по напряжению	
061	Сигнал повышения тока	
062	Сигнал понижения тока	
063	Сигнал перекоса фаз по току	
064	Сигнал неисправности аналогового входа	
065	Сигнал о замыкании (утечке) на корпус	

Чтение 3 входов с адреса 053.

Запрос RTU		**Ответ RTU**	
Название поля	*(Hex)*	*Название поля*	*(Hex)*
Адрес	01	Адрес	01
Функция	02	Функция	02
Начальный адрес старший	00	Счетчик байт	01
Начальный адрес младший	35	Данные входов 055 – 053	02
Количество входов старший	00	CRC	20 49
Количество входов младший	03		
CRC	28 05		

Статус входов 055 – 053 = 0x02h = 0000 0010.
Читая с нулевого бита, определяем состояние входов 055 – 053.
53 – «0», 54 – «1», 55 – «0», далее идут нули, независимо от состояния входов.

Чтение 8 входов с адреса 050.

Запрос RTU		**Ответ RTU**	
Название поля	*(Hex)*	*Название поля*	*(Hex)*
Адрес	01	Адрес	01
Функция	02	Функция	02
Начальный адрес старший	00	Счетчик байт	01
Начальный адрес младший	32	Данные входов 057 – 050	10
Количество входов старший	00	CRC Lo	A0 44
Количество входов младший	08		
CRC	D8 03		

Статус входов 057 – 050 = 0x10h = 0001 0000. В младшем бите состояние регистра 050.

Чтение 16 входов с адреса 050.

Запрос RTU		**Ответ RTU**	
Название поля	*(Hex)*	*Название поля*	*(Hex)*
Адрес	01	Адрес	01
Функция	02	Функция	02
Начальный адрес старший	00	Счетчик байт	02
Начальный адрес младший	32	Данные входов 057 – 050	10
Количество входов старший	00	Данные входов 065 – 058	40
Количество входов младший	10	CRC	B5 88
CRC	D8 09		

Статус входов 057 – 050 = 0x10h = 0001 0000. В младшем бите состояние регистра 050.
Статус входов 065– 058 = 0x40h = 0100 0000. В младшем бите состояние регистра 058.

3.3 Регистры задания параметров объекта (Holding Register).

Чтение регистров производится командой **03h (Read Holding Register)**.

Данная функция позволяет получить двоичное содержимое 16-ти разрядных регистров адресуемого контроллера. Адресация позволяет получить за каждый запрос до 125 регистров. Широковещательный режим не допускается.

Адресуемый контроллер посылает в ответе свой адрес, код выполненной функции и информационное поле. Информационное поле содержит 2 байта, описывающих количество возвращаемых байт данных. Длина каждого регистра данных – 2 байта. Первый байт данных в посылке является старшим байтом регистра, второй – младшим.

Адрес (dec)	Название	Диапазон допустимых значений		
		Тип переменной	Диапазон значений	Реальное значение
100	Команды диагностики и калибровки. 7799 – режим диагностики, 9999 – доступ к изменению калибровочных коэффициентов 164 … 170, 5555 – полный сброс данных контроллера, 1010 – установка ручного режима (переключение режима: кнопкой «Ручной/Автомат» в 119), 1020 – установка автомат. режима (переключение режим: кнопкой «Ручной/Автомат» в 119).			
101	Режим работы	0 – автомат по датчикам уровня dL/dH или dP, 1 – автомат по таймеру и датчику уровня dL(dH), 2 – управление по линии связи RS-485.		
102	Функция управления процессом	0 – прямая (налив), 1 – обратная (дренаж).		
103	Тип датчиков уровня (давления)	0 – дискретные датчики уровня (давления) на входах DI.x, 1 – аналоговый датчик давления (уровня) на входе dP.		
104	Тип сигнала аналогового датчика dP	0 – (0...20мА), 1 – (4...20мА).		
105	Нижнее значение диапазона аналогового датчика, усл.единиц	int	10 : 30000	0,10 : 300,00
106	Верхнее значение диапазона аналогового датчика, усл.единиц	int	10 : 30000	0,10 : 300,00
107	Смещение нулевой точки (глубина колодца) аналогового датчика	int	10 : 30000	0,10 : 300,00
108	Максимальная уставка давления (уровня), усл.единиц	int	10 : 30000	0,10 : 300,00
109	Минимальная уставка давления (уровня), усл.единиц		10 : 30000	0,10 : 300,00
110	Фиксированное значение 1 аналогового датчика давления (уровня), мА	int	0 : 2000	0,00 : 20,00
111	Фиксированное значение 2 аналогового датчика давления (уровня), мА	int	0 : 2000	0,00 : 20,00
112	Тип датчика (трансформатора) тока	0 – тип Т03-90А (измеряемый ток до 100А), 1 – ТТ с унифицированным токовым выходом 5А, 2 – Т03-90А в цепи ТТ с униф. выходом 5А(10 витков), 3 – тип Т2000:1 (измеряемый ток до 120А) 4 – Т2000:1 в цепи ТТ с униф. выходом 5А(10 витков)		
113	Диапазон трансформатора тока с унифицированным токовым выходом(5А), А (если выбран в 112)	int	50 : 10000	5,0 : 1000,0
114	Проверка дискретного датчика «сухого» хода (dS)	0 – не проверяется, 1 – проверяется всегда, 2 – проверяется после запуска.		
115	Сигнал «Внешнее управление» (E.Run)	0 – не используется, 1 – разрешение работы по датчикам уровня, 2 – принудительное включение нагрузки.		
116	Функция датчика аварийного уровня dAV (в автоматическом режиме работы)	0 – не используется, 1 – аварийный останов до снятия сигнала, 2 – аварийный останов с выдержкой времени, 3 – отработка сигнала «Перелив».		

117	Функция универсального реле K2	0 – реле не используется, 1 – работа (питание подано), 2 – авария, 3 – двигатель Включен/Отключен, 4 – состояние входа DI.1 (ОТКЛ., если вход разомкнут), 5 – состояние входа DI.2 (ВКЛ., если вход замкнут), 6 – состояние входа DI.3, 7 – состояние входа DI.4, 8 – состояние входа DI.5, 9 – состояние входа DI.6, 10 – ручной / автоматический режим, 11 – достижение фиксированного значения в mA-1, 12 – достижение фиксированного значения в mA-2, 13 – минимальный уровень воды, 14 – максимальный уровень воды, 15 – авариный уровень воды (перелив), 16 – реле управляется сигналом от ПК(ПЛК) по RS-485, 17 – кратковременное включение на 3 секунды при таймауте соединения с ПК.		
118	Функция универсального реле K3			
119	Выбор переключателя режима «Ручной/Автомат»	0 – кнопка на лицевой панели L3, 1 – дискретный вход DI.X с установленной функцией входа: «Auto».		
120	Проверка замыкания (утечки) на корпус	0 – нет, 1 – да		
121	Включение/отключение звуковых сигналов L3	0 – отключить сигналы, 1 – включить сигналы		
122	Номер пункта меню, постоянно отображаемый на индикаторе после запуска двигателя	1 – средний ток двигателя, 2 – ток фазы A, 3 – ток фазы B, 4 – ток фазы C, 5 – напряжение фазы A, 6 – напряжение фазы B, 7 – напряжение фазы C, 8 – полная мощность S, 9 – активная мощность P, 10 – коэффициент мощности cos(φ), 11,12 – значение аналогового датчика dP в мА и усл.ед., 13 – время наработки, 14 – количество пусков двигателя.		
123	Тип питающей сети	0 – однофазная, 1 – трехфазная		
124	Максимальный ток (перегрузка), A	int	5 : 10000	0,5 : 1000,0
125	Минимальный ток (недогрузка), A		0 : 10000	0,0 : 1000,0
126	Значение перекоса фаз по току в %	char	0 : 40 (0 – отключает проверку)	
127	Максимальное напряжение, В	int	2300 : 2700	230,0 : 270,0
128	Минимальное напряжение, В		1600 : 2200	160,0 : 220,0
129	Значение перекоса фаз по напряжению, В		0 : 500	0,0 : 50,0
130	Значение минимального коэффициента мощности cos(φ) (0 – отключает эту защиту)	char	0 : 100	0,0 : 1,00
131	Разрешаемое количество пусков в час	0 : 100 (0 – отключает проверку)		
132	Количество повторяющихся подряд аварий № 02…№ 09 в час для срабатывания блокировки запуска	0 : 30 (0 – отключение защиты)		
133	Время работы насоса по таймеру, минут	char	1 : 180	

134	Задержка пуска после подачи питания, секунд			0 : 180	
135	Время блокировки пускового тока, секунд			1 : 90	
136	Время срабатывания аварии, секунд			1 : 30	
137	Время выдержки после аварии, минут			1 : 60	
138	Время срабатывания датчика «сухого» хода dS, секунд			1 : 120	
139	Время выдержки после «сухого» хода, минут			1 : 60	
140	Время выдержки после внешней аварии (E.Error), минут			0 : 60 (0 – без выдержки времени)	
141	Таймер ограничения времени работы, минут			0 : 240	
142	Таймер задержки включения, секунд			0 : 600	
143	Таймер задержки отключения, секунд			0 : 600	
144	Подключаемый сигнал входа DI.1		0 – вход не используется, 1 – датчик верхнего уровня (dH), 2 – датчик нижнего уровня (dL), 3 – датчик «сухого» хода (dS), 4 – датчик аварийного уровня(перелив) (dAV), 5 – сигнал принудительного сброса аварии, 6 – сигнал внешнего управления (E.Run), 7 – сигнал внешней аварии (E.Error), 8 – переключатель «Ручной/Автомат» (Auto), 9 – сигнал «Пуск/Стоп» (в ручном режиме), 10 – сигнал «Пуск» (в ручном режиме), 11 – сигнал «Стоп» (в ручном режиме).		
145	Подключаемый сигнал входа DI.2				
146	Подключаемый сигнал входа DI.3				
147	Подключаемый сигнал входа DI.4				
148	Подключаемый сигнал входа DI.5				
149	Подключаемый сигнал входа DI.6				
150	Тип контакта входа DI.1		0 – н.з. 1 – н.о.		
151	Тип контакта входа DI.2				
152	Тип контакта входа DI.3				
153	Тип контакта входа DI.4				
154	Тип контакта входа DI.5				
155	Тип контакта входа DI.6				
156	Связь	Адрес устройства в сети Modbus	char	1 … 247	
157		Скорость передачи, бит/сек.	0 – 2400, 1 – 4800, 2 – 9600,	3 – 14400, 4 – 19200, 5 – 38400,	6 – 56000, 7 – 57600, 8 – 115200
158		Количество бит данных	7 … 8		
159		Четность	0 – нет (none), 1 – нечетный (odd), 2 – четный (even)		
160		Стоп-бит	1 … 2		
161		Максимальный сетевой таймаут (соединение по Modbus), секунд	int	0 : 600	
162		Действие при таймауте соединения по Modbus	0 – ничего не предпринимать, 1 – авария по таймауту.		

163	Сброс данных контроллера	10 – разрешение отображения калибровочных коэфф., 20 – сброс моточасов, 25 – сброс количества запусков, 50 – установка заводских значений (без калибровочных коэффициентов 164 … 170), 77 – полный сброс всех данных.		
164	Калибровочный коэффициент Ia	int	700 : 1200	0.700 : 1.200
165	Калибровочный коэффициент Ib			
166	Калибровочный коэффициент Ic			
167	Калибровочный коэффициент Ua			
168	Калибровочный коэффициент Ub			
169	Калибровочный коэффициент Uc			
170	Калибровочный коэффициент dP			

Запись в регистры производится командой **06h (Write Single Register)**

В случае успешного выполнения функции ответное сообщение идентично запросу.
При попытке записи значений вне допустимого диапазона, буден записано минимальное или максимальное значение этого диапазона.

Чтение 3 регистров с адреса 101.

Запрос RTU		**Ответ RTU**	
Название поля	*(Hex)*	*Название поля*	*(Hex)*
Адрес	01	Адрес	01
Функция	03	Функция	03
Начальный адрес старший	00	Счетчик байт	06
Начальный адрес младший	65	Данные регистра 101 старший	00
Количество регистров старший	00	Данные регистра 101 младший	00
Количество регистров младший	03	Данные регистра 102 старший	00
CRC	15 D4	Данные регистра 102 младший	00
		Данные регистра 103 старший	01
		Данные регистра 103 младший	90
		CRC	20 89

Запись регистра по адресу 108.
Нормальный ответ контроллера повторяет запрос.

Запрос RTU		**Ответ RTU**	
Название поля	*(Hex)*	*Название поля*	*(Hex)*
Адрес	01	Адрес	01
Функция	06	Функция	06
Начальный адрес старший	00	Начальный адрес старший	00
Начальный адрес младший	6C	Начальный адрес младший	6C
Данные регистра 108 старший	00	Данные регистра 108 старший	00
Данные регистра 108 младший	0A	Данные регистра 108 младший	0A
CRC	C9 D0	CRC	C9 D0

Запись нескольких регистров настройки производится командой **10h (Write Multiple Registers)**

Данная функция позволяет записать до 64 16-ти разрядных регистров настройки. В случае успешного выполнения функции ответное сообщение состоит из первых 6 байт запроса.
При попытке записи значений вне допустимого диапазона, буден записано минимальное или максимальное значение этого диапазона.

Запись 5 регистров с адреса 101(значения 0, 0, 400, 300, 10).

Запрос RTU		**Ответ RTU**	
Название поля	*(Hex)*	*Название поля*	*(Hex)*
Адрес	01	Адрес	01
Функция	10	Функция	10
Начальный адрес старший	00	Начальный адрес старший	00
Начальный адрес младший	65	Начальный адрес младший	65
Количество регистров старший	00	Количество регистров старший	00
Количество регистров младший	05	Количество регистров младший	05
Количество байт в регистрах	0A	CRC	10 15
Данные регистра 101 старший	00		
Данные регистра 101 младший	00		
Данные регистра 102 старший	00		
Данные регистра 102 младший	00		
Данные регистра 103 старший	01		
Данные регистра 103 младший	90		
Данные регистра 104 старший	01		
Данные регистра 104 младший	2C		
Данные регистра 105 старший	00		
Данные регистра 105 младший	0A		
CRC	E5 63		

3.4 Регистры чтения данных объекта (Input Register).

Чтение регистров производится командой **04 (Read Input Register)**.

Данная функция позволяет получить двоичное содержимое 16-ти разрядных регистров адресуемого контроллера. Адресация позволяет получить за каждый запрос до 125 регистров. Широковещательный режим не допускается. Адресуемый контроллер посылает в ответе свой адрес, код выполненной функции и информационное поле. Информационное поле содержит 2 байта, описывающих количество возвращаемых байт данных. Длина каждого регистра данных – 2 байта. Первый байт данных в посылке является старшим байтом регистра, второй – младшим.

Адрес	**Название**	**Описание**	
300	Версия ПО	100…1000 (100 = v.1.00)	
301	Дата ПО	День	0…31
302		Месяц	0…12
303		Год	2023…2100

<table>
<tr><td rowspan="6">304</td><td rowspan="6">Управление
<u>(чтение и запись)</u></td><td>бит</td><td colspan="2"></td></tr>
<tr><td>0</td><td>Реле включения двигателя K1</td><td>1: Вкл, 0: Откл</td></tr>
<tr><td>1</td><td>Реле универсальное K2</td><td>1: Включено</td></tr>
<tr><td>2</td><td>Реле универсальное K3</td><td>1: Включено</td></tr>
<tr><td>3</td><td>Сброс аварии</td><td>1: команда сброса</td></tr>
<tr><td>4</td><td>Сброс (перезагрузка) контроллера</td><td>1: команда сброса</td></tr>
<tr><td colspan="5">Примечание – Данный регистр дублирует функции управления, описанные в разделе 3.1 «Чтение и запись состояния дискретных выходов». Управлять нагрузкой возможно как чтением и записью битовых регистров по адресу 000 – 004 (функция 01 и 05), так и чтением и записью одного регистра по адресу 304.</td></tr>
<tr><td>305</td><td>Режим работы L3</td><td colspan="3">0: Ручное (местное) управление
1: Автоматический режим</td></tr>
<tr><td>306</td><td>Состояние L3</td><td colspan="3">0: Работа L3 приостановлена
1: Проверка параметров перед запуском двигателя
2: Запуск двигателя (на Время блокировки пускового тока, рег.135)
3: Ожидание срабатывания сигнала «Внешнее управление» E.Run
4: Ожидание нажатия кнопки «ПУСК» в ручном режиме
5: Ожидание нажатия кнопки «СТОП» в ручном режиме
6: Ожидание верхнего уровня
7: Ожидание нижнего уровня
8: Ожидание срабатывания датчика dL (Режим – По таймеру)
9: Ожидание срабатывания датчика dH (Режим – По таймеру)
10: Двигатель включен на время работы по таймеру (время в рег.133)
11: Задержка пуска на заданное время (время задержки в рег.142)
12: Задержка останова на заданное время (время задержки в рег.143)
13: -
14: Задержка пуска после подачи питания (время задержки в рег.134)
15: Ожидание команды включения (по линии связи)
16: Ожидание команды отключения (по линии связи)
17: -
18: -
19: Самодиагностика L3 после подачи питания
20: Ожидание срабатывания датчика dS (сухой ход)
21: Ожидание снятия сигнала «Внешняя авария» (E.Error)
22: Ожидание снятия сигнала с датчика аварийного уровня (dAV)
23: Запущен режим диагностики по RS-485
24: -
25: Принудительный запуск двигателя по сигналу "Внешнее управление" E.Run (значение рег.115 равно 2).</td></tr>
<tr><td rowspan="17">307</td><td rowspan="17">Состояние дискретных входов и аварийных сигналов</td><td>бит</td><td colspan="2"></td></tr>
<tr><td>0</td><td>Дискретный вход DI.1</td><td rowspan="8">0: разомкнут
1: замкнут</td></tr>
<tr><td>1</td><td>Дискретный вход DI.2</td></tr>
<tr><td>2</td><td>Дискретный вход DI.3</td></tr>
<tr><td>3</td><td>Дискретный вход DI.4</td></tr>
<tr><td>4</td><td>Дискретный вход DI.5</td></tr>
<tr><td>5</td><td>Дискретный вход DI.6</td></tr>
<tr><td>6</td><td>-</td></tr>
<tr><td>7</td><td>-</td></tr>
<tr><td>8</td><td>Сигнал повышения напряжения</td><td rowspan="8">0: нет аварийного сигнала

1: есть аварийный сигнал</td></tr>
<tr><td>9</td><td>Сигнал понижения напряжения</td></tr>
<tr><td>10</td><td>Сигнал перекоса фаз по напряжению</td></tr>
<tr><td>11</td><td>Сигнал повышения тока</td></tr>
<tr><td>12</td><td>Сигнал понижения тока</td></tr>
<tr><td>13</td><td>Сигнал перекоса фаз по току</td></tr>
<tr><td>14</td><td>Сигнал неисправности аналогового входа</td></tr>
<tr><td>15</td><td>Замыкание (утечка) на корпус</td></tr>
<tr><td colspan="5">Примечание – Данный регистр дублирует функцию чтения дискретных входов, описанную в разделе 3.2 «Чтение состояния дискретных входов». Возможно как чтение одиночных битовых регистров по адресам 50 – 65 (функция 02), так и чтение одного регистра по адресу 307 (функция 04).</td></tr>
</table>

308	Код аварии	0: Нет аварии 1: Неправильное чередование фаз или напряжение одной из фаз меньше 50В 2: Повышение напряжения выше заданной уставки 3: Понижение напряжения ниже заданной уставки 4: Перекос фаз по напряжению 5: Повышение тока выше заданной уставки 6: Понижение тока ниже заданной уставки 7: Перекос фаз по току 8: «Сухой» ход (срабатывание входа dS) 9: Замыкание (утечка) на корпус обмоток электродвигателя 10: Неправильное срабатывание дискретных датчиков уровня (давления) 11: Превышено время работы двигателя (задается в рег.141) 12: Внешняя авария (срабатывание входа E.Error) 13: Внутренняя авария L3 (нет связи с измерительным модулем) 14: Превышение количества пусков в час (рег.131) в автоматическом режиме 15: Отказ аналогового датчика давления (уровня) 0…20(4…20)мА 16: Таймаут соединения с ПК (ПЛК) по RS-485 (задается в рег.161, 162) 17: Понижение коэффициента мощности cos(φ) ниже заданной уставки 18: Не заданы дискретные входы DI.x для датчиков уровня (dL и/или dH) 19: Срабатывание датчика аварийного уровня dAV (перелив) 20: - 21: - 22: … 29: Превышено количество аварий № 02...№ 09 в час (задается в рег.132)			
309	Оставшееся время выдержки при аварии в секундах		int	0 : 3600	
310	Напряжение фазы A(L1), В		int	0 : 3000	0,0 : 300,0
311	Напряжение фазы B(L2), В				
312	Напряжение фазы C(L3), В				
313	Среднее напряжение по фазам, В				
314	Перекос по напряжению, В			0 : 1000	0,0 : 100,0
315	Ток фазы A(L1), А			0 : 10000	0,0 : 1000,0
316	Ток фазы B(L2), А				
317	Ток фазы C(L3), А				
318	Средний ток по фазам, А				
319	Перекос по току, %			0 : 1000	0,0 : 100,0
320	Коэффициент мощности двигателя cos(φ)	Фаза A(L1)	char	0 : 100	0,0 : 1,00
321		Фаза B(L2)			
322		Фаза C(L3)			
323		Среднее значение			
324	Полная мощность S, кВА		int	0 : 65000	0,0 : 650,00
325	Активная мощность P, кВт				
326	Состояние аналогового датчика dP 0…20mA (4…20mA)		0 – исправен, 1 – обрыв, 2 – короткое замыкание.		
327	Значение аналогового датчика dP в mA		int	0 : 2500	0,00 : 25,00
328	Значение аналогового датчика dP с учетом заданной глубины колодца (рег.107), усл. единиц			0 : 30000	0,00 : 300,00
329	Время работы по таймеру (сколько прошло) в секундах		int	0 : 20000	
330	Время работы по таймеру (сколько осталось) в секундах				
331	Время наработки двигателя (сбрасываемое), часов		int	0 : 9999	
332	Количество пусков двигателя (сбрасываемое)				

Чтение 3 регистров с адреса 300 (0x012Ch) (полученные значения 2, 0, 4).

Запрос RTU		**Ответ RTU**	
Название поля	*(Hex)*	*Название поля*	*(Hex)*
Адрес	01	Адрес	01
Функция	04	Функция	04
Начальный адрес старший	01	Счетчик байт	06
Начальный адрес младший	2C	Данные регистра 300 старший	00
Количество регистров старший	00	Данные регистра 300 младший	02
Количество регистров младший	03	Данные регистра 301 старший	00
CRC	70 3E	Данные регистра 301 младший	00
		Данные регистра 302 старший	00
		Данные регистра 302 младший	04
		CRC	18 90

4. Исключительные ситуации

L4 поддерживает сообщения информирования клиента (мастера) Modbus об исключительных ситуациях (Exception). Формат возвращаемых пакетов полностью соответствует документу «Modbus Application Protocol Specification v1.1a». Сообщения об исключительных ситуациях возникают только на запросы, адресованные данному устройству с правильным значением LRC или CRC пакета.

Код ошибки	**Название**	**Описание**
01	Неподдерживаемая команда	Возникает только при запросе с номером команды, которую не поддерживает данное устройство.
02	Неподдерживаемый адрес данных	Возникает только при запросе с адресом данных, которых нет в таблицах соответствия между адресами Modbus и внутренней памятью устройства
03	Неверное количество данных	В запросе содержится значения недопустимые для сервера. Например, запрос количества регистров более чем 120.

Когда контроллер обнаруживает одну их этих ошибок, он посылает ответное сообщение, содержащее адрес, код функции, код ошибки и контрольную сумму. Для указания на то, что ответное сообщение – это уведомление об ошибке, старший бит поля кода функции устанавливается в 1.

Запрос RTU

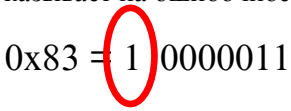
Адрес	Функция	Старший байт адреса	Младший байт адреса	Старший байт числа ячеек	Младший байт числа ячеек	CRC
01	03	00	12	00	06	65 CD

Этот запрос требует от устройства с адресом 01 данных 6 регистров с адреса 18. Но, например, это устройство имеет максимальный адрес 0x0016, а запрашиваемое количество данных превышает диапазон адреса и является ошибочным. Соответственно, будет сгенерировано следующее ответное сообщение:

Ответ RTU

Адрес	Функция	Код исключительной ситуации	CRC
01	83	03	XX XX

Значение в поле функции равно оригинальному значению с установленным в единицу старшим битом. Код исключительной ситуации 03 указывает на ошибочное количество данных.



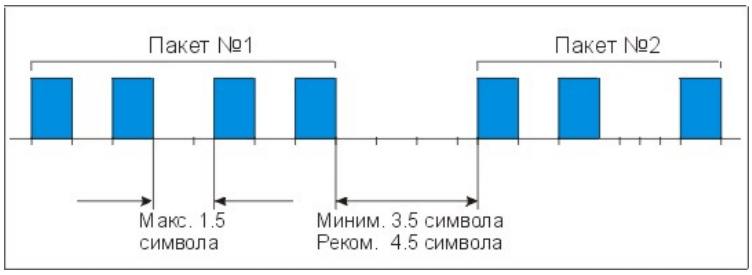
5. Задержки между пакетами

Временные задержки между пакетами и символами пакетов полностью соответствуют «Modbus over Serial Line Specification & Implementation guideV1.0».

В ASCII-режиме сообщение начинается с символа "двоеточие" (“:”ASCII, 3A hex), и заканчивается символами "Возврат каретки/Перевод строки" (CR/LF, ASCII 0D и 0A hex). Опрашиваемые устройства в сети непрерывно отслеживают символ "двоеточие". После него начинается прием тела сообщения до символов CR/LF. Интервалы между символами могут быть до 1 секунды. Если интервал больше, принимающее устройство считает это ошибкой и игнорирует сообщение.

В RTU-режиме сообщение начинается с интервала тишины, равного времени передачи 3.5 символов при данной скорости передачи в сети. Затем первым символом передается адрес устройства и остальные байты пакета. Между символами одного пакета может быть задержка длинной не более полутора символов. Между пакетами должна быть задержка не менее 3,5 символов. Рекомендуется начать передавать следующий пакет не ранее чем через 4,5 символа после получения последнего бита предыдущего пакета. Если в интервале между 1,5 символами и 3,5 символами после прихода последнего символа, приходит первый символ следующего пакета, сбрасываются оба пакета.

Комментарий: задержка длинной в символ - это время необходимое, для того чтобы передать 8 бит данных при данной скорости передачи и параметрах соединения.



Рекомендуемые задержки между пакетами для режима RTU

Скорость интерфейса при параметрах 8N1	Минимальное время между символами в пакете	Минимальная задержка между пакетами	Рекомендуемая задержка между пакетами
2400 бит/с	6,3 мс	14,6 мс	18,8 мс
4800 бит/с	3,2 мс	7,3 мс	9,4 мс
9600 бит/с	1,6 мс	3,6 мс	4,7 мс
14400 бит/с	1,0 мс	2,4 мс	3,1 мс
19200 бит/с	0,8 мс	1,8 мс	2,3 мс
38400 бит/с	0,4 мс	0,9 мс	1,2 мс
57600 бит/с	0,3 мс	0,6 мс	0,8 мс
115200 бит/с	0,2 мс	0,3 мс	0,4 мс

6 Генерация контрольной суммы

6.1 Контрольная сумма CRC

CRC - это 16-ти разрядная величина, т.е. два байта. CRC вычисляется передающим устройством и добавляется к RTU-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.

Используются два метода вычисления CRC: цикличный или табличный.

При цикличном методе 16-ти битовый регистр CRC предварительно загружается числом FFFF hex. 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. Результата 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 FFFFh (все 1), 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 нулем.
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.

При табличном вычислении 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. Индексация CRC в этом случая обеспечивает быстрое выполнение вычислений новой величины CRC для каждого нового байта из буфера сообщения.

Размещение CRC в сообщении

При передаче 16 бит контрольной суммы CRC в сообщении, сначала передается младший байт, затем старший. Например, если CRC равна 1288 hex:

Адрес	Функция	Счетчик байт	Данные	Данные	Данные	Данные	CRC Lo	CRC Hi
							88	**12**

Пример функции на языке С, реализующей генерацию CRC цикличным методом

```
void main (void)
{
 unsigned char CRC_Hi, CRC_Lo;   // Старший и младший байты контрольной суммы
 unsigned char Array[128];              // Массив данных, содержащий ответ
       …
       // В массиве Array[] содержатся 8 байт ответного сообщения
       Create_CRC(8);
       // Добавляем к массиву Array[] 9 и 10 байты контрольной суммы (CRC_Lo и CRC_Hi)
       Array[9]= CRC_Lo;
       Array[10]= CRC_Hi;
       // Отправляем ответ
       …
}// End
void Create_CRC(unsigned char NBytes)// Подсчет CRC по количеству байт массива Array[]
{
 unsigned int Register = 0xFFFF;
 unsigned int Lsb;
 unsigned char i, j;

    for (i=0; i<NBytes; i++)
    {
        Register^=Array[i];
        for ( j=0; j<8; j++)
        {
            Lsb = Register & 0x0001;
            Register = Register >> 1;  //  Сдвигаем на 1 вправо
            if (Lsb) Register^=0xA001; // Если мл. бит 0 - исключающее ИЛИ с числом 0xA001
        }
    }
    CRC_Hi = Register >> 8;
    CRC_Lo = Register;

}// End
```

Пример функции на языке С, реализующей генерацию CRC табличным методом

```
void main (void)
{
 unsigned char CRC_Hi, CRC_Lo;   // Старший и младший байты контрольной суммы
 unsigned char Array[128];              // Массив данных, содержащий ответ
       …
       // В массиве Array[] содержатся 8 байт ответного сообщения
       Create_CRC_TAB(Array, 8);
       // Добавляем к массиву Array[] 9 и 10 байты контрольной суммы (CRC_Lo и CRC_Hi)
       Array[9]= CRC_Lo;
       Array[10]= CRC_Hi;
       // Отправляем ответ
       …
}// End

unsigned short Create_CRC_TAB(unsigned char *Buffer, unsigned char NBytes)
{
  unsigned char uchCRCHi = 0xFF, uchCRCLo = 0xFF;
  unsigned int uIndex;

       while( NBytes--)
       {
              uIndex = uchCRCHi ^ *Buffer++;
              uchCRCHi = uchCRCLo ^ auchCRCHi[uIndex];
              uchCRCLo = auchCRCLo[uIndex];
       }
       CRC_Hi= uchCRCHi;
       CRC_Lo= uchCRCLo;

}// End
```

```
const unsigned 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 unsigned 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
    };
```