СЕТЕВОЙ ПРОТОКОЛ MODBUS

СОДЕРЖАНИЕ

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ВВЕДЕНИЕ

ModBus - открытый коммуникационный протокол, который был разработан компанией Modicon (сейчас принадлежит Schneider Electric). Впервые спецификация протокола была опубликована в 1979 году. Это был открытый стандарт, описывающий формат сообщений и способы их передачи в сети, состоящей из различных электронных устройств.

По способу и формату передачи данных протокол разделяется на:

- ModBus ASCII
- ModBus RTU
- ModBus TCP

ModBus ASCII (используется редко)

- Вариант передачи данных для последовательного интерфейса:
  - UART, RS-232, RS-422, RS-485
- При обмене используются только ASCII-символы
- Начало сообщения помечается символом ***:***
- Конец сообщения помечается символами ***CR***/***LF*** (перевод каретки и новая строка)
- Проверка целостности пакета: Однобайтовая контрольная сумма (CRC)

ModBus RTU

- Вариант передачи данных для последовательного интерфейса:
  - UART, RS-232, RS-422, RS-485
- При обмене используется компактный двоичный цифровой сигнал
- Сообщения разделяются паузой на линии
- Проверка целостности пакета: Двубайтовая контрольная сумма (CRC)

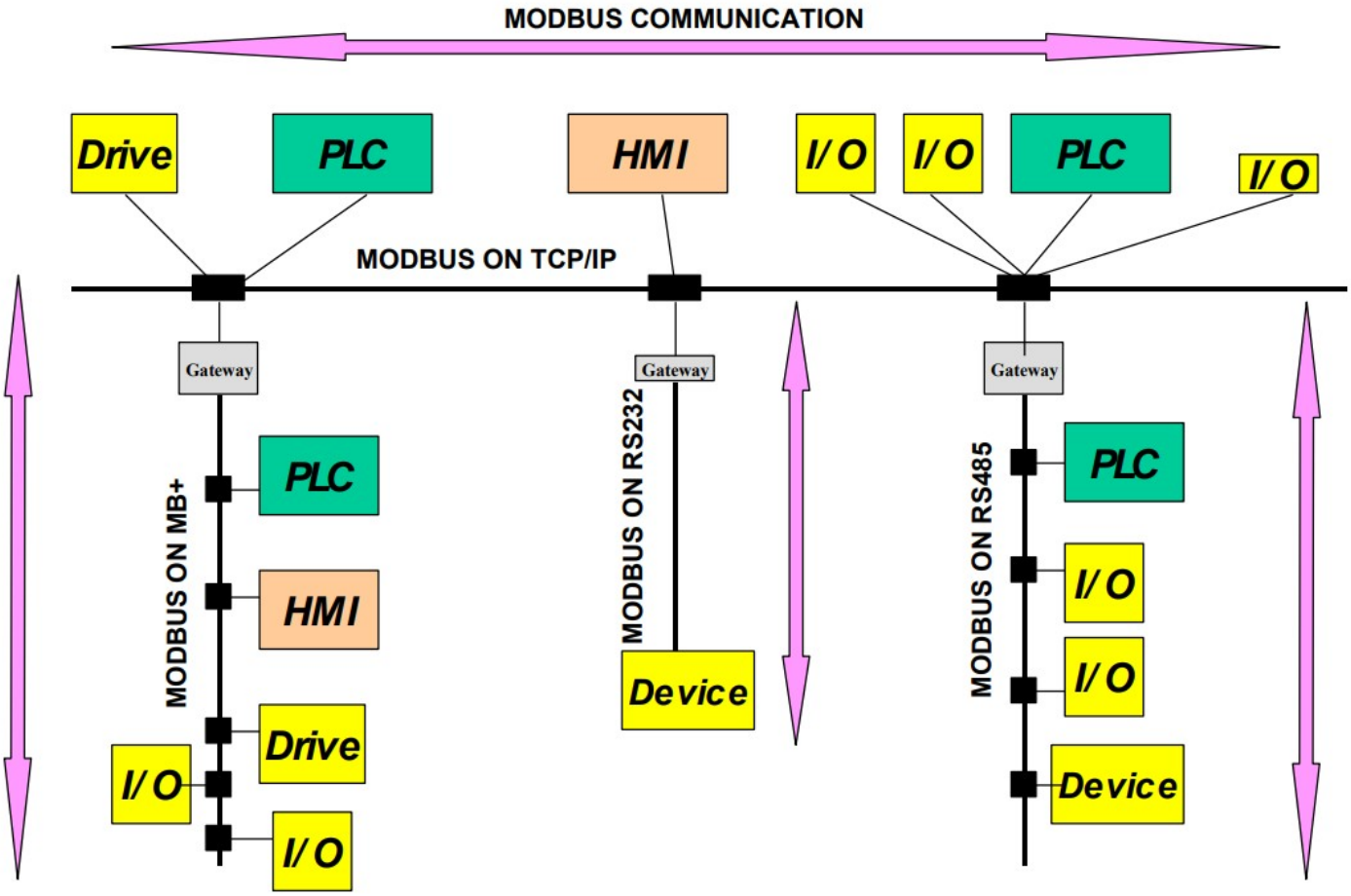
ModBus TCP

- Вариант передачи данных для Ethernet
  - TCP/IP
- При обмене используется пакет, по структуре аналогичный ModBus RTU
- Контроль целостности обеспечивается средствами TCP/IP (поэтому нет CRC)

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ТОПОЛОГИЯ СЕТИ

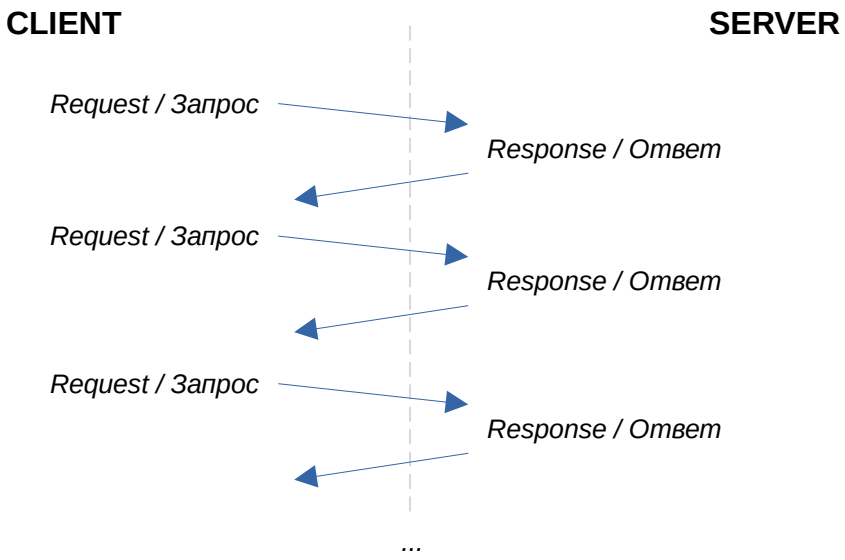
- Протокол поддерживает работу во всех сетевых топологиях
- Связь может осуществляться по последовательным линиям и по сетям Ethernet
- Для связи между сегментами с различной топологией и/или интерфейсами используются специальные Преобразователи (конвертеры) интерфейсов, поддерживающие протокол ModBus (например, преобразователь RS-485 / Ethernet - Moxa Nport 5130)



СЕТЕВОЙ ПРОТОКОЛ MODBUS

МОДЕЛЬ ОБЩЕНИЯ

- Один Ведущий / Master / Client / ***Клиент***
- Несколько Ведомых / Slave / Server / ***Сервер***
- Транзакция «***Запрос-Ответ***» / «Request-Response»
  - инициирует / шлет запросы - Клиент (режим опроса / pooling mode)
  - Отвечает - Сервер (режим ожидания / waiting mode)



СЕТЕВОЙ ПРОТОКОЛ MODBUS

ТИПЫ СООБЩЕНИЙ

- Минимальная единица информации — 1 байт
- Типы транзакций определены:

ЗАПРОС / **REQUEST**
ОТВЕТ ПОЛОЖИТЕЛЬНЫЙ / RESPONSE
ОТВЕТ ОТРИЦАТЕЛЬНЫЙ С КОДОМ ОШИБКИ / ERROR CODE

Request / Response Transaction (ModBus TCP)

ID транзакции	ID протокола	Длина пакета	SERVER ADDRESS	FUNCTION CODE	DATA
2 байта	*2 байта*	*2 байта*	*1 байт*	*1 байт*	*до 252 байт*

Request / Response Transaction (ModBus RTU)

SERVER ADDRESS	FUNCTION CODE	DATA	CRC
1 байт	*1 байт*	*до 252 байт*	*2 байта*

256 байт максимум

<u>ПОЛЕ: SERVER ADDRESS / АДРЕС ВЕДОМОГО</u>

- Размер: 1 байт
- Обязательное для всех транзакций
- Значение:
  - целое беззнаковое число, означающее:
    адрес Ведомого (кому адресован запрос)

Запросы принимает Ведомый с указанным адресов.
Устройство с отличным адресом просто игнорирует пришедший запрос (молчит).
Ведомый (адресат) в ответе должен вернуть свой адрес.

Диапазоны значений адресов определены:

0	Код широковещательного запроса всем устройствам
1 … 247	Адрес конкретного устройства на линии ***(уникальное значение для каждого устройства)***
248 … 255	Резерв

СЕТЕВОЙ ПРОТОКОЛ MODBUS

КОДЫ ФУНКЦИЙ

ПОЛЕ: FUNCTION CODE / КОД ФУНКЦИИ

- Размер: 1 байт
- Обязательное для транзакций:
  - Запрос / Request
  - Ответ положительный / Response
- Значение:
  - целое беззнаковое число, означающее:
    код функции, которую должен выполнить Ведомый
    - прислать данные определенного рода
    - выполнить определенное действие

    Ведомый в ответе должен вернуть код обработанной функции или код ошибки.

    Коды функций определены:

01	Чтение одного или нескольких значений битовых флагов Read ***COIL***
02	Чтение одного или нескольких значений битовых входов Read ***INPUT***
03	Чтение одного или нескольких значений числовых регистров данных Read ***HOLDING REGISTERS***
04	Чтение одного или нескольких значений числовых регистров ввода Read ***INPUT REGISTERS***
05	Запись одного значения битового флага Write Single ***COIL***
06	Запись одного значения числового регистра данных Write Single ***HOLDING REGISTER***
15	Запись одного или нескольких значений битовых флагов Write Multiple ***COIL***
16	Запись одного или нескольких значений числовых регистров данных Write Multiple ***HOLDING REGISTERS***
17	Запрос Адреса Ведомого Report ***SERVER ADDRESS***
07	Чтение кода последней исключительной ситуации Read ***EXCEPTION STATUS***
08	Диагностика ***DIAGNOSTICS*** *(в поле DATA отправляется дополнительный код — SUB-FUNCTION CODE)*

СЕТЕВОЙ ПРОТОКОЛ MODBUS

КОДЫ ФУНКЦИЙ ДИАГНОСТИКИ

Коды функций диагностики (***DIAGNOSTIC SUB-FUNCTION CODE***) определены:

00	«Зеркальный ответ» Echo / Loopback *(возврат данных, полученных в запросе)*
01	Перезапустить сетевой интерфейс Restart Communications Option
02	Вернуть значение диагностического регистра Return Diagnostic Register
10	Сбросить все счетчики и значение дигностического регистра Clear Counters and Diagnostic Register
11	Вернуть значение счетчика успешных ответов Return Bus Message Count
12	Вернуть значение счетчика ошибок Return Bus Communication Error Count
13	Вернуть значение счетчика исключительных ситуаций Return Bus Exception Error Count
14	Вернуть значение счетчика широковещательных сообщений Return Server (broadcast) Message Count
15	Вернуть значение счетчика проигнорированных запросов (когда запрос был адресован другому устройству) Return Server No Response Count
17	Вернуть значение счетчика проигнорированных запросов (когда запрос был адресован этому устройству, но устройство не ответило, т. к. было занято выполнением более приоритетной задачи) Return Server No Response Count
18	Вернуть значение счетчика переполнений буфера данных (когда запрос был адресован этому устройству, но устройство не ответило, т. к. был переполнен буфер данных) Return Bus Character Overrun Count
20	Сбросить значение счетчика переполнений буфера данных Clear Overrun Counter and Flag

СЕТЕВОЙ ПРОТОКОЛ MODBUS

КОДЫ ОШИБОК

ПОЛЕ: ERROR CODE / КОД ОШИБКИ

- Размер: 1 байт
- Заменяет поле FUNCTION CODE в транзакции типа «Ответ отрицательный»
- Значение:
  - целое беззнаковое число, обозначающее:

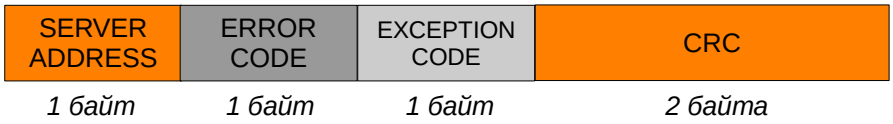
ERROR CODE = 128 + FUNCTION CODE

где, 128_{10} ($0x80_{16}$) — добавочный префикс
FUNCTION CODE — код функции из транзакции запроса

Наличие добавочного кода к коду функции в Ответе означает, что Ведомый не смог выполнить запрошенную функцию.

Причина, по которой Ведомый не смог выполнить функцию, указана в коде Исключительной ситуации / Exception code, который передается в поле DATA.

Response Error Transaction



Коды исключительных ситуаций (***EXCEPTION CODE***) определены:

01	Код функции не может быть обработан (или не поддерживается) ILLEGAL FUNCTION
02	Адрес регистра данных недоступен ILLEGAL DATA ADDRESS
03	Значение для регистра данных является недопустимой величиной (например, недопустимый размер — количество переданных байт) ILLEGAL DATA VALUE
04	Ошибка при обработке кода функции SERVER DEVICE FAILURE
05	Ведомый получил запрос и обрабатывает код функции (на выполнение операции требуется больше времени, чем положено) ACKNOWLEDGE
06	Ведомое устройство занято и не может обработать код функции (необходимо повторить запрос через некоторое время) SERVER DEVICE BUSY
07	Ведомое устройство занято и не может обработать код функции (необходимо повторить запрос через некоторое время) SERVER DEVICE BUSY

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ДАННЫЕ

ПОЛЕ: DATA / ДАННЫЕ

- Размер: до 252 байт
- Обязательное для всех транзакций
- Значение зависит от типа транзакции:
  - для запроса:
    - адрес регистра данных
    - количество регистров данных
    - последовательность (массив) бинарных или числовых значений (данных)
    - код диагностики
  - для ответа:
    - последовательность (массив) бинарных или числовых значений
    - код исключительной ситуации

Бинарные значения

- В памяти Клиента и Сервера одно значение занимает 1 байт
- Диапазон одного означения ограничен
- По сети передаются в упакованном виде:
  - значение (0, 1) бита раскладываются по разрядам группы чисел типа БАЙТ / BYTE
    + 1 BYTE = 8 бит = 1 байт (беззнаковое целое число)
    + группа может состоять как из нескольких БАЙТ, так и из одного
    + по сети БАЙТЫ передаются последовательно друг за другом
    + количество передаваемых БАЙТ указывается в поле DATA
  - алгоритмы упаковки:
  - + логические операции: И, ИЛИ, НЕ и поразрядный сдвиг влево
    + битовые поля / bit-fields
- Отправитель перед отправкой упаковывает биты
- Получатель при получении распаковывает биты
  - извлекает значение бита из соответствующего разряда полученного БАЙТА
  - сохраняет значение бита в памяти в виде 1-байтового числа типа BOOL
    + 1 BOOL = 8 бит = 1 байт (беззнаковое целое число, значения 0 или 1)
- Сервер может хранить битовые данные в упакованном виде

Алгоритм упаковки будет рассмотрен далее — в разделе «Модель данных».

Тип данных	Размер	Значения
BOOL BYTE UINT	1 байт *(8 бит)*	0, 1 FALSE, TRUE

BYTE, UINT — синонимы BOOL *FALSE, TRUE — синонимы 0, 1*

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ДАННЫЕ

Числовые значения

- В памяти Клиента и Сервера одно значение занимает от 1 до нескольких байт
- По сети передаются в упакованном виде:
  - значение числа раскладывается на группу 16-битных чисел типа СЛОВО / WORD
    + 1 WORD = 16 бит = 2 байта (беззнаковое целое число)
    + группа может состоять как из нескольких СЛОВ, так и из одного
    + по сети СЛОВА передаются побайтово:
      - сначала старший байт СЛОВА (Hi)
      - далее младший байт СЛОВА (Lo)
    + количество передаваемых СЛОВ указывается в поле DATA
  - алгоритмы упаковки:
    + объединения / union
- Отправитель перед отправкой упаковывает числа
- Получатель при получении распаковывает числа
  - компонует значение числа из соответствующей группы СЛОВ
  - сохраняет значение в памяти в виде числа нужного типа
- Сервер может хранить числовые данные в упакованном виде

Алгоритм упаковки будет рассмотрен далее — в разделе «Модель данных».

Тип данных	Размер	Значения
WORD	2 байта *(16 бит)*	0 … 65535

ПОЛЕ: CRC / КОНТРОЛЬНАЯ СУММА

- Размер: 2 байта
- Обязательное для всех транзакций
- Значение:
  - циклический код CRC-16 (Cyclic Redundancy Check)

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ТАБЛИЦЫ ДАННЫХ

Типичные функции протокола:

- чтение и запись данных в регистры памяти какого-то устройства

- Спецификация протокола определяет для Сервера до четырех таблиц данных:
  - таблица — массив значений в определенной области памяти
  - каждая таблица:
    - может содержать до 65536 элементов (регистров)
    - определяет базовый тип значений регистров (битовые или числовые)
    - определяет права доступа к таблице в целом
  - каждый элемент массива:
    - имеет уникальный адрес, соответствующий положению (индексу) в таблице (соответственно, адрес от 0 до 65536 максимум)
    - адрес регистра указывается в запросе — в поле DATA

Определены следующие таблицы:

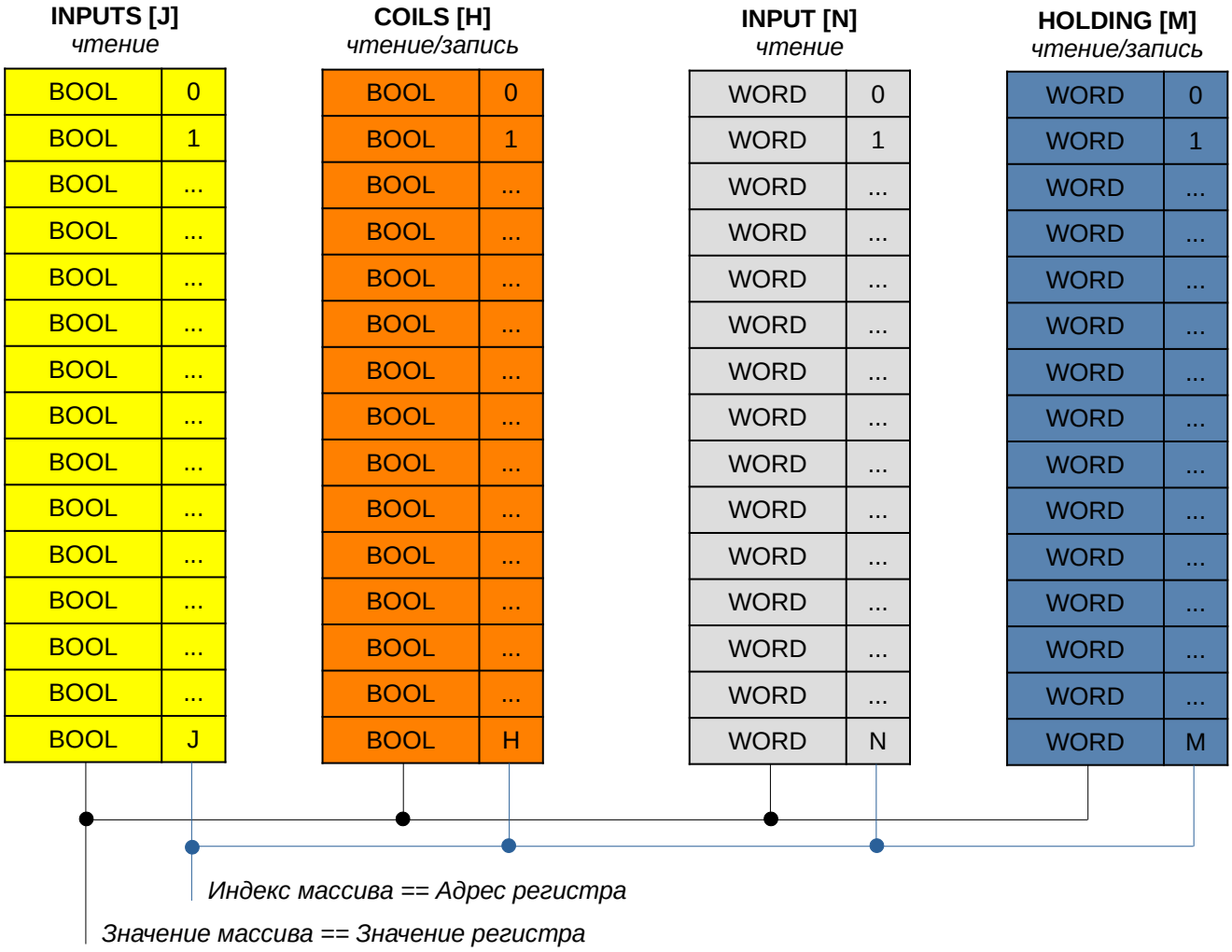
<table>
<tr><th rowspan="2">Таблица</th><th colspan="2">Код функции</th><th rowspan="2">Что хранится</th><th rowspan="2">Как хранится значение одного регистра в памяти (диапазон значений)</th></tr>
<tr><th>чтение</th><th>запись</th></tr>
<tr><td>COILS
(битовые флаги / катушки)</td><td>1</td><td>5, 15</td><td>• Выходы дискретные
• Битовые команды, уставки
• Битовые пользовательские данные</td><td rowspan="2">8-бит
BOOL
BYTE
UINT
(0, 1)</td></tr>
<tr><td>INPUTS
(битовые входы)</td><td>2</td><td></td><td>• Входы дискретные
• Битовые флаги, состояния</td></tr>
<tr><td>HOLDING REGISTERS
(числовые данные)</td><td>3</td><td>6, 16</td><td>• Выходы аналоговые
• Числовые настройки, уставки
• Числовые пользовательские данные</td><td rowspan="2">16-бит
WORD
(0 … 65535)</td></tr>
<tr><td>INPUT REGISTERS
(числовые входы)</td><td>4</td><td></td><td>• Аналоговые входы
• Числовые константы
• Числовые коды состояния</td></tr>
</table>

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ТАБЛИЦЫ ДАННЫХ

Спецификация протокола не определяет жесткого требования по количеству таблиц данных, которые должны быть определены в целевом устройстве (например, некоторые модели ПЛК имеют одну таблицу — HOLDING REGISTERS, где хранятся регистры бинарных и числовых типов с полным доступом «чтение/запись»).

Спецификация протокола не определяет жестких границ по размерам таблицы данных и адресации регистров в пределах таблицы (например, ПЛК может иметь одну таблицу HOLDING REGISTERS, где регистры с данными сгруппированы и расположены по секторам — адресация не с 0, между секторами имеются «пустоты»).



СЕТЕВОЙ ПРОТОКОЛ MODBUS

УПАКОВКА БИТОВЫХ ЗНАЧЕНИЙ В СЛОВА

Упаковка битовых регистров

- Алгоритмы:
  - логические операции: И, ИЛИ, НЕ и поразрядный сдвиг влево (распространенный)
  - битовые поля / bit-fields

Упаковка с помощью логических операций

- Значение (0, 1) бита раскладываются по разрядам группы чисел типа БАЙТ / BYTE
  - 1 BYTE = 8 бит = 1 байт (беззнаковое целое число)
    + 1-й бит > 0-й разряд БАЙТА
    + …
    + 8-й бит > 7-й разряд БАЙТА

БАЙТ = (ЗНАЧЕНИЕ БИТА << НОМЕР РАЗРЯДА) + БАЙТ*

где, БАЙТ — упакованный БАЙТ (новое значение)
ЗНАЧЕНИЕ БИТА — значение упаковываемого бита (0 или 1)
<< - порядрядный сдвиг влево
НОМЕР РАЗРЯДА — номер разряда БАЙТа (0 … 7), куда будет установлено ЗНАЧЕНИЕ БИТА
БАЙТ — упакованный БАЙТ (предыдущее значение)*

- Группа может состоять как из нескольких БАЙТ, так и из одного
- По сети БАЙТЫ передаются последовательно друг за другом
- Количество передаваемых БАЙТ указывается в поле DATA
- Количество передаваемых БАЙТ зависит от количества битовых регистров:

КОЛ-ВО БАЙТ = КОЛ-ВО БИТОВЫХ РЕГИСТРОВ / 8 БИТ ОДНОГО БАЙТА

где, полученное кол-во байт округляется вверх (например, 1,25 ~ 2)

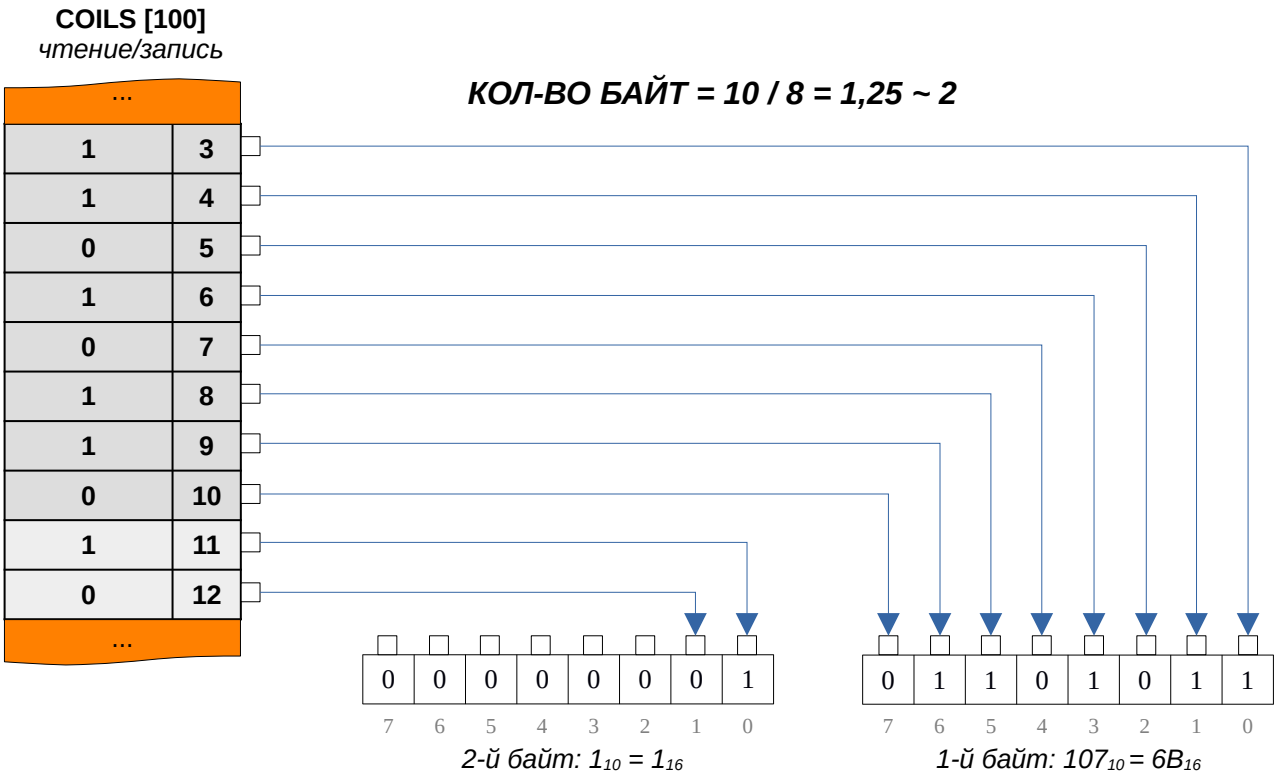
- Аналогичным способом возможна упаковка битовых регистров в числа больших разрядов, например, в СЛОВА / WORD (16 бит), ДВОЙНЫЕ СЛОВА / DWORD (32-бит) и т.д.

СЕТЕВОЙ ПРОТОКОЛ MODBUS

УПАКОВКА БИТОВЫХ ЗНАЧЕНИЙ В СЛОВА

Пример 1

- Упаковка битовых регистров из таблицы COILS:
  - адреса с 3 по 12
    (10 регистров)



1-й байт: $107_{10} = 6B_{16}$

НОМЕР РАЗРЯДА	ЗНАЧЕНИЕ БИТА	МАСКА = (С << В)			БАЙТ*			БАЙТ = (МАСКА+БАЙТ*)		
		DEC	HEX	BIN	DEC	HEX	BIN	DEC	HEX	BIN
0	1	1	01	0000 0001	0	00	0000 0000	1	01	0000 0001
1	1	2	02	0000 0010	1	01	0000 0001	3	03	0000 0011
2	0	0	00	0000 0000	3	03	0000 0011	3	03	0000 0011
3	1	8	08	0000 1000	3	03	0000 0011	11	0B	0000 1011
4	0	0	00	0000 0000	11	0B	0000 1011	11	0B	0000 1011
5	1	32	20	0010 0000	11	0B	0000 1011	43	2B	0010 1011
6	1	64	40	0100 0000	43	2B	0010 1011	107	6B	0110 1011
7	0	0	00	0000 0000	107	6B	0110 1011	107	6B	0110 1011

2-й байт: $1_{10} = 1_{16}$

НОМЕР РАЗРЯДА	ЗНАЧЕНИЕ БИТА	МАСКА = (С << В)			БАЙТ*			БАЙТ = (МАСКА+БАЙТ*)		
		DEC	HEX	BIN	DEC	HEX	BIN	DEC	HEX	BIN
0	1	1	01	0000 0001	0	00	0000 0000	1	01	0000 0001
1	0	0	00	0000 0000	1	01	0000 0001	1	01	0000 0001
2	0	0	00	0000 0000	1	01	0000 0001	1	01	0000 0001
3	0	0	00	0000 0000	1	01	0000 0001	1	01	0000 0001
4	0	0	00	0000 0000	1	01	0000 0001	1	01	0000 0001
5	0	0	00	0000 0000	1	01	0000 0001	1	01	0000 0001
6	0	0	00	0000 0000	1	01	0000 0001	1	01	0000 0001
7	0	0	00	0000 0000	1	01	0000 0001	1	01	0000 0001

СЕТЕВОЙ ПРОТОКОЛ MODBUS

РАЗЛОЖЕНИЕ ЧИСЕЛ НА СЛОВА

Разложение чисел на СЛОВА / WORD

- Сервер хранит числовые значения в таблицах:
  - размерность одной ячейки = 16 бит
  - тип данных одной ячейки - СЛОВО / WORD
- Алгоритмы:
  - логические операции: И, ИЛИ, НЕ и поразрядный сдвиг влево
  - «объединения» / union (распространенный)

  (аналогично выполняется разложение числа на СЛОВА для передачи по сети)

Объединение / Union

- Область памяти, которая используется для хранения значений разных типов
- Позволяет интерпретировать один и тот же набор битов по разному
- Объединение на языке C/C++ напоминает объявление структуры

Пример 1

- Определен новый тип данных union32_t:

```
typedef union {
    char     vChar;
    uint8_t  vByte;
    int16_t  vInt;
    uint16_t vWord;
    float    vFloat;
    uint32_t vDWord;
    uint8_t  bytes[4];
    uint16_t words[2];
} union32_t;
```

это объединение переменных следующих типов данных:

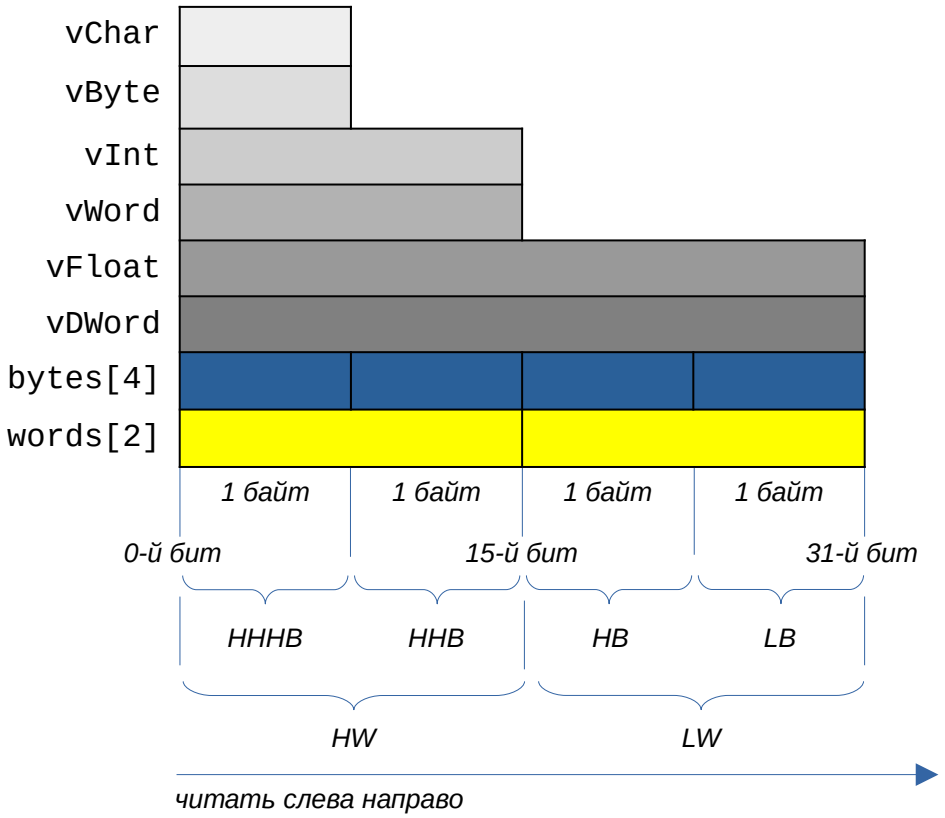
Имя переменной	**Тип данных**	**Размер** *(значения)*
vChar	CHAR	1 байт / 8 бит *(-128 … 127)*
vByte	BYTE	1 байт / 8 бит *(0 … 255)*
vInt	INT	2 байта / 16 бит *(-32768 … 32767)*
vWord	WORD	2 байта / 16 бит *(0 … 65535)*
vFloat	FLOAT REAL	4 байта / 32 бит *($3{,}4e^{-38}$ … $3{,}4e^{+38}$)*
vDWord	DWORD	4 байта / 32 бит *(0 … 4294967295)*
bytes[4]	BYTE[4]	массив из 4 байт
words[4]	WORD[4]	массив из 2 слов

СЕТЕВОЙ ПРОТОКОЛ MODBUS

РАЗЛОЖЕНИЕ ЧИСЕЛ НА СЛОВА

Пример 1 (продолжение)

- объединенные переменные занимают одну и ту же область памяти (***общая память***)
- объединенные переменные разделяют в общей памяти одно и то же значение (***общие данные***)
- каждая переменная перекрывает соизмеримое с ее типом пространство памяти и, соответственно, принимает данные (побитово) в пределах этих же границ
- размер общей памяти равен размеру наибольшей по количеству байт переменной
  = 4 байта / 32 бита



Где, LB — младший байт
HB — старший байт
…
LW — младшее слово
HW — старшее слово

Здесь представлен обычный (прямой) порядок следования байт и слов:
0-1 2-3
или
LB-HB-HHB-HHHB
(младший — старший — самый-старший — самый-самый-старший - …)

СЕТЕВОЙ ПРОТОКОЛ MODBUS

РАЗЛОЖЕНИЕ ЧИСЕЛ НА СЛОВА

Пример 1 (продолжение)

Зададим значение ***vDWord = 1300789005***

```
vChar (1 byte / 8 bit)
======================
  dec: 13
words: 30477
bytes: 13
 bits: 1011 0000

vInt (2 byte / 16 bit)
======================
  dec: 30477
words: 30477
bytes: 13 119
 bits: 1011 0000  1110 1110

vWord (2 byte / 16 bit)
======================
  dec: 30477
words: 30477
bytes: 13 119
 bits: 1011 0000  1110 1110

vFloat (4 byte / 32 bit)
======================
  dec: 286187936.000000
words: 30477 19848
bytes: 13 119 136 77
 bits: 1011 0000  1110 1110  0001 0001  1011 0010

vDWord (4 bytes / 32 bit)
======================
  dec: 1300789005
words: 30477 19848
bytes: 13 119 136 77
 bits: 1011 0000  1110 1110  0001 0001  1011 0010
```

читать слева направо

СЕТЕВОЙ ПРОТОКОЛ MODBUS

РАЗЛОЖЕНИЕ ЧИСЕЛ НА СЛОВА

Пример 1 (продолжение)

Зададим значение ***vFloat = -31.5***

```
vChar (1 byte / 8 bit)
==========================
  dec: 0
words: 0
bytes: 0
 bits: 0000 0000

vInt (2 byte / 16 bit)
==========================
  dec: 0
words: 0
bytes: 0 0
 bits: 0000 0000  0000 0000

vWord (2 byte / 16 bit)
==========================
  dec: 0
words: 0
bytes: 0 0
 bits: 0000 0000  0000 0000

vFloat (4 byte / 32 bit)
==========================
  dec: -31.500000
words: 0 49660
bytes: 0 0 252 193
 bits: 0000 0000  0000 0000  0011 1111  1000 0011

vDWord (4 bytes / 32 bit)
==========================
  dec: 3254517760
words: 0 49660
bytes: 0 0 252 193
 bits: 0000 0000  0000 0000  0011 1111  1000 0011
```

читать слева направо

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ПОРЯДОК СЛЕДОВАНИЯ БАЙТ В ЧИСЛОВЫХ ЗНАЧЕНИЯХ

Порядок следования слов и байт для 1-й и 2-байтовых числовых значений

Хранение в памяти Сервера:

- Значения располагаются в таблицах HOLDING и INPUTS REGISTERS
  - 1-байтовое значение занимает 1 СЛОВО / WORD
    (заняты только первые 8 бит слова, остальные 8 бит =0)
  - 2-байтовое значение занимает 1 СЛОВО / WORD
    (заняты все 16 бит слова)
- Порядок следования слов в памяти:
  - не имеет значения (может быть изменен при передаче)
  - ***1 значение занимает 1 регистр в таблице и 1 адрес***
- Порядок следования байт слова в памяти:
  - не имеет значения (может быть изменен при передаче)

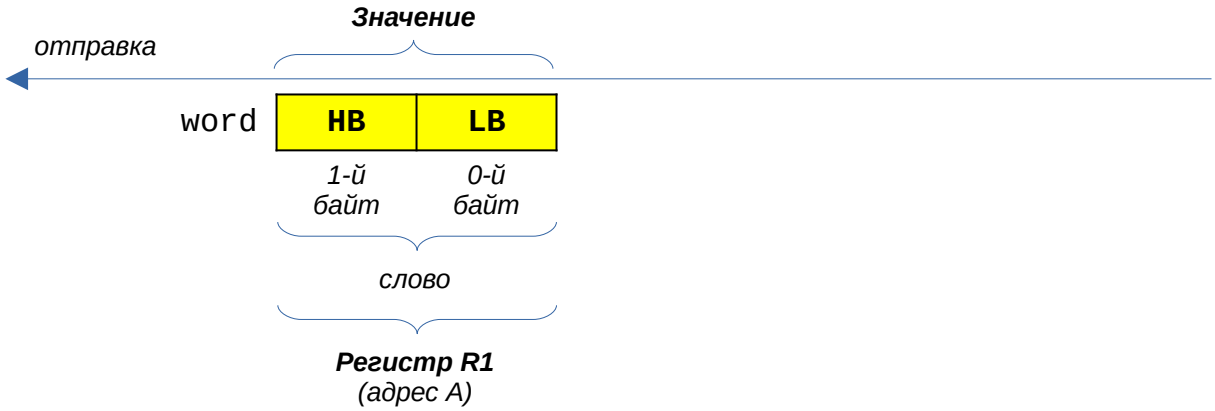
Хранение в памяти Клиента:

- Определяется на стороне Клиента

Передача по сети:

- Значения передаются словами побайтово
  - байт за байтом
  - слово за словом
- Порядок следования слов:
  - не имеет значения (каждое значение занимает 1 СЛОВО)
- Порядок следования байт в слове:
  - ***старший байт (HB) вперед***
  - ***младший байт (LB) далее***

1-0 *или* ***HB-LB***



- *Базовый порядок слов и байт при передаче определен стандартом протокола.*
- *Иной порядок необходимо отразить в карте адресов, т. к. Клиент должен знать его, чтобы при получении байт правильно собрать исходное значение.*
- *Например, на стороне Клиента порядок следования байт может задаваться в настройках OPC-сервера.*

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ПОРЯДОК СЛЕДОВАНИЯ СЛОВ

Порядок следования слов и байт для многобайтовых числовых значений

Хранение в памяти Сервера:

- Значения располагаются в таблицах HOLDING и INPUTS REGISTERS
  - 4-байтовое значение занимает 2 СЛОВА / WORD (заняты все 16 бит каждого слова)
  - 8-байтовое значение занимает 4 СЛОВА / WORD (заняты все 16 бит слова)
  - и так далее
- Порядок следования слов в памяти:
  - не имеет значения (может быть изменен при передаче)
  - ***1 значение занимает несколько регистров в таблице и несколько адресов***
- Порядок следования байт в слове:
  - не имеет значения (может быть изменен при передаче)

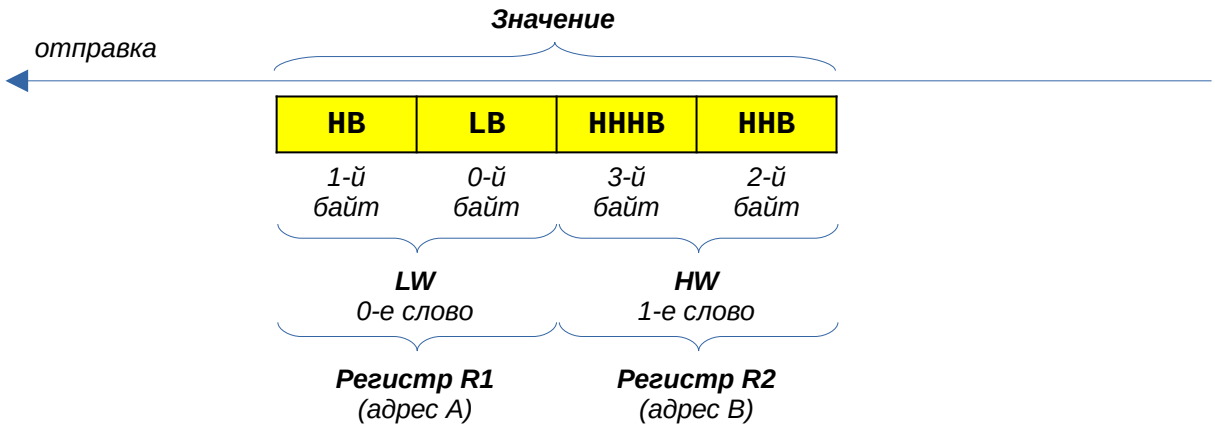
Хранение в памяти Клиента:

- Определяется на стороне Клиента

Передача по сети:

- Значения передаются словами побайтово
  - байт за байтом
  - слово за словом
- Порядок следования слов:
  - ***младшее слово (LW) вперед***
  - ***старшее слово (HW) далее***
- Порядок следования байт в слове:
  - ***старший байт (HB) вперед***
  - ***младший байт (LB) далее***

1-0 3-2 *или* ***HB-LB HHHB-HHB***



- *Базовый порядок слов и байт при передаче определен стандартом протокола.*
- *Иной порядок необходимо отразить в карте адресов, т. к. Клиент должен знать его, чтобы при получении байт правильно собрать исходное значение.*
- *Например, на стороне Клиента порядок следования байт может задаваться в настройках OPC-сервера.*

СЕТЕВОЙ ПРОТОКОЛ MODBUS

НАРУШЕНИЕ СТАНДАРТА

- Некоторые производители оборудования нарушают стандарт ModBus:
  - предлагают иной порядок следования байт числовых данных
  - предлагают иные таблицы данных
  - предлагают к передаче специальные структурированные типы данных
- Поэтому, перед приобретением того или иного оборудования, следует изучить его аппаратно-программные особенности.

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ПРИМЕР ТРАНЗАКЦИИ: ЧТЕНИЕ COILS

FC: 01 (0x01) Чтение битовых флагов / Read COILS

Запрос / Request

Поле	Размер	Значение
Адрес сервера	1 Байт	0x01
Код функции	1 Байт	0x01
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0x13
Количество регистров (HB)	2 Байта	0x00
Количество регистров (LB)		0x13
CRC (LB)	2 Байта	...
CRC (HB)		...

Ответ / Response

Адрес сервера	1 Байт	0x01
Код функции	1 Байт	0x01
Количество байт	1 Байт	0x03
1-й байт значений (19-26)	1 Байт	0xCD
2-й байт значений (27-34)	1 Байт	0x6B
3-й байт значений (35-37)	1 Байт	0x05
CRC (LB)	2 Байта	...
CRC (HB)		...

- Запрос:
  - серверу с адресом: 01 (0x01)
  - код функции: 01 (0x01)
  - чтение 19 регистров битовых флагов с адресами: 19 — 37

- Ответ:
  - сервера с адресом: 01 (0x01)
  - код функции: 01 (0x01)
  - 3 байта с упакованными значениями битовых флагов
    - 1-й байт содержит значения регистров с адресами: 19 — 26
    - 2-й байт содержит значения регистров с адресами: 27 — 34
    - 3-й байт содержит значения регистров с адресами: 35 - 37

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ПРИМЕР ТРАНЗАКЦИИ: ЧТЕНИЕ INPUTS

FC: 02 (0x02) Чтение битовых входов / Read INPUTS

Запрос / Request

Поле	Размер	Значение
Адрес сервера	1 Байт	0x01
Код функции	1 Байт	0x02
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0xC4
Количество регистров (HB)	2 Байта	0x00
Количество регистров (LB)		0x16
CRC (LB)	2 Байта	...
CRC (HB)		...

Ответ / Response

Адрес сервера	1 Байт	0x01
Код функции	1 Байт	0x02
Количество байт	1 Байт	0x03
1-й байт значений (196-203)	1 Байт	0xAC
2-й байт значений (204-211)	1 Байт	0xDB
3-й байт значений (212-217)	1 Байт	0x35
CRC (LB)	2 Байта	...
CRC (HB)		...

- Запрос:
  - серверу с адресом: 01 (0x01)
  - код функции: 02 (0x02)
  - чтение 22 регистра битовых входов с адресами: 196 — 217

- Ответ:
  - сервера с адресом: 01 (0x01)
  - код функции: 02 (0x02)
  - 3 байта с упакованными значениями битовых входов
    - 1-й байт содержит значения регистров с адресами: 196 — 203
    - 2-й байт содержит значения регистров с адресами: 204 — 211
    - 3-й байт содержит значения регистров с адресами: 212 - 217

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ПРИМЕР ТРАНЗАКЦИИ: ЧТЕНИЕ HOLDING REGISTERS

FC: 03 (0x03) Чтение числовых данных / Read HOLDING REGISTERS

Запрос / Request

Поле	Размер	Значение
Адрес сервера	1 Байт	0x10
Код функции	1 Байт	0x03
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0x6B
Количество регистров (HB)	2 Байта	0x00
Количество регистров (LB)		0x02
CRC (LB)	2 Байта	...
CRC (HB)		...

Ответ / Response

Адрес сервера	1 Байт	0x10
Код функции	1 Байт	0x03
Количество байт	1 Байт	0x04
Байт 1-го слова (107.HB)	1 Байт	0x02
Байт 1-го слова (107.LB)	1 Байт	0x2B
Байт 2-го слова (108.HB)	1 Байт	0x00
Байт 2-го слова (108.LB)	1 Байт	0x00
CRC (LB)	2 Байта	...
CRC (HB)		...

- Запрос:
  - серверу с адресом: 16 (0x10)
  - код функции: 03 (0x03)
  - чтение 2 регистров числовых данных с адресами: 107 — 108

- Ответ:
  - сервера с адресом: 16 (0x10)
  - код функции: 03 (0x03)
  - 4 байта (2 слова) с числовыми данными:
    - 1-й байт со старшей частью значения регистра с адреса: 107
    - 2-й байт с младшей частью значения регистра с адреса: 107
    - 3-й байт со старшей частью значения регистра с адреса: 108
    - 4-й байт с младшей частью значения регистра с адреса: 108

Если подразумевалась передача значения 4-байтного числа, которое располагается в 2-х регистрах, то, после получения, его необходимо привести к общему виду — объединить.

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ПРИМЕР ТРАНЗАКЦИИ: ЧТЕНИЕ INPUT REGISTERS

FC: 04 (0x04) Чтение числовых входов / Read INPUT REGISTERS

Запрос / Request

Поле	**Размер**	**Значение**
Адрес сервера	1 Байт	0x10
Код функции	1 Байт	0x04
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0x08
Количество регистров (HB)	2 Байта	0x00
Количество регистров (LB)		0x01
CRC (LB)	2 Байта	...
CRC (HB)		...

Ответ / Response

Адрес сервера	1 Байт	0x10
Код функции	1 Байт	0x04
Количество байт	1 Байт	0x01
Байт 1-го слова (8.HB)	1 Байт	0x00
Байт 1-го слова (8.LB)	1 Байт	0x0A
CRC (LB)	2 Байта	...
CRC (HB)		...

- Запрос:
  - серверу с адресом: 16 (0x10)
  - код функции: 04 (0x04)
  - чтение 1 регистра числовых входов с адресом: 8

- Ответ:
  - сервера с адресом: 16 (0x10)
  - код функции: 04 (0x04)
  - 2 байта (1 слово) с числовыми данными:
    - 1-й байт со старшей частью значения регистра с адреса: 8
    - 2-й байт с младшей частью значения регистра с адреса: 8

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ПРИМЕР ТРАНЗАКЦИИ: ЗАПИСЬ COIL

FC: 05 (0x05) Запись одного битового флага / Write single COIL

Запрос / Request

Поле	Размер	Значение
Адрес сервера	1 Байт	0x0A
Код функции	1 Байт	0x05
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0xAC
Значение (172.HB)	2 Байта	0xFF
Значение (172.LB)		0x00
CRC (LB)	2 Байта	...
CRC (HB)		...

Ответ / Response

Адрес сервера	1 Байт	0x0A
Код функции	1 Байт	0x05
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0xAC
Значение (172.HB)	2 Байта 1 Байт 1 Байт 1 Байт 1 Байт 1 Байт 1 Байт	0xFF
Значение (172.LB)		0x00
CRC (LB)	2 Байта	...
CRC (HB)		...

- Запрос:
  - серверу с адресом: 10 (0x0A)
  - код функции: 05 (0x05)
  - запись значения TRUE (0xFF00) регистра битовых флагов с адресом: 172

- Ответ:
  - сервера с адресом: 10 (0x0A)
  - код функции: 05 (0x05)
  - записано значение TRUE (0xFF00) регистра битовых флагов с адресом: 172

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ПРИМЕР ТРАНЗАКЦИИ: ЗАПИСЬ COIL

FC: 15 (0x0F) Запись нескольких битовых флагов / Write multiple COILS

Запрос / Request

Поле	Размер	Значение
Адрес сервера	1 Байт	0x20
Код функции	1 Байт	0x0F
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0x13
Количество регистров (HB)	2 Байта	0x00
Количество регистров (LB)		0x0A
Количество байт	1 Байт	0x02
Значение (HB) (27-29)	2 Байта	0xCD
Значение (LB) (19-26)		0x01
CRC (LB)	2 Байта	...
CRC (HB)		...

Ответ / Response

Адрес сервера	1 Байт	0x20
Код функции	1 Байт	0x0F
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0x13
Количество регистров (HB)	2 Байта	0x00
Количество регистров (LB)		0x0A
CRC (LB)	2 Байта	...
CRC (HB)		...

- Запрос:
  - серверу с адресом: 32 (0x20)
  - код функции: 15 (0x0F)
  - запись значений 10 (0x0A) регистров битовых флагов с адресами: 19 — 29

- Ответ:
  - сервера с адресом: 32 (0x20)
  - код функции: 15 (0x0F)
  - записаны значения 10 (0x0A) регистров битовых флагов с адресами: 19 - 29

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ПРИМЕР ТРАНЗАКЦИИ: ЗАПИСЬ HOLDING REGISTERS

FC: 06 (0x06) Запись одного числового значения / Write single HOLDING REGISTER

Запрос / Request

Поле	Размер	Значение
Адрес сервера	1 Байт	0x0A
Код функции	1 Байт	0x06
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0x01
Значение (1.HB)	2 Байта	0x00
Значение (1.LB)		0x03
CRC (LB)	2 Байта	...
CRC (HB)		...

Ответ / Response

Адрес сервера	1 Байт	0x0A
Код функции	1 Байт	0x06
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0x01
Значение (1.HB)	2 Байта	0x00
Значение (1.LB)		0x03
CRC (LB)	2 Байта	...
CRC (HB)		...

- Запрос:
  - серверу с адресом: 10 (0x0A)
  - код функции: 06 (0x06)
  - запись значения одного регистра числовых данных с адресом: 1

- Ответ:
  - сервера с адресом: 10 (0x0A)
  - код функции: 06 (0x06)
  - записано значение одного регистра числовых данных с адресом: 1

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ПРИМЕР ТРАНЗАКЦИИ: ЗАПИСЬ HOLDING REGISTERS

FC: 16 (0x10) Запись нескольких числовых данных / Write multiple HOLDINGS

Запрос / Request

Поле	Размер	Значение
Адрес сервера	1 Байт	0x20
Код функции	1 Байт	0x10
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0x01
Количество регистров (HB)	2 Байта	0x00
Количество регистров (LB)		0x02
Количество байт	1 Байт	0x04
Байт 1-го слова (1.HB)	1 Байт	0x00
Байт 1-го слова (1.LB)	1 Байт	0x0A
Байт 2-го слова (2.HB)	1 Байт	0x01
Байт 2-го слова (2.LB)	1 Байт	0x02
CRC (LB)	2 Байта	...
CRC (HB)		...

Ответ / Response

Адрес сервера	1 Байт	0x20
Код функции	1 Байт	0x10
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0x01
Количество регистров (HB)	2 Байта	0x00
Количество регистров (LB)		0x02
CRC (LB)	2 Байта	...
CRC (HB)		...

- Запрос:
  - серверу с адресом: 32 (0x20)
  - код функции: 16 (0x10)
  - запись значений 2 (0x02) регистров числовых данных с адресами: 1 — 2

- Ответ:
  - сервера с адресом: 32 (0x20)
  - код функции: 16 (0x10)
  - записаны значения 2 (0x02) регистров числовых данных с адресами: 1 - 2

СЕТЕВОЙ ПРОТОКОЛ MODBUS

ПРИМЕР ТРАНЗАКЦИИ: КОД ОШИБКИ

Код ошибки

Запрос / Request

Поле	Размер	Значение
Адрес сервера	1 Байт	0x10
Код функции	1 Байт	0x03
Стартовый адрес регистра (HB)	2 Байта	0x00
Стартовый адрес регистра (LB)		0x6B
Количество регистров (HB)	2 Байта	0x00
Количество регистров (LB)		0x02
CRC (LB)	2 Байта	...
CRC (HB)		...

Ответ / Response

Адрес сервера	1 Байт	0x10
Код ошибки (= 128 + Код функции)	1 Байт	0x83 (= 0x80 + 0x03)
Код исключительной ситуации	1 Байт	0x02
CRC (LB)	2 Байта	...
CRC (HB)		...

- Запрос:
  - серверу с адресом: 16 (0x10)
  - код функции: 03 (0x03)
  - чтение 2 регистров числовых данных с адресами: 107 — 108

- Ответ:
  - сервера с адресом: 16 (0x10)
  - код ошибки: 131 (0x83)
  - код исключительной ситуации: 02 (0x02)
    = Адрес регистра данных недоступен / ILLEGAL DATA ADDRESS

СЕТЕВОЙ ПРОТОКОЛ MODBUS

КАРТА АДРЕСОВ РЕГИСТРОВ

- Таблица с описанием адресов регистров

ИМЯ ТАБЛИЦЫ РЕГИСТРОВ
(коды функций: чтение=hex-код, запись=hex-код)

Адрес		**Порядок слов / байт** *(для чисел)*	**Тип данных** *(размерность в байтах)*	**Описание**
dec	hex			
97	0x61	1-0	FLOAT (4 байта)	Вес материала = 0,0 … 1000,0 кг AI01 / тензодатчик 2-WT-001 = 4 … 20 мА
98	0x62	3-2		

где,

ИМЯ ТАБЛИЦЫ РЕГИСТРОВ
- INPUTS
- COILS
- INPUT REGISTERS
- HOLDING REGISTERS

+ (указание поддерживаемых кодов функций на чтение/запись в системе hex)

Адрес регистра:
- dec — в десятичной системе счисления
- hex — в шестнадцатеричной системе счисления

Порядок байт / слов (для числовых значений)
- порядок слов и байт слов, в котором они передаются по сети (по запросу)
- информация необходима для Клиента, чтобы правильно «собрать» значение
  например: 1-0 — младшее слово старшим байтом вперед
  3-2 — старшее слово старшим байтом вперед

Тип данных (размерность байт)
- тип данных, в котором хранится значение
- (в скобках) необходимо указать размерность этого типа данных в байтах

Описание
- краткое описание значения
  = диапазон значений + единица измерения
  = значение «по-умолчанию»
  (для каждого битового значения дается описание: например, 0 — выкл., 1 — вкл.)
- краткое описание источника сигнала
  (номер канала В/В / датчик, исполнительный механизм, константа)
  = диапазон сигнала + единица измерения

СЕТЕВОЙ ПРОТОКОЛ MODBUS

КАРТА АДРЕСОВ РЕГИСТРОВ

Пример

COILS
(коды функций: чтение=0x01, запись=0x05,0x0F)

Адрес		**Порядок слов / байт** *(для чисел)*	**Тип данных** *(размерность)*	**Описание**
dec	hex			
0010	0x000A	1-0	BOOL (1 байт)	Команда: Возврат к заводским настройкам = 0 — нет (по-умолчанию) = 1 — да (после выполнения команды: = 0)
0011	0x000B		BOOL (1 байт)	Команда: Сброс таймера безопасного состояния каналов вывода = 0 — нет (по-умолчанию) = 1 — да (после выполнения команды: = 0)
0012	0x000C		BOOL (1 байт)	Команда: Принять настройки интерфейса COM1 = 0 — нет (по-умолчанию) = 1 — да (после выполнения команды: = 0)
0013	0x000D		BOOL (1 байт)	Команда: Перезагрузить устройство = 0 — нет (по-умолчанию) = 1 — да (после выполнения команды: = 0)
100	0x0064		BOOL (1 байт)	Канал DI0: Режим работы = 0 — выключен (по-умолчанию) = 1 — включен + EEPROM (хранимое)
200	0x00C8		BOOL (1 байт)	Канал AI0: Режим работы = 0 — выключен (по-умолчанию) = 1 — включен + EEPROM (хранимое)
300	0x012C		BOOL (1 байт)	Канал DO0: Режим работы = 0 — выключен (по-умолчанию) = 1 — включен + EEPROM (хранимое)
301	0x012D		BOOL (1 байт)	Канал DO0: Уровень = 0 — низкий (FALSE) (по-умолчанию) = 1 — высокий (TRUE)
302	0x012E		BOOL (1 байт)	Канал DO0: Уровень безопасного состояния = 0 — низкий (FALSE) (по-умолчанию) = 1 — высокий (TRUE) + EEPROM (хранимое)
400	0x0190		BOOL (1 байт)	Канал AO0: Режим работы = 0 — выключен (по-умолчанию) = 1 — включен (TRUE) + EEPROM (хранимое)

СЕТЕВОЙ ПРОТОКОЛ MODBUS

КАРТА АДРЕСОВ РЕГИСТРОВ

Пример

INPUTS
(коды функций: чтение=0x02)

Адрес		**Порядок слов / байт** *(для чисел)*	**Тип данных** *(размерность)*	**Описание**
dec	hex			
0010	0x000A		BOOL (1 байт)	Состояние управляющей программы = 0 — останов (по-умолчанию) = 1 — работа
0011	0x000B		BOOL (1 байт)	Состояние 1-го цикла управляющей программы = 0 — не выполнен (по-умолчанию) = 1 — выполнен
0012	0x000C		BOOL (1 байт)	Результат выполнения команды Возврат к заводским настройкам = 0 — не выполнено (по-умолчанию) = 1 — выполнено
0013	0x000D		BOOL (1 байт)	Результат выполнения команды Сброс таймера безопасного состояния каналов вывода = 0 — не выполнено (по-умолчанию) = 1 — выполнено
0014	0x000E		BOOL (1 байт)	Результат выполнения команды Принять настройки интерфейса COM1 = 0 — не выполнено (по-умолчанию) = 1 — выполнено
0100	0x0064		BOOL (1 байт)	Канал DI0: Уровень = 0 — низкий (FALSE) (по-умолчанию) = 1 — высокий (TRUE)

СЕТЕВОЙ ПРОТОКОЛ MODBUS

КАРТА АДРЕСОВ РЕГИСТРОВ

Пример

HOLDING REGISTERS
(коды функций: чтение=0x03, запись=0x06,0x10)

Адрес		**Порядок слов / байт** *(для чисел)*	**Тип данных** *(размерность)*	**Описание**
dec	hex			
0020	0x0014	1-0	WORD (2 байт)	Адрес устройства в сети ModBus = 1 (по-умолчанию) = 1 … 250 + EEPROM (хранимое)
0021	0x0015	1-0	WORD (2 байт)	Настройки сетевого интерфейса COM1 = 1 (по-умолчанию) + EEPROM (хранимое) * см. описание ниже
0022	0x0016	1-0	WORD (2 байт)	Таймаут перевода каналов вывода в безопасное состояние, секунды = 0 сек. — выключено (по-умолчанию) = 1 … 65535 сек. — включено на заданное время + EEPROM (хранимое)
0200	0x00C8	1-0	FLOAT (4 байт)	Канал AO0: Уровень, В = 0,0 В (по-умолчанию) = 0,0 … 10,0 В
0201	0x00C9	3-2		
0202	0x00CA	1-0	FLOAT (4 байт)	Канал AO0: Уровень безопасного состояния, В = 0,0 В (по-умолчанию) = 0,0 … 10,0 В + EEPROM (хранимое)
0203	0x00CB	3-2		
1000	0x03E8	1-0	WORD (2 байт)	Пользовательское значение 1 + EEPROM (хранимое)
1001	0x03E9	1-0	WORD (2 байт)	Пользовательское значение 2 + EEPROM (хранимое)
1002	0x03EA	1-0	WORD (2 байт)	Пользовательское значение 3 + EEPROM (хранимое)
1003	0x03EB	1-0	WORD (2 байт)	Пользовательское значение 4 + EEPROM (хранимое)
1004	0x03EC	1-0	WORD (2 байт)	Пользовательское значение 5
1005	0x03ED	1-0	WORD (2 байт)	Пользовательское значение 6
1006	0x03EE	1-0	WORD (2 байт)	Пользовательское значение 7
1007	0x03EF	1-0	WORD (2 байт)	Пользовательское значение 8

СЕТЕВОЙ ПРОТОКОЛ MODBUS

КАРТА АДРЕСОВ РЕГИСТРОВ

Пример

21: Настройки сетевого интерфейса COM1 (RS-485)

Группа битовых настроек, упакованных в 16-битное целое число (СЛОВО / WORD).
Значение каждого бита (группы битов) определено:

0	0	0	0	BO2	BO1	BO0	MODE
15	14	13	12	11	10	9	8

SBITS	PRTY1	PRTY0	DBITS	BAUD3	BAUD2	BAUD1	BAUD0
7	6	5	4	3	2	1	0

где,

BAUD3-0 – скорость (бит/сек):
= 0000 (0) – 1200
= 0001 (1) – 2400
= 0010 (2) – 4800
= 0011 (3) – 9600
= 0100 (4) – 19200
= 0101 (5) – 38400
= 0110 (6) – 57600
= 0111 (7) – 115200

DBITS – количество битов данных:
= 00 (0) – 8
= 01 (1) – 9

PRTY1-0 – бит четности:
= 00 (0) – нет (None)
= 01 (1) – нечетный (Odd)
= 11 (3) – четный (Even)

SBITS – количество стоп-битов:
= 0 – 1
= 1 – 2

MODE – режим работы:
= 0 – Slave
= 1 – Master

BO2-0 - порядок следования байт для двух- или четырех-байтных значений:
= 000 (0) – «1-0» или «1-0 3-2» (старший байт вперед, младшее слово вперед),
= 001 (1) – «0-1» или «0-1 2-3» (младший байт вперед, младшее слово вперед),
= 010 (2) – «3-2 1-0» (старший байт вперед, старшее слово вперед),
= 011 (3) – «2-3 0-1» (младший байт вперед, старшее слово вперед).

Количество бит данных постояное: «8»

СЕТЕВОЙ ПРОТОКОЛ MODBUS

КАРТА АДРЕСОВ РЕГИСТРОВ

Пример

INPUT REGISTERS
(коды функций: чтение=0x04)

Адрес		**Порядок слов / байт** *(для чисел)*	**Тип данных** *(размерность)*	**Описание**
dec	hex			
0020	0x0014	1-0	WORD (2 байт)	Версия аппаратной части ПЛК (схемотехника) = 3
0021	0x0015	1-0	WORD (2 байт)	Версия программной части ПЛК (целевая система) = 11
0022	0x0016	1-0	WORD (2 байт)	Год выпуска ПЛК = 2021
0200	0x00C8	1-0	FLOAT (4 байт)	Температура ЦПУ ПЛК, °C Встроенный датчик температуры = -20,0 … 120,0 °C
0201	0x00C9	3-2		
0202	0x00CA	1-0	FLOAT (4 байт)	Канал AI0: Уровень, В = 0,0 В (по-умолчанию) = 0,0 … 10,0 В
0203	0x00CB	3-2		