ООО «АВТОМАТИКА»

СЕТЬ ПРИБОРОВ

ПРОТОКОЛ MODBUS

Версия 1.3 от 17.05.2010

Санкт-Петербург
2010 г.

Содержание

Введение

В этом руководстве описываются основные принципы построения RS485 сети приборов, работающей по протоколу Modbus. Описываются реализованные в приборах функции. Представлены карты адресных пространств памяти приборов.

Целью данного руководства не является дублирование текстов стандартов. Здесь лишь акцентируется внимание на ключевых моментах. Детально изучить тонкости протокола и интерфейса можно ознакомившись с литературой, указанной в разделе “Использованные источники информации”.

1. Интерфейс RS-485

Стандарт ANSI TIA/EIA-485, более известный как RS485, определяет сбалансированный способ надёжной передачи данных на длинные расстояния в условиях промышленных помех. Также стандарт определяет топологию сети и описывает способы согласования полного сопротивления линии интерфейса и предоставляет результаты лабораторных тестов.

Физически, интерфейс RS485 является дифференциальным, обеспечивает многоточечные соединения и позволяет передавать и принимать данные в обоих направлениях.

Упрощённо, сеть интерфейса RS485 представляет собой приемопередатчики, соединенные при помощи витой пары - двух скрученных проводов (см. рис.1).

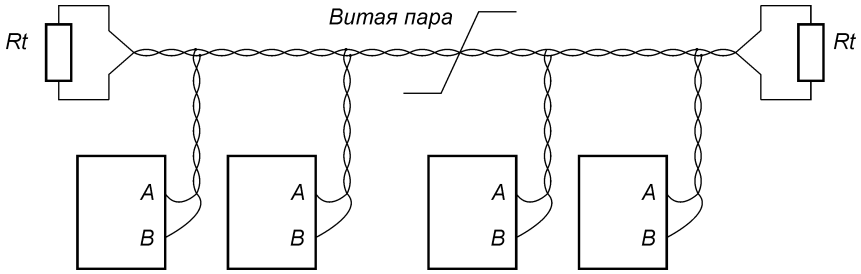

Рис.1 Структура сети RS-485

В основе интерфейса RS485 лежит принцип дифференциальной передачи сигнала. Суть его заключается в передаче одного сигнала по двум проводам. Причем по одному проводу (условно А) идет оригинальный сигнал, а по другому (условно В) - его инверсная копия.

Сигналы линий (А и В) интерфейса RS485 Таблица 1.

Напряжение	B > A	A > B
Двоичное состояние	1	0
Напряжение на выходе передатчика	A +1 Вольт B +4 Вольта	A +4 Вольта B +1 Вольт

Типовая разница напряжений между линиями А и В передатчика равна 3В, минимальная 1.5В, максимальная 5В.

Разница напряжений между линиями А и В на приёмнике должна быть не менее 0.2В и абсолютная разница потенциалов относительно общего провода должно быть в диапазоне (–7…+12) В.

Таким образом, между двумя проводами витой пары всегда есть разность потенциалов. Именно этой разностью потенциалов и передается сигнал. Такой способ передачи обеспечивает высокую устойчивость к синфазной помехе. Максимальная скорость связи прибора по интерфейсу RS485 может достигать нескольких Мбод. Максимальное расстояние - 1200 метров. Если необходимо организовать связь на расстоянии больше чем 1200 метров или подключить больше устройств, чем допускает нагрузочная способность передатчика - применяют специальные повторители (репитеры). Типовое правило для расчёта максимальной длины линии связи таково: произведение скорости передачи в бодах на длину в метрах должно дать результат не более чем 10^8.

При значительных расстояниях между устройствами, связанными по витой паре или высоких скоростях передачи начинают проявляться так называемые эффекты длинных линий. Электромагнитный сигнал имеет свойство отражаться от открытых концов линии передачи и ее ответвлений. Фронт сигнала, отразившийся от конца линии, может исказить

текущий или следующий сигнал. В таких случаях нужно подавлять эффект отражения.

Существуют стандартные решения этой проблемы (R, RC - терминаторы). У любой линии связи есть такой параметр, как волновое сопротивление Zв. Оно зависит от характеристик используемого кабеля и не зависит от его длины. Для обычно применяемых в линиях связи витых пар волновое сопротивление Zв составляет (90-120) Ом. Рассмотрим варианты:

1. Если на удаленном конце линии, между проводниками витой пары включить резистор с номинальным омическим сопротивлением равным волновому сопротивлению линии, то электромагнитная волна, дошедшая до «тупика» поглощается на таком резисторе. Отсюда его названия - согласующий резистор или «терминатор».

Помимо достоинств этого метода (повышение скорости, увеличение длины и подавление отражений), есть и недостатки (дополнительная нагрузка на драйверы повышает энергопотребление, остальные ответвления линии продолжают вносить искажения, драйвер приёмника находится в неоднозначном состоянии: либо режим ожидания, либо режим приёма).

2. Если на удалённом конце вместо резистора установить RC цепочку R=(90-120) Ом, C=1000 пФ, то можно устранить проблему повышенного энергопотребления и проблему неопределённости драйвера приёмника (для приёмников с функциями open-line и fail-safe). Но из-за постоянной времени RC цепи, максимальная скорость передачи и длинна линии будут меньшими.

Эффект отражения и необходимость правильного согласования накладывают ограничения на конфигурацию линии связи (топология сети). Линия связи должна представлять собой один кабель витой пары. К этому кабелю присоединяются все приемники и передатчики (гирлянда). Расстояние от линии до микросхем интерфейса RS485 должно быть как можно короче, так как длинные ответвления вносят рассогласование и вызывают отражения. В оба наиболее удаленных конца кабеля включают терминаторы. Калибр витой пары достаточно не более AWG24.

В случае наличия клемм заземления у приборов, логичным является их заземление. Но из-за этого возможно возникновение паразитных токов во время кратковременных высоковольтных помех. Для устранения этого эффекта необходимо применять следующую схему заземления с ограничительными резисторами.

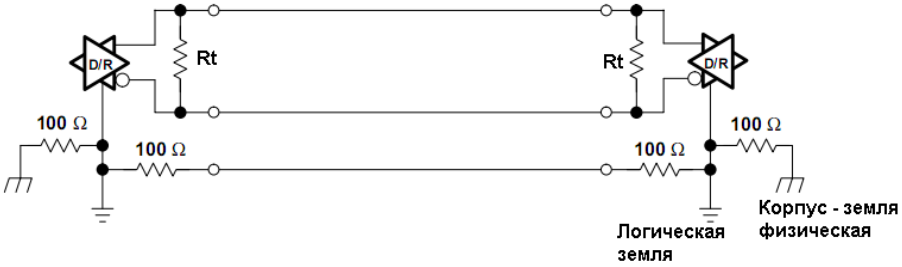

Рис.2 Структура сети RS485 с заземлением

Следует также сказать, что к линии интерфейса все устройства подключаются через специализированные микросхемы (драйверы интерфейса RS485). Они могут быть разных производителей и с различными техническими параметрами и как следствие различной стоимости. Эти драйверы в значительной степени могут определять эксплуатационные свойства приборов: дальность передачи, количество приборов в одном участке сети и надёжность передачи.

2. Протокол MODBUS

MODBUS – это протокол уровня приложений (уровень 7 модели OSI), что обеспечивает связь между устройствами, соединёнными различными каналами связи и сетями.

Де-факто, MODBUS является стандартом в сетях промышленного назначения с 1979 года. Он обеспечивает связь миллионам устройств во всём мире, в том числе и через Интернет. Есть различные реализации протокола:

- Для асинхронных беспроводных, оптических и проводных каналов связи (RS-232, RS-485, RS-422).
- Для TCP/IP (порт 502) через интернет
- MODBUS-PLUS - для высокоскоростных сетей с передачей меток (high speed token passing network).

Кроме того, разнородные участки сетей могут объединяться шлюзами (специальными конверторами).

Для асинхронных последовательных каналов связи существует две реализации MODBUS-SERIAL-LINE протокола MODBUS-RTU и MODBUS-ASCII (уровень 1 и 2 модели OSI). Разница между ними заключается в способе кодировки данных, способе синхронизации фреймов, и алгоритме обеспечения целостности данных. В нашем случае, в сети RS485 обмен данными реализован посредством протокола MODBUS-RTU. Далее по тексту будем рассматривать ситуацию только в этом аспекте.

MODBUS-SERIAL-LINE протокол – это протокол типа MASTER-SLAVE (протокол запросов-ответов). Ведущий в сети (MASTER) всегда один. Каждый подчинённый (SLAVE) должен иметь уникальный номер 1-247. Адрес 0 – это широковещательный запрос, адресованный сразу всем подчинённым. Таким образом, логически в одном участке сети может быть до 248 устройств (включая MASTER). Каждый запрос содержит код функции. Под MODBUS-функциями понимают определённые сервисы предоставляемые подчинёнными ведущему. Таким образом, роль клиента играет MASTER, а роль сервера, с определённым набором функций-сервисов, SLAVE.

3. Функции протокола MODBUS

Каждый SLAVE может содержать уникальный набор функций-сервисов, но есть и ряд стандартных функций, которые подробно описаны в документе “MODBUS Application Protocol Specification” (www.modbus.org). Также полезная информация может быть найдена в документе “MODBUS over serial line specification and implementation guide” (www.modbus.org).

Поддерживаемые нами функции (см. табл. 3.1-3.2)

Стандартные функции. Таблица 3.1

Функция	Название	Тип данных (big-endian)
1	read coils	sram bits
2	read discrete inputs	sram bits
3	read holding registers	eeprom words
4	read input registers	sram words
5	write single coil	sram bit
16	write holding registers	eeprom words

Пользовательские функции. Таблица 3.2

Функция	Название	Тип данных (little-endian)
100	read eeprom bytes	eeprom byte
101	write eeprom bytes	eeprom byte
102	read dataflash bytes	dataflash byte
104	read sram bytes	sram byte
108	service function	byte
109	archive function	byte

В более ранних версиях приборов (до 2010г) были реализованы лишь пользовательские функции, но со временем стало понятно, что для обеспечения совместного использования приборов с ПЛК (минуя ПК) необходимы и стандартные функции. Будьте внимательны и обратите внимание на то, что стандартные функции оперируют только со словами (16-бит) и в формате big-endian.

Далее будут в таблицах 3.3-3.8 будут представлены развёрнутые описания пользовательских функций.

Стандартные функции (см. таб. *3.1*) подробно описаны в документе “MODBUS Application Protocol Specification” (www.modbus.org).

Пользовательская функция 100. Таблица 3.3

Функция 100. Чтение энергонезависимой памяти (EEPROM)				
байт	**описание**	**значение**	**порядок**	**размер**
Фрейм ЗАПРОСА:				
0	slave's adr	1-247		1 byte
1	function	100		1 byte
2	address	0-END	low	2 bytes
3			high	
4	bytes qty	1-248	low	2 bytes
5			high	
6	CRC16		low	2 bytes
7			high	
Фрейм ОТВЕТА:				
0	slave's adr	1-247		1 byte
1	function	100		1 byte
2	address	0-END	low	2 bytes
3			high	
4	bytes qty	1-248	low	2 bytes
5			high	
6	data (byte0)	(1-248) байт данных		1 byte
7	data (byte1)			1 byte
…	…			1 byte
252	data (byte246)			1 byte
253	data (byte247)			1 byte
254	CRC16		low	2 bytes
255			high	
Фрейм ОТВЕТА С КОДОМ ОШИБКИ:				
0	slave's adr	1-247		1 byte
1	function	100 + 0x80		1 byte
2	exception code	1	wrong function	1 byte
		2	wrong address	
		3	wrong bytes qty	
		4	read error	
3	CRC16		low	2 bytes
4			high	

Пользовательская функция 101. Таблица 3.4

Функция 101. Запись в энергонезависимую память (EEPROM)				
байт	**описание**	**значение**	**порядок**	**размер**
Фрейм ЗАПРОСА:				
0	slave's adr	1-247		1 byte
1	function	101		1 byte
2	address	0-END	low	2 bytes
3			high	
4	bytes qty	1-248	low	2 bytes
5			high	
6	data (byte0)	(1-248) байт данных		1 byte
7	data (byte1)			1 byte
…	…			1 byte
252	data (byte246)			1 byte
253	data (byte247)			1 byte
254	CRC16		low	2 bytes
255			high	
Фрейм ОТВЕТА:				
0	slave's adr	1-247		1 byte
1	function	101		1 byte
2	address	0-2047	low	2 bytes
3			high	
4	bytes qty	1-248	low	2 bytes
5			high	
6	CRC16		low	2 bytes
7			high	
Фрейм ОТВЕТА С КОДОМ ОШИБКИ:				
0	slave's adr	1-247		1 byte
1	function	101 + 0x80		1 byte
2	exception code	1	wrong function	1 byte
		2	wrong address	
		3	wrong bytes qty	
		4	write error	
3	CRC16		low	2 bytes
4			high	

Пользовательская функция 102. Таблица 3.5

Функция 102. Чтение энергонезависимой памяти (DATA_FLASH)				
байт	**описание**	**значение**	**порядок**	**размер**
Фрейм ЗАПРОСА:				
0	slave's adr	1-247		1 byte
1	function	102		1 byte
2	page	0-8191	low	2 bytes
3			high	
4	address	0-527	low	2 bytes
5			high	
6	bytes qty	1-246	low	2 bytes
7			high	
8	CRC16		low	2 bytes
9			high	
Фрейм ОТВЕТА:				
0	slave's adr	1-247		1 byte
1	function	102		1 byte
2	page	0-8191	low	2 bytes
3			high	
4	address	0-527	low	2 bytes
5			high	
6	bytes qty	1-246	low	2 bytes
7			high	
8	data (byte0)	(1-246) байт данных		1 byte
9	data (byte1)			1 byte
…	…			1 byte
252	data (byte244)			1 byte
253	data (byte245)			1 byte
254	CRC16		low	2 bytes
255			high	
Фрейм ОТВЕТА С КОДОМ ОШИБКИ:				
0	slave's adr	1-247		1 byte
1	function	102 + 0x80		1 byte
2	exception code	1	wrong function	1 byte
		2	wrong address	
		3	wrong bytes qty	
		4	read error	
3	CRC16		low	2 bytes
4			high	

Пользовательская функция 104. Таблица 3.6

Функция 104. Чтение оперативной памяти (SRAM)				
байт	**описание**	**значение**	**порядок**	**размер**
Фрейм ЗАПРОСА:				
0	slave's adr	1-247		1 byte
1	function	104		1 byte
2	address	0-END	low	2 bytes
3			high	
4	bytes qty	1-248	low	2 bytes
5			high	
6	CRC16		low	2 bytes
7			high	
Фрейм ОТВЕТА:				
0	slave's adr	1-247		1 byte
1	function	104		1 byte
2	address	0-END	low	2 bytes
3			high	
4	bytes qty	1-248	low	2 bytes
5			high	
6	data (byte0)	(1-248) байт данных		1 byte
7	data (byte1)			1 byte
…	…			1 byte
252	data (byte246)			1 byte
253	data (byte247)			1 byte
254	CRC16		low	2 bytes
255			high	
Фрейм ОТВЕТА С КОДОМ ОШИБКИ:				
0	slave's adr	1-247		1 byte
1	function	104 + 0x80		1 byte
2	exception code	1	wrong function	1 byte
		2	wrong address	
		3	wrong bytes qty	
		4	read error	
3	CRC16		low	2 bytes
4			high	

Пользовательская функция 108. Таблица 3.7

Функция 108. Служебные команды				
байт	**описание**	**значение**	**порядок**	**размер**
Фрейм ЗАПРОСА:				
0	slave's adr	1-247		1 byte
1	function	108		1 byte
2	SubFunction	1 - DEVICE_START 2 - DEVICE_STOP 3 - DEVICE_RESET 4 - RS485_RESET 5 - SETPOINTS_APPLY 6 - SETTINGS_APPLY 7 - GET_DEVICE_NAME 8 - GET_DEVICE_ID 9 - UPDATE_FIRMWARE 10 - GET_FULL_ID 11 - GET_BOOT_SIZE		1 byte
3	CRC16		low	2 bytes
4			high	
Фрейм ОТВЕТА:				
0	slave's adr	1-247		1 byte
1	function	108		1 byte
2	SubFunction	1-11		1 byte
3	data (byte0)	(0-251) байт данных, если таковые имеются		1 byte
…	…			1 byte
252	data(byte249)			1 byte
253	data(byte250)			1 byte
254	CRC16		low	2 bytes
255			high	
Фрейм ОТВЕТА С КОДОМ ОШИБКИ:				
0	slave's adr	1-247		1 byte
1	function	108 + 0x80		1 byte
2	exception code	1	wrong function	1 byte
		2	wrong address	
		3	wrong bytes qty	
		4	read error	
3	CRC16		low	2 bytes
4			high	

Функция 108 «Служебные команды» имеет следующие коды подфункций (см. таб.3.7а).

Подфункции, возвращающие какие-либо данные, имеют префикс GET. Подфункции, не возвращающие данных, не содержат поля данных и, при удачном выполнении, возвращаются эхом.

Коды подфункций функции 108. Таблица 3.7а

№	Имя	Описание	Размер блока данных, байт
1	DEVICE_START	Запустить прибор	0
2	DEVICE_STOP	Остановить прибор	0
3	DEVICE_RESET	Перезагрузить прибор	0
4	RS485_RESET	Проинициализировать интерфейс	0
5	SETPOINTS_APPLY	Применить параметры настроек	0
6	SETTINGS_APPLY	Применить уставки	0
7	GET_DEVICE_NAME	Получить имя прибора (строка)	≤ 251
8	GET_DEVICE_ID	Получить идентификационный номер прибора	1
9	UPDATE_FIRMWARE	Перейти в режим обновления микропрограммы	0
10	GET_FULL_ID	Получить полный идентификационный номер прибора	4
11	GET_BOOT_SIZE	Получить размер секции загрузчика	2

Коды приборов подфункции GET_DEVICE_ID Таблица 3.7б

DEVICE_ID_PARAGRAPH	5
DEVICE_ID_VEHA_C	8
DEVICE_ID_VEHA_T	9
DEVICE_ID_OMIX	10
DEVICE_ID_DRMT2	11

Пользовательская функция 109. Таблица 3.8

Функция 109. Работа с файловой системой DFFS				
байт	**описание**	**значение**	**порядок**	**размер**
Фрейм ЗАПРОСА:				
0	slave's adr	1-247		1 byte
1	function	109		1 byte
2	SubFunction	1.GET_FIRST_VOLUME 2.GET_LAST_VOLUME 3.GET_VOL_BEGIN_PAGE 4.GET_VOL_END_PAGE		1 byte
3	CRC16		low	2 bytes
4			high	
Фрейм ОТВЕТА:				
0	slave's adr	1-247		1 byte
1	function	109		1 byte
2	SubFunction	1-4		1 byte
3	data (byte0)	(2 или 4) байта данных		1 byte
4	data (byte1)			1 byte
…				1 byte
6	data (byte3)			1 byte
7	CRC16		low	2 bytes
8			high	
Фрейм ОТВЕТА С КОДОМ ОШИБКИ:				
0	slave's adr	1-247		1 byte
1	function	109 + 0x80		1 byte
2	exception code	1	wrong function	1 byte
		2	wrong address	
		3	wrong bytes qty	
		4	read error	
3	CRC16		low	2 bytes
4			high	

Подфункции 1 и 2, возвращающие номер тома всегда возвращают 4-х байтное значение типа DWORD.

Подфункции 3 и 4, возвращающие номера страниц могут возвращать как 2-х байтные (WORD), так и 4-х байтные (DWORD) значения, в зависимости от модели прибора.

4. Карты распределения памяти приборов

В следующих таблицах представлены карты распределения памяти приборов. Следует отметить тот факт, что в стандартных MODBUS функциях размеры типов данных могут отличаться от типов данных пользовательских функций (в большую сторону) в случае, если размер типа данных некратен типу WORD (2 байта).

Массивы обозначены словом **array**, а количество элементов массив указано в квадратных скобках **[n]** поле имени переменной.

Используются следующие типы данных:

BIT – бит;

BYTE – байт размером в 8 бит;

WORD – слово без знака размером 2 байта;

DWORD – двойное слово без знака размером 4 байта;

SIGNED WORD – слово со знаком размером 2 байта;

float – число с плавающей точкой в формате IEE754 размером 4 байта.

```
typedef struct
{
   WORD  f;        // Value (4 significant digit).
   BYTE   n_z;     // Position of decimal point from right.
   BYTE   x;       // Multiplier x1 x10 x100.
} sfloat_t;         //  – структура размером 4 байта, а
                    // в стандартных функциях 4 слова.
typedef struct
{
   BYTE  sec;      // [0..59]
   BYTE  min;      // [0..59]
   BYTE  hour;     // [0..23]
   BYTE  hsec;     // [0..99]  sec/100 = hundredth of sec
   BYTE  day;      // [1..31]
   BYTE  month;    // [1..12]
   BYTE  year;     // [0..99],  0 = 2000
} rtc_t;            //  – структура размером 7 байт, а
                    // в стандартных функциях 7 слов.
```

Карта распределения памяти Веха-С. Таблица 4.1

Veha-C memory's map					
Address	**Data type**	**Name**	**Byte order**	**Size**	**Access**
Coils - std modbus functions 1 & 5 (bit addressing mode)					
0	array of BIT	relay[2]		2 BITS	r + w
Discrete inputs - std modbus function 2 (bit addressing mode)					
0	BIT	ext_key_stop		BIT	r
1	BIT	ext_key_reset		BIT	r
Holding registers - std modbus function 3 & 16 (word addressing mode)					
0	array of sfloat_t	ee_sp[2]	big endian	6 WORDS	r + w
6	array of sfloat_t	ee_relay_n_on[2]	big endian	6 WORDS	r + w
12	array of sfloat_t	ee_relay_t_on[2]	big endian	6 WORDS	r + w
18	BYTE	ee_cnt_logic	big endian	WORD	r + w
19	BYTE	ee_sensor_type	big endian	WORD	r + w
20	WORD	ee_divisor	big endian	WORD	r + w
21	sfloat_t	ee_multiplier	big endian	3 WORDS	r + w
24	sfloat_t	ee_init_value	big endian	3 WORDS	r + w
27	BYTE	ee_overflow_logic	big endian	WORD	r + w
28	BYTE	ee_power_off_logic	big endian	WORD	r + w
29	BYTE	ee_ext_stop_logic	big endian	WORD	r + w
30	array of BYTE	ee_relay_logic[2]	big endian	2 WORDS	r + w
32	BYTE	ee_pass_level	big endian	WORD	r + w
33	BYTE	ee_rs485_number	big endian	WORD	r + w
34	BYTE	ee_rs485_speed	big endian	WORD	r + w
35	BYTE	ee_rs485_byte_len	big endian	WORD	r + w
36	BYTE	ee_rs485_parity	big endian	WORD	r + w
37	BYTE	ee_rs485_stop_bits	big endian	WORD	r + w
38	float	ee_backup_counter	big endian	2 WORDS	r + w
40	BYTE	ee_led_br_level	big endian	WORD	r + w
41	sfloat_t	ee_dac_min	big endian	3 WORDS	r + w
44	sfloat_t	ee_dac_max	big endian	3 WORDS	r + w
47	BYTE	ee_dac_type	big endian	WORD	r + w
Input registers - std modbus function 4 (word addressing mode)					
0	float	cnt.result.floating	big endian	2 WORDS	r
2	DWORD	cnt.result.integer	big endian	2 WORDS	r
4	DWORD	cnt.result.wo_multiplier	big endian	2 WORDS	r
SRAM - user's function 104 (byte addressing mode)					
0	float	cnt.result.floating	little endian	4 BYTES	r
4	DWORD	cnt.result.integer	little endian	4 BYTES	r
8	DWORD	cnt.result.wo_multiplier	little endian	4 BYTES	r

EEPROM - user's function 100 & 101 (byte addressing mode)					
1	array of sfloat_t	ee_sp[2]	little endian	8 BYTES	r + w
9	array of sfloat_t	ee_relay_n_on[2]	little endian	8 BYTES	r + w
17	array of sfloat_t	ee_relay_t_on[2]	little endian	8 BYTES	r + w
25	BYTE	ee_cnt_logic	little endian	BYTE	r + w
26	BYTE	ee_sensor_type	little endian	BYTE	r + w
27	WORD	ee_divisor	little endian	2 BYTES	r + w
29	sfloat_t	ee_multiplier	little endian	4 BYTES	r + w
33	sfloat_t	ee_init_value	little endian	4 BYTES	r + w
37	BYTE	ee_overflow_logic	little endian	BYTE	r + w
38	BYTE	ee_power_off_logic	little endian	BYTE	r + w
39	BYTE	ee_ext_stop_logic	little endian	BYTE	r + w
40	array of BYTE	ee_relay_logic[2]	little endian	2 BYTES	r + w
42	BYTE	ee_pass_level	little endian	BYTE	r + w
43	BYTE	ee_rs485_number	little endian	BYTE	r + w
44	BYTE	ee_rs485_speed	little endian	BYTE	r + w
45	BYTE	ee_rs485_byte_len	little endian	BYTE	r + w
46	BYTE	ee_rs485_parity	little endian	BYTE	r + w
47	BYTE	ee_rs485_stop_bits	little endian	BYTE	r + w
48	float	ee_backup_counter	little endian	4 BYTES	r + w
52	BYTE	ee_led_br_level	little endian	BYTE	r + w
53	BYTE	ee_dac_type	little endian	BYTE	r + w
54	sfloat_t	ee_dac_min	little endian	4 BYTES	r + w
58	sfloat_t	ee_dac_max	little endian	4 BYTES	r + w

Карта распределения памяти Веха-Т. Таблица 4.2

Veha-T memory's map					
Address	Data type	Name	Byte order	Size	Access
Coils - std modbus functions 1 & 5 (bit addressing mode)					
0	array of BIT	relay[2]		2 BITS	r + w
Discrete inputs - std modbus function 2 (bit addressing mode)					
0	BIT	ext_key_start		BIT	r
1	BIT	ext_key_div1_div2		BIT	r
Holding registers - std modbus function 3 & 16 (word addressing mode)					
0	array of sfloat_t	ee_sp[2]	big endian	6 WORDS	r + w
6	array of sfloat_t	ee_dsp[2]	big endian	6 WORDS	r + w
12	BYTE	ee_indicator_format	big endian	WORD	r + w
13	WORD	ee_f_input_max	big endian	WORD	r + w
14	WORD	ee_waiting_time	big endian	WORD	r + w
15	WORD	ee_divisor1	big endian	WORD	r + w

16	WORD	ee_divisor2	big endian	WORD	r + w
17	sfloat_t	ee_multiplier	big endian	3 WORDS	r + w
20	WORD	ee_start_delay	big endian	WORD	r + w
21	BYTE	ee_start_logic	big endian	WORD	r + w
22	array of BYTE	ee_relay_logic[2]	big endian	2 WORDS	r + w
24	BYTE	ee_pass_level	big endian	WORD	r + w
25	sfloat_t	ee_dac_speed_min	big endian	3 WORDS	r + w
28	sfloat_t	ee_dac_speed_max	big endian	3 WORDS	r + w
31	BYTE	ee_dac_type	big endian	WORD	r + w
32	BYTE	ee_rs485_number	big endian	WORD	r + w
33	BYTE	ee_rs485_speed	big endian	WORD	r + w
34	BYTE	ee_rs485_byte_len	big endian	WORD	r + w
35	BYTE	ee_rs485_parity	big endian	WORD	r + w
36	BYTE	ee_rs485_stop_bits	big endian	WORD	r + w
37	BYTE	ee_show_mode	big endian	WORD	r + w
38	DWORD	ee_uptime	big endian	2 WORDS	r + w
40	BYTE	ee_led_br_level	big endian	WORD	r + w
41	array of sfloat_t	ee_lo_alarm[2]	big endian	6 WORDS	r + w
47	array of sfloat_t	ee_hi_alarm[2]	big endian	6 WORDS	r + w
Input registers - std modbus function 4 (word addressing mode)					
0	float	tacho.result.floating_abs	big endian	2 WORDS	r
2	float	tacho.result.floating	big endian	2 WORDS	r
4	DWORD	tacho.result.integer_abs	big endian	2 WORDS	r
6	SIGNED DWORD	tacho.result.integer	big endian	2 WORDS	r
8	BYTE	tacho.result.dir	big endian	WORD	r
9	DWORD	tacho.result.uptime_sec	big endian	2 WORDS	r
11	DWORD	tacho.result.uptime_hour	big endian	2 WORDS	r
SRAM - user's function 104 (byte addressing mode)					
0	float	tacho.result.floating_abs	little endian	4 BYTES	r
4	float	tacho.result.floating	little endian	4 BYTES	r
8	DWORD	tacho.result.integer_abs	little endian	4 BYTES	r
12	SIGNED DWORD	tacho.result.integer	little endian	4 BYTES	r
16	BYTE	tacho.result.dir	little endian	BYTE	r
17	DWORD	tacho.result.uptime_sec	little endian	4 BYTES	r
21	DWORD	tacho.result.uptime_hour	little endian	4 BYTES	r
EEPROM - user's function 100 & 101 (byte addressing mode)					
1	array of sfloat_t	ee_sp[2]	little endian	8 BYTES	r + w
9	array of sfloat_t	ee_dsp[2]	little endian	8 BYTES	r + w
17	BYTE	ee_indicator_format	little endian	BYTE	r + w

18	WORD	ee_f_input_max	little endian	2 BYTES	r + w
20	WORD	ee_waiting_time	little endian	2 BYTES	r + w
22	WORD	ee_divisor1	little endian	2 BYTES	r + w
24	sfloat_t	ee_multiplier	little endian	4 BYTES	r + w
28	WORD	ee_start_delay	little endian	2 BYTES	r + w
30	BYTE	ee_start_logic	little endian	BYTE	r + w
31	array of BYTE	ee_relay_logic[2]	little endian	2 BYTES	r + w
33	BYTE	ee_pass_level	little endian	BYTE	r + w
34	sfloat_t	ee_dac_speed_min	little endian	4 BYTES	r + w
38	sfloat_t	ee_dac_speed_max	little endian	4 BYTES	r + w
42	BYTE	ee_rs485_number	little endian	BYTE	r + w
43	BYTE	ee_rs485_speed	little endian	BYTE	r + w
44	BYTE	ee_rs485_byte_len	little endian	BYTE	r + w
45	BYTE	ee_rs485_parity	little endian	BYTE	r + w
46	BYTE	ee_rs485_stop_bits	little endian	BYTE	r + w
47	BYTE	ee_show_mode	little endian	BYTE	r + w
48	DWORD	ee_uptime	little endian	4 BYTES	r + w
52	BYTE	ee_led_br_level	little endian	BYTE	r + w
53	array of sfloat_t	ee_lo_alarm[2]	little endian	8 BYTES	r + w
61	array of sfloat_t	ee_hi_alarm[2]	little endian	8 BYTES	r + w
69	WORD	ee_divisor2	little endian	2 BYTES	r + w
71	BYTE	ee_dac_type	little endian	BYTE	r + w

Карта распределения памяти Параграф. Таблица 4.3

Paragraph memory's map					
Address	Data type	Name	Byte order	Size	Access
Coils - std modbus functions 1 & 5 (bit addressing mode)					
0	array of BIT	relay[4]		4 BITS	r + w
Discrete inputs - std modbus function 2 (bit addressing mode)					
0	BIT	ext_key_start		BIT	r
1	BIT	ext_key_stop		BIT	r
Holding registers - std modbus function 3 & 16 (word addressing mode)					
0	array of float	ee_sp[4]	big endian	8 WORDS	r + w
8	array of float	ee_dsp[4]	big endian	8 WORDS	r + w
16	array of float	ee_amplifier[3]	big endian	6 WORDS	r + w
22	array of float	ee_offset[3]	big endian	6 WORDS	r + w
28	array of float	ee_analog_in_min[2]	big endian	4 WORDS	r + w
32	array of float	ee_analog_in_max[2]	big endian	4 WORDS	r + w
36	array of float	ee_dac_min[2]	big endian	4 WORDS	r + w
40	array of float	ee_dac_max[2]	big endian	4 WORDS	r + w
44	array of BYTE	ee_relay_owner[4]	big endian	4 WORDS	r + w

48	array of BYTE	ee_sensor_type[2]	big endian	2 WORDS	r + w
50	array of BYTE	ee_math_processing_type[2]	big endian	2 WORDS	r + w
52	array of BYTE	ee_compensation_state[2]	big endian	2 WORDS	r + w
54	array of BYTE	ee_relay_logic[4]	big endian	4 WORDS	r + w
58	array of BYTE	ee_dac_owner[2]	big endian	2 WORDS	r + w
60	array of BYTE	ee_dac_type[2]	big endian	2 WORDS	r + w
62	BYTE	ee_rs485_number	big endian	WORD	r + w
63	BYTE	ee_rs485_speed	big endian	WORD	r + w
64	BYTE	ee_rs485_byte_len	big endian	WORD	r + w
65	BYTE	ee_rs485_parity	big endian	WORD	r + w
66	BYTE	ee_rs485_stop_bits	big endian	WORD	r + w
67	BYTE	ee_plot_x_scale	big endian	WORD	r + w
68	BYTE	ee_dffs_rec_interval	big endian	WORD	r + w
69	BYTE	ee_access	big endian	WORD	r + w
70	BYTE	ee_dffs_rec_mode	big endian	WORD	r + w
71	BYTE	ee_work_ctl	big endian	WORD	r + w
72	BYTE	ee_work_rec	big endian	WORD	r + w
73	BYTE	ee_ext_keys_logic	big endian	WORD	r + w
74	array of BIT	ee_plot_visible_series[2]	big endian	WORD	r + w
75	array of BYTE	ee_median_filter_rank[3]	big endian	3 WORDS	r + w
78	array of float	ee_integral[2]	big endian	4 WORDS	r + w
82	rtc_t	ee_rtc_edit	big endian	7 WORDS	r + w
89	array of BYTE	ee_led_n_f[2]	big endian	2 WORDS	r + w
91	BYTE	ee_language	big endian	WORD	r + w
92	BYTE	ee_plot_y_autoscale	big endian	WORD	r + w
93	float	ee_plot_y_min	big endian	2 WORDS	r + w
95	float	ee_plot_y_max	big endian	2 WORDS	r + w
97	BYTE	ee_plot_y_min_n_f	big endian	WORD	r + w
98	BYTE	ee_plot_y_max_n_f	big endian	WORD	r + w
99	float	ee_pid_t	big endian	2 WORDS	r + w
101	array of float	ee_pid_xp[4]	big endian	8 WORDS	r + w
109	array of float	ee_pid_ti[4]	big endian	8 WORDS	r + w
117	array of float	ee_pid_td[4]	big endian	8 WORDS	r + w
125	array of float	ee_pid_min_power[4]	big endian	8 WORDS	r + w
133	array of float	ee_pid_max_power[4]	big endian	8 WORDS	r + w
141	array of float	ee_pid_alarm_power[4]	big endian	8 WORDS	r + w
149	array of BYTE	ee_sensor_validate[2]	big endian	2 WORDS	r + w

151	array of BYTE	ee_relay_on_alarm_state[4]	big endian	4 WORDS	r + w
155	BYTE	ee_rtc_summer_time_mode_edit	big endian	WORD	r + w
156	WORD	ee_rtc_daily_time_correction	big endian	WORD	r + w
Input registers - std modbus function 4 (word addressing mode)					
0	array of float	analog_inputs[3]	big endian	6 WORDS	r
6	array of WORD	dac_pwm[2]	big endian	2 WORDS	r
8	array of BYTE	relays_state[4]	big endian	4 WORDS	r
12	rtc_t	rtc	big endian	7 WORDS	r
SRAM - user's function 104 (byte addressing mode)					
0	array of float	analog_inputs[3]	little endian	12 BYTES	r
12	array of WORD	dac_pwm[2]	little endian	4 BYTES	r
16	array of BYTE	relays_state[4]	little endian	4 BYTES	r
20	rtc_t	rtc	little endian	7 BYTES	r
EEPROM - user's function 100 & 101 (byte addressing mode)					
1	array of float	ee_sp[4]	little endian	16 BYTES	r + w
17	array of float	ee_dsp[4]	little endian	16 BYTES	r + w
33	array of float	ee_amplifier[3]	little endian	12 BYTES	r + w
45	array of float	ee_offset[3]	little endian	12 BYTES	r + w
57	array of float	ee_analog_in_min[2]	little endian	8 BYTES	r + w
65	array of float	ee_analog_in_max[2]	little endian	8 BYTES	r + w
73	array of float	ee_dac_min[2]	little endian	8 BYTES	r + w
81	array of float	ee_dac_max[2]	little endian	8 BYTES	r + w
101	array of BYTE	ee_relay_owner[4]	little endian	4 BYTES	r + w
105	array of BYTE	ee_sensor_type[2]	little endian	2 BYTES	r + w
107	array of BYTE	ee_math_processing_type[2]	little endian	2 BYTES	r + w
109	array of BYTE	ee_compensation_state[2]	little endian	2 BYTES	r + w
111	array of BYTE	ee_relay_logic[4]	little endian	4 BYTES	r + w
115	array of BYTE	ee_dac_owner[2]	little endian	2 BYTES	r + w
117	array of BYTE	ee_dac_type[2]	little endian	2 BYTES	r + w
119	BYTE	ee_rs485_number	little endian	BYTE	r + w
120	BYTE	ee_rs485_speed	little endian	BYTE	r + w
121	BYTE	ee_rs485_byte_len	little endian	BYTE	r + w
122	BYTE	ee_rs485_parity	little endian	BYTE	r + w
123	BYTE	ee_rs485_stop_bits	little endian	BYTE	r + w
124	BYTE	ee_plot_x_scale	little endian	BYTE	r + w
125	BYTE	ee_dffs_rec_interval	little endian	BYTE	r + w

126	BYTE	ee_access	little endian	BYTE	r + w
127	BYTE	ee_dffs_rec_mode	little endian	BYTE	r + w
128	BYTE	ee_work_ctl	little endian	BYTE	r + w
129	BYTE	ee_work_rec	little endian	BYTE	r + w
130	BYTE	ee_ext_keys_logic	little endian	BYTE	r + w
131	array of BIT	ee_plot_visible_series[2]	little endian	BYTE	r + w
132	array of BYTE	ee_median_filter_rank[3]	little endian	3 BYTES	r + w
135	array of float	ee_integral[2]	little endian	8 BYTES	r + w
144	rtc_t	ee_rtc_edit	little endian	7 BYTES	r + w
152	array of BYTE	ee_led_n_f[2]	little endian	2 BYTES	r + w
170	BYTE	ee_language	little endian	BYTE	r + w
171	BYTE	ee_plot_y_autoscale	little endian	BYTE	r + w
172	float	ee_plot_y_min	little endian	4 BYTES	r + w
176	float	ee_plot_y_max	little endian	4 BYTES	r + w
180	BYTE	ee_plot_y_min_n_f	little endian	BYTE	r + w
181	BYTE	ee_plot_y_max_n_f	little endian	BYTE	r + w
182	float	ee_pid_t	little endian	4 BYTES	r + w
186	array of float	ee_pid_xp[4]	little endian	16 BYTES	r + w
202	array of float	ee_pid_ti[4]	little endian	16 BYTES	r + w
218	array of float	ee_pid_td[4]	little endian	16 BYTES	r + w
234	array of float	ee_pid_min_power[4]	little endian	16 BYTES	r + w
250	array of float	ee_pid_max_power[4]	little endian	16 BYTES	r + w
266	array of float	ee_pid_alarm_power[4]	little endian	16 BYTES	r + w
282	array of BYTE	ee_sensor_validate[2]	little endian	2 BYTES	r + w
284	array of BYTE	ee_relay_on_alarm_state[4]	little endian	4 BYTES	r + w
296	BYTE	ee_rtc_summer_time_mode_edit	little endian	BYTE	r + w
297	WORD	ee_rtc_daily_time_correction	little endian	2 BYTES	r + w

5. Обратная связь

Со всеми вопросами, предложениями обращайтесь по адресу электронной почты support@automatix.ru или по телефонам:

(812) 327-32-74 - многоканальный, (812) 928-32-74.

Почтовый адрес: 191104, Санкт-Петербург, а.я. 59.

Офис, выставка: г. Санкт-Петербург, ул. Политехническая,
д. 29, СПбГПУ (Политехнический Университет), Гидротехнический корпус 1, аудитория 246.

Дополнительная информация на нашем интерне-сайте www.automatix.ru

6. Использованные источники информации

1. Electrical Characteristics of Balanced Voltage Digital Interface Circuits, ANSI/TIA/EIA-422-B-1994, Telecommunications Industry Association, 1994
2. Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems, ANSI/TIA/EIA-485-A-1998, Telecommunications Industry Association, 1998
3. Application Guidelines for TIA/EIA-485-A, TIA/EIA Telecommunications Systems Bulletin, Telecommunications Industry Association, 1998
4. A Comparison of Differential Termination Techniques, Joe Vo, National Semiconductor, Application Note AN-903
5. Data Transmission Design Seminar Reference Manual, 1998, Texas Instruments, literature number SLLE01
6. Data Transmission Line Circuits Data Book, 1998, Texas Instruments, literature number SLLD001
7. MODBUS Application Protocol Specification
8. MODBUS over serial line specification and implementation guide