



Электропривод

<https://electroprivod.ru>

Блок управления шаговым двигателем

Модели SMSD-4.2LAN и SMSD-8.0LAN

Протокол обмена данными

Ver. 05

2020

1. Основные сведения.	4
2. Принцип передачи данных по Ethernet и USB.	4
3. Заводские настройки	4
4. Структура информационного пакета передачи данных	5
4.1. Назначение области XOR_SUM	5
4.2. Назначение области VER	5
4.3. Назначение области CMD_IDENTIFICATION	6
4.4. Назначение области LENGTH_DATA	6
4.5. Назначение области DATA[LENGTH_DATA]	6
4.6. Назначение области CMD_TYPE	6
4.6.1 Начало сессии обмена данными с контроллером. Команда передачи данных CODE_CMD_REQUEST	7
4.6.2 Команда передачи данных CODE_CMD_RESPONSE	8
4.6.3 Команда передачи данных CODE_CMD_POWERSTEP01	9
4.6.4 Команды передачи данных CODE_CMD_POWERSTEP01_W_MEM0..MEM3	10
4.6.5 Команды передачи данных CODE_CMD_POWERSTEP01_R_MEM0..MEM3	11
4.6.6 Команда передачи данных CODE_CMD_CONFIG_SET	12
4.6.7 Команда передачи данных CODE_CMD_CONFIG_GET	12
4.6.8 Команда передачи данных CODE_CMD_PASSWORD_SET	13
4.6.9 Команда передачи данных CODE_CMD_ERROR_GET	13
5. Структура COMMANDS_RETURN_DATA_Type	14
5.1 Назначение битовых полей STATUS_POWERSTEP01	15
5.2 Список возможных значений поля ERROR_OR_COMMAND	15
6. Структура исполнительных команды управления SMSD_CMD_Type	16
6.1 Исполнительная команда CMD_PowerSTEP01_END	17
6.2 Исполнительная команда CMD_PowerSTEP01_GET_SPEED	18
6.3 Исполнительная команда CMD_PowerSTEP01_STATUS_IN_EVENT	19
6.4 Исполнительная команда CMD_PowerSTEP01_SET_MODE	19
6.5 Исполнительная команда CMD_PowerSTEP01_GET_MODE	22
6.6 Исполнительная команда CMD_PowerSTEP01_SET_MIN_SPEED	23
6.7 Исполнительная команда CMD_PowerSTEP01_SET_MAX_SPEED	23
6.8 Исполнительная команда CMD_PowerSTEP01_SET_ACC	24
6.9 Исполнительная команда CMD_PowerSTEP01_SET_DEC	24
6.10 Исполнительная команда CMD_PowerSTEP01_SET_FS_SPEED	24
6.11 Исполнительная команда CMD_PowerSTEP01_SET_MASK_EVENT	25
6.12 Исполнительная команда CMD_PowerSTEP01_GET_ABS_POS	25
6.13 Исполнительная команда CMD_PowerSTEP01_GET_EL_POS	25
6.14 Исполнительная команда CMD_PowerSTEP01_GET_STATUS_AND_CLR	26
6.15 Исполнительная команда CMD_PowerSTEP01_RUN_F	26
6.16 Исполнительная команда CMD_PowerSTEP01_RUN_R	27
6.17 Исполнительная команда CMD_PowerSTEP01_MOVE_F	27
6.18 Исполнительная команда CMD_PowerSTEP01_MOVE_R	27
6.19 Исполнительная команда CMD_PowerSTEP01_GO_TO_F	28
6.20 Исполнительная команда CMD_PowerSTEP01_GO_TO_R	28
6.21 Исполнительная команда CMD_PowerSTEP01_GO_UNTIL_F	29
6.22 Исполнительная команда CMD_PowerSTEP01_GO_UNTIL_R	29



6.23	Исполнительная команда	CMD_PowerSTEP01_SCAN_ZERO_F	29
6.24	Исполнительная команда	CMD_PowerSTEP01_SCAN_ZERO_R	30
6.25	Исполнительная команда	CMD_PowerSTEP01_SCAN_LABEL_F	30
6.26	Исполнительная команда	CMD_PowerSTEP01_SCAN_LABEL_R	30
6.27	Исполнительная команда	CMD_PowerSTEP01_GO_ZERO	31
6.28	Исполнительная команда	CMD_PowerSTEP01_GO_LABEL	31
6.29	Исполнительная команда	CMD_PowerSTEP01_GO_TO	31
6.30	Исполнительная команда	CMD_PowerSTEP01_RESET_POS	32
6.31	Исполнительная команда	CMD_PowerSTEP01_RESET_POWERSTEP01	32
6.32	Исполнительная команда	CMD_PowerSTEP01_SOFT_STOP	32
6.33	Исполнительная команда	CMD_PowerSTEP01_HARD_STOP	33
6.34	Исполнительная команда	CMD_PowerSTEP01_SOFT_HI_Z	33
6.35	Исполнительная команда	CMD_PowerSTEP01_HARD_HI_Z	33
6.36	Исполнительная команда	CMD_PowerSTEP01_SET_WAIT	34
6.37	Исполнительная команда	CMD_PowerSTEP01_SET_RELE	34
6.38	Исполнительная команда	CMD_PowerSTEP01_CLR_RELE	34
6.39	Исполнительная команда	CMD_PowerSTEP01_GET_RELE	35
6.40	Исполнительная команда	CMD_PowerSTEP01_WAIT_IN0	35
6.41	Исполнительная команда	CMD_PowerSTEP01_WAIT_IN1	35
6.42	Исполнительная команда	CMD_PowerSTEP01_GOTO_PROGRAM	36
6.43	Исполнительная команда	CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN0	36
6.44	Исполнительная команда	CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN1	37
6.45	Исполнительная команда	CMD_PowerSTEP01_LOOP_PROGRAM	37
6.46	Исполнительная команда	CMD_PowerSTEP01_CALL_PROGRAM	38
6.47	Исполнительная команда	CMD_PowerSTEP01_RETURN_PROGRAM	38
6.48	Исполнительная команда	CMD_PowerSTEP01_START_PROGRAM_MEM0	39
6.49	Исполнительная команда	CMD_PowerSTEP01_STOP_PROGRAM_MEM	39
6.50	Исполнительная команда	CMD_PowerSTEP01_STEP_CLOCK	39
6.51	Исполнительная команда	CMD_PowerSTEP01_STOP_USB	40
6.52	Исполнительная команда	CMD_PowerSTEP01_GET_MIN_SPEED	40
6.53	Исполнительная команда	CMD_PowerSTEP01_GET_MAX_SPEED	40
6.54	Исполнительная команда	CMD_PowerSTEP01_GET_STACK	41
6.55	Исполнительная команда	CMD_PowerSTEP01_GOTO_PROGRAM_IF_ZERO	41
6.56	Исполнительная команда	CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN_ZERO	42
6.57	Исполнительная команда	CMD_PowerSTEP01_WAIT_CONTINUE	42
6.58	Исполнительная команда	CMD_PowerSTEP01_SET_WAIT_2	43
6.59	Исполнительная команда	CMD_PowerSTEP01_SCAN_MARK2_F	43
6.60	Исполнительная команда	CMD_PowerSTEP01_SCAN_MARK2_R	44
7. Структура SMSD_LAN_Config_Type			44
8. Отличия между Ethernet и USB в передаче потока данных			45



1. Основные сведения.

Блок управления SMSD_LAN, далее по тексту - Контроллер, предназначен для управления шаговыми двигателями. Контроллер представляет собой электронное устройство, состоящее из корпуса, электронной платы, разъемов, передней панели на которой находятся органы управления и индикации.

В режиме работы по локальной сети Ethernet (на индикаторе «LA»), Контроллер создаёт сокет для подключения к нему управляющей пользовательской программы или устройства, далее по тексту - Пользователь. Данные передаются по физической линии Ethernet (протокол TCP).

Для управления и настройки, вместо сети Ethernet, Контроллер можно подключить через интерфейс USB (имитация COM-port). При этом система команд аналогичная, за исключением незначительных отличий в передаче потока данных на прикладном уровне, данные отличия описаны ниже.

Данное руководство относится к стандартной версии прошивки контроллеров. Пожалуйста, обратитесь к руководству «Протокол обмена – версия Modbus TCP» для работы с контроллерами с прошивкой Modbus.

2. Принцип передачи данных по Ethernet и USB.

Передачу данных необходимо осуществлять законченными информационными пакетами, содержащими только одну команду управления типа CMD_TYPE.

Не допускается передача одновременно нескольких команд управления подряд в одном пакете. После приёма команды Контроллер производит необходимые действия и отправляет, ответ содержащий статус результата работы или данные. Ответ производится по тому же физическому каналу, по которому поступила управляющая команда. Последовательность байт в структурах пакетов – обратный, «от младшего к старшему» (Intel).

3. Заводские настройки

Параметры подключения по сети Ethernet, установленные в контроллере по умолчанию:

- MAC адрес: 0x00 0xf8 0xdc 0x3f 0x00 0x00
- IP адрес: 192.168.1.2
- Порт: 5000
- Маска подсети: 255.255.0.0
- Основной шлюз: 192.168.1.1

Данные параметры в дальнейшем можно поменять как по сети Ethernet, так и через USB интерфейс.

Параметры передачи данных RS-232 (подключение USB):

- Скорость: 115200
- Бит данных: 8
- Проверка четности: нет
- Стоп биты: 1



4. Структура информационного пакета передачи данных

Структура информационного пакета передачи данных:

```
typedef struct
{
    uint8_t  XOR_SUM;
    uint8_t  VER;
    uint8_t  CMD_TYPE;
    uint8_t  CMD_IDENTIFICATION;
    uint16_t LENGTH_DATA;
    uint8_t  DATA[LENGTH_DATA];
} LAN_COMMAND_Type;
```

XOR_SUM – контрольная сумма – младший байт от суммы всех байт.

VER – версия протокола.

CMD_TYPE – тип команды, передаваемой по сети.

CMD_IDENTIFICATION – уникальный идентификатор, будет передан в ответном сообщении от Контроллера на данную команду, что позволяет однозначно сопоставить переданную команду и полученный ответ.

LENGTH_DATA – длина информационной части пакета, значения от 0 до 1024.

DATA[LENGTH_DATA] – информационная часть пакета длиной LENGTH_DATA байт.

4.1. Назначение области XOR_SUM

Протокол TCP подразумевает под собой механизм гарантированной доставки сообщения получателю и включает в себя проверку и исправление возможных ошибок. Тем не менее, в команде управления предусмотрено поле XOR_SUM – контрольная сумма команды запроса/ответа, предназначенная для проверки на целостность пакета в случае его передачи по USB. Алгоритм вычисления контрольной суммы XOR_SUM:

```
COMMAND.XOR_SUM=0x00;
COMMAND.XOR_SUM=xor_sum((uint8_t*)&COMMAND.XOR_SUM,
sizeof(COMMAND));

uint8_t xor_sum(uint8_t *data,uint16_t length)
{
    uint8_t xor_temp=0xFF;
    while(length--){xor_temp+=*data;data++;}
    return (xor_temp^0xFF);
}

Where:
(uint8_t*)& COMMAND.XOR_SUM– начало передаваемого пакета,
sizeof(COMMAND) – длина передаваемого пакета (в байтах).
```

4.2. Назначение области VER

Поле данных VER длиной 1 байт - текущая версия коммуникационного протокола.



4.3. Назначение области CMD_IDENTIFICATION

Поле CMD_IDENTIFICATION длиной 1 байт предназначено для однозначной идентификации ответа на отправленную команду. Пользователь должен обеспечить уникальность значения в рамках текущего обмена информацией хотя бы в пределах нескольких команд.

4.4. Назначение области LENGTH_DATA

Поле LENGTH_DATA длиной 2 байта определяет длину информационной части пакета - от 0 до 1024.

4.5. Назначение области DATA[LENGTH_DATA]

Поле DATA[LENGTH_DATA] – массив DATA длиной LENGTH_DATA байт - является информационной частью пакета. Структура и размер поля зависят от типа передаваемой команды CMD_TYPE.

4.6. Назначение области CMD_TYPE

Поле данных CMD_TYPE длиной 1 байт - команда, передаваемая по сети. Числовые значения начинаются с 0 и последовательно инкрементируются. Список возможных вариантов значений поля CMD_TYPE:

0x00 - **CODE_CMD_REQUEST** – команда авторизации (поле DATA пакета содержит информацию для авторизации)

0x01 - **CODE_CMD_RESPONSE** – команда подтверждения (поле DATA пакета зависит от отправленной контроллеру команды)

0x02 - **CODE_CMD_POWERSTEP01** – команда управления в реальном масштабе времени (поле DATA пакета содержит команды POWERSTEP01 типа SMSD_CMD_Type).

0x03 - **CODE_CMD_POWERSTEP01_W_MEM0** – команда записи программы управления в банк памяти 0.

0x04 - **CODE_CMD_POWERSTEP01_W_MEM1** – команда записи программы управления в банк памяти 1

0x05 - **CODE_CMD_POWERSTEP01_W_MEM2** – команда записи программы управления в банк памяти 2

0x06 - **CODE_CMD_POWERSTEP01_W_MEM3** – команда записи программы управления в банк памяти 3

0x07 - **CODE_CMD_POWERSTEP01_R_MEM0** – команда чтения программы управления из банка памяти 0

0x08 - **CODE_CMD_POWERSTEP01_R_MEM1** – команда чтения программы управления из банка памяти 1

0x09 - **CODE_CMD_POWERSTEP01_R_MEM2** – команда чтения программы управления из банка памяти 2

0x0A - **CODE_CMD_POWERSTEP01_R_MEM3** – команда чтения программы управления из банка памяти 3

0x0B - **CODE_CMD_CONFIG_SET** – команда записи настроек LAN

0x0C - **CODE_CMD_CONFIG_GET** - команда чтения настроек LAN

0x0D - **CODE_CMD_PASSWORD_SET** – изменения пароля для авторизации

0x0E - **CODE_CMD_ERROR_GET** - чтения количества включений рабочего режима Контроллера и статистики по ошибкам.

4.6.1 Начало сессии обмена данными с контроллером. Команда передачи данных CODE_CMD_REQUEST

Команда CODE_CMD_REQUEST используется для авторизации пользователя. Пакет данных с командой CODE_CMD_REQUEST отправляется контроллером пользователю как ответ на факт подключения контроллера (только в случае подключения по сети Ethernet, не используется при подключении USB).

Порядок обмена пакетами в процессе авторизации:

1. Из контроллера (только при подключении по Ethernet):

Поле	Значение	Порядок данных в пакете
XOR (1 байт)	x	0
VER (1 байт)	x	1
CMD_TYPE (1 байт)	CODE_CMD_REQUEST= 0x00	2
CMD_IDENTIFICATION (1 байт)	x	3
LENGTH_DATA (младший байт)	0x00	4
LENGTH_DATA (старший байт)	0x00	5
DATA	-	-

После получения пакета с кодом команды CODE_CMD_REQUEST, пользователь должен отправить пакет с кодом команды CODE_CMD_REQUEST, поле DATA должно содержать пароль для авторизации. Поле VER (версия протокола) в этой команде контроллером не проверяется. Заводское значение пароля: x01 0x23 0x45 0x67 0x89 0xAB 0xCD 0xEF. Это значение можно изменить при помощи команды CODE_CMD_PASSWORD_SET.

2. В контроллер:

Поле	Значение	Порядок данных в пакете
XOR (1 байт)	x	0
VER (1 байт)	x	1
CMD_TYPE (1 байт)	CODE_CMD_REQUEST = 0x00	2
CMD_IDENTIFICATION (1 байт)	x	3
LENGTH_DATA (младший байт)	0x08	4
LENGTH_DATA (старший байт)	0x00	5
DATA [0] (пароль – младший байт)	x	6
DATA [1]	x	7
DATA [2]	x	8
DATA [3]	x	9
DATA [4]	x	10
DATA [5]	x	11
DATA [6]	x	12
DATA [7] (пароль – старший байт)	x	13

Контроллер проверяет полученные данные и отправляет ответ, содержащий информацию о результате авторизации. Тип команды CMD_TYPE - CODE_CMD_RESPONSE, поле DATA ответа содержит структуру COMMANDS_RETURN_DATA. Описание структуры COMMANDS_RETURN_DATA приведено далее в данном руководстве.

3. Из контроллера:

Поле	Значение	Порядок данных в пакете
XOR (1 байт)	x	0
VER (1 байт)	x	1
CMD_TYPE (1 байт)	CODE_CMD_RESPONSE = 0x01	2
CMD_IDENTIFICATION (1 байт)	x	3
LENGTH_DATA (младший байт)	sizeof(COMMANDS_RETURN_DATA_Type)	0x07
LENGTH_DATA (старший байт)		0x00
DATA [0] (COMMANDS_RETURN_DATA – младший байт)	x	6
DATA [1]	x	7
DATA [2] = ERROR_OR_COMMAND	x	8
DATA [3]	0	9
DATA [4]	0	10
DATA [5]	0	11
DATA [6] (COMMANDS_RETURN_DATA старший байт)	0	12

В случае получения правильного пароля контроллер открывает доступ к управлению, а поле ERROR_OR_COMMAND структуры COMMANDS_RETURN_DATA содержит ответ OK_ACCESS. В случае неверного пароля ERROR_OR_COMMAND = ERROR_ACCESS, и контроллер закрывает соединение. Следующее соединение и попытка авторизации возможны не ранее, чем через 1 секунду. В случае, если попытка подключения и авторизации происходят ранее установленного таймута 1 с, контроллер отправляет пакет с кодом CODE_CMD_RESPONSE, поле ERROR_OR_COMMAND = ERROR_ACCESS_TIMEOUT независимо от правильности пароля. Таймат 1 с предотвращает подбор пароля автоматическими сервисами.

4.6.2 Команда передачи данных CODE_CMD_RESPONSE

Пакет данных с командой CODE_CMD_RESPONSE отправляется контроллером в ответ на некоторые команды пользователя (CODE_CMD_POWERSTEP01, CODE_CMD_CONFIG_SET, CODE_CMD_ID_SET, CODE_CMD_POWERSTEP01_W_MEM), а так же, в случае возникновения разного рода ошибок. Поле DATA пакета содержит структуру COMMANDS_RETURN_DATA (описание структуры приведено далее в данном руководстве).

Из контроллера:

Поле	Значение	Порядок данных в пакете
XOR (1 байт)	x	0
VER (1 байт)	x	1
CMD_TYPE (1 байт)	CODE_CMD_RESPONSE = 0x01	2
CMD_IDENTIFICATION (1 байт)	x	3
LENGTH_DATA (Младший байт)	Sizeof(COMMANDS_RETURN_DATA_Type)	0x07
LENGTH_DATA (Старший байт)		0x00
DATA [0] (COMMANDS_RETURN_DATA Младший байт)	x	6
DATA [1]	x	7
DATA [2] = ERROR_OR_COMMAND	x	8

DATA [3]	x	9
DATA [4]	x	10
DATA [5]	x	11
DATA [6] (COMMANDS_RETURN_DATA Старший байт)	x	12

4.6.3 Команда передачи данных CODE_CMD_POWERSTEP01

Команда передачи данных CODE_CMD_POWERSTEP01 используется для управления приводом в режиме реального времени (каждая отправленная команда сразу обрабатывается контроллером). Поле DATA пакета содержит структуру SMSD_CMD_Type, содержащую команду управления. Описание структуры SMSD_CMD_Type и команд управления приведены далее в этом руководстве.

В контроллер:

Поле	Значение		Порядок данных в пакете
XOR (1 байт)	x		0
VER (1 байт)	x		1
CMD_TYPE (1 байт)	CODE_CMD_POWERSTEP01 = 0x02		2
CMD_IDENTIFICATION (1 байт)	x		3
LENGTH_DATA (Младший байт)	sizeof(SMSD_CMD_Type)=0x04	0x04	4
LENGTH_DATA (Старший байт)		0x00	5
DATA [0] (SMSD_CMD_Type Младший байт)	x		6
DATA [1]	x		7
DATA [2]	x		8
DATA [3] (SMSD_CMD_Type Старший байт)	x		9

В ответ контроллер отправляет пакет с командой CMD_TYPE = CODE_CMD_POWERSTEP01, поле DATA содержит структуру COMMANDS_RETURN_DATA.

Из контроллера:

Поле	Значение		Порядок данных в пакете
XOR (1 байт)	x		0
VER (1 байт)	x		1
CMD_TYPE (1 байт)	CODE_CMD_POWERSTEP01 = 0x02		2
CMD_IDENTIFICATION (1 байт)	x		3
LENGTH_DATA (Младший байт)	sizeof(COMMANDS_RETURN_DATA_Type)	0x07	4
LENGTH_DATA (Старший байт)		0x00	5
DATA [0] Младший байт (COMMANDS_RETURN_DATA)	x		6
DATA [1]	x		7
DATA [2] = ERROR_OR_COMMAND	x		8
DATA [3]	x		9
DATA [4]	x		10
DATA [5]	x		11
DATA [6] - Старший байт (COMMANDS_RETURN_DATA)	x		12

Содержимое структуры COMMANDS_RETURN_DATA_Type зависит от отправленной пользователем команды.

4.6.4 Команды передачи данных CODE_CMD_POWERSTEP01_W_MEM0..MEM3

Четыре команды передачи данных - CODE_CMD_POWERSTEP01_W_MEM0, CODE_CMD_POWERSTEP01_W_MEM1, CODE_CMD_POWERSTEP01_W_MEM2, CODE_CMD_POWERSTEP01_W_MEM3 используются для записи исполнительных программ в соответствующие области памяти контроллера. Поле DATA пакета содержит последовательность исполнительных команд в формате SMSD_CMD_Type. Максимальное количество команд для записи в контроллер – 255. Кодовое расстояние в адресном пространстве 4 байта.

В контроллер:

Поле	Значение	Порядок данных в пакете
XOR (1 байт)	x	0
VER (1 байт)	x	1
CMD_TYPE (1 байт)	CODE_CMD_POWERSTEP01_W_MEM0 = 0x03 or CODE_CMD_POWERSTEP01_W_MEM1 = 0x04 or CODE_CMD_POWERSTEP01_W_MEM2 = 0x05 or CODE_CMD_POWERSTEP01_W_MEM3 = 0x06	2
CMD_IDENTIFICATION (1 байт)	x	3
LENGTH_DATA (Младший байт)	x	4
LENGTH_DATA (Старший байт)	x	5
(1 ^я исполнительная команда) DATA [0] (SMSD_CMD_Type Младший байт)	x	6
DATA [1]	x	7
DATA [2]	x	8
(1 ^я исполнительная команда) DATA [3] (SMSD_CMD_Type Старший байт)	x	9
.....		
(последняя исполнительная команда – всего n команд) DATA [0] (SMSD_CMD_Type Младший байт)	x	n*4 - 3
DATA [1]	x	n*4 - 2
DATA [2]	x	n*4 - 1
(последняя исполнительная команда – всего n команд) DATA [3] (SMSD_CMD_Type Старший байт)	x	n*4

n<=255.

В ответ на запись программы контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE.

4.6.5 Команды передачи данных CODE_CMD_POWERSTEP01_R_MEM0..MEM3

Четыре команды передачи данных CODE_CMD_POWERSTEP01_R_MEM0, CODE_CMD_POWERSTEP01_R_MEM1, CODE_CMD_POWERSTEP01_R_MEM2, CODE_CMD_POWERSTEP01_R_MEM3 предназначены для чтения исполнительных программ из четырех банков памяти контроллера.

В контроллер:

Поле	Значение	Порядок данных в пакете
XOR (1 байт)	x	0
VER (1 байт)	x	1
CMD_TYPE (1 байт)	CODE_CMD_POWERSTEP01_R_MEM0 = 0x07 or CODE_CMD_POWERSTEP01_R_MEM1 = 0x08 or CODE_CMD_POWERSTEP01_R_MEM2 = 0x09 or CODE_CMD_POWERSTEP01_R_MEM3 = 0x0A	2
CMD_IDENTIFICATION (1 байт)	x	3
LENGTH_DATA (Младший байт)	0	4
LENGTH_DATA (Старший байт)	0	5
DATA	-	-

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_POWERSTEP01_R_MEM0 (or CODE_CMD_POWERSTEP01_R_MEM1 or CODE_CMD_POWERSTEP01_R_MEM2 or CODE_CMD_POWERSTEP01_R_MEM3). Поле DATA пакета содержит последовательность исполнительных команд в формате SMSD_CMD_Type. Кодовое расстояние в адресном пространстве 4 байта.

Из контроллера:

Поле	Значение	Порядок данных в пакете
XOR (1 байт)	X	0
VER (1 байт)	x	1
CMD_TYPE (1 байт)	CODE_CMD_POWERSTEP01_R_MEM0 = 0x07 or CODE_CMD_POWERSTEP01_R_MEM1 = 0x08 or CODE_CMD_POWERSTEP01_R_MEM2 = 0x09 or CODE_CMD_POWERSTEP01_R_MEM3 = 0x0A	2
CMD_IDENTIFICATION (1 байт)	x	3
LENGTH_DATA (Младший байт)	x	4
LENGTH_DATA (Старший байт)	x	5
(1 ^я исполнительная команда) DATA [0] (SMSD_CMD_Type Младший байт)	x	6
DATA [1]	x	7
DATA [2]	x	8
(1 ^я исполнительная команда) DATA [3] (SMSD_CMD_Type Старший байт)	x	9
.....		
(последняя исполнительная команда – всего n команд) DATA [0]	x	n*4 - 3

(SMSD_CMD_Type Младший байт)		
DATA [1]	x	n*4 – 2
DATA [2]	x	n*4 – 1
(последняя исполнительная команда – всего n команд) DATA [3] (SMSD_CMD_Type Старший байт)	x	n*4

n<=255.

4.6.6 Команда передачи данных CODE_CMD_CONFIG_SET

Команда передачи данных CODE_CMD_CONFIG_SET предназначена для записи в контроллер параметров подключения по сети Ethernet. Поле DATA пакета содержит структуру SMSD_LAN_CONFIG_Type (описание структуры далее в руководстве).

В контроллер:

Поле	Значение	Порядок данных в пакете
XOR (1 байт)	x	0
VER (1 байт)	x	1
CMD_TYPE (1 байт)	CODE_CMD_CONFIG_SET = 0x0B	2
CMD_IDENTIFICATION (1 байт)	X	3
LENGTH_DATA (Младший байт)	Sizeof(SMSD_LAN_CONFIG_Type)	0x19
LENGTH_DATA (Старший байт)		0x00
DATA [0] (SMSD_LAN_CONFIG_Type – Младший байт)	x	6
.....		
DATA [24] (SMSD_LAN_CONFIG_Type – Старший байт)	x	30

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE.

4.6.7 Команда передачи данных CODE_CMD_CONFIG_GET

Команда передачи данных CODE_CMD_CONFIG_GET предназначена для чтения из контроллера параметров подключения по сети Ethernet.

В контроллер:

Поле	Значение	Порядок данных в пакете
XOR (1 байт)	x	0
VER (1 байт)	x	1
CMD_TYPE (1 байт)	CODE_CMD_CONFIG_GET = 0x0C	2
CMD_IDENTIFICATION (1 байт)	X	3
LENGTH_DATA (Младший байт)	0	4
LENGTH_DATA (Старший байт)	0	5
Data	-	-

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_CONFIG_GET. Поле DATA пакета содержит структуру SMSD_LAN_CONFIG_Type (описание структуры далее в руководстве).

Из контроллера:

Поле	Значение		Порядок данных в пакете
XOR (1 байт)	x		0
VER (1 байт)	x		1
CMD_TYPE (1 байт)	CODE_CMD_CONFIG_GET = 0x0C		2
CMD_IDENTIFICATION (1 байт)	X		3
LENGTH_DATA (Младший байт)	Sizeof(SMSD_LAN_CONFIG_Type)	0x19	4
LENGTH_DATA (Старший байт)		0x00	5
DATA [0] (SMSD_LAN_CONFIG_Type – Младший байт)	x		6
.....			
DATA [24] (SMSD_LAN_CONFIG_Type – Старший байт)	x		30

4.6.8 Команда передачи данных CODE_CMD_PASSWORD_SET

Команда передачи данных CODE_CMD_PASSWORD_SET предназначена для задания нового пароля для авторизации.

В контроллер:

Поле	Значение		Порядок данных в пакете
XOR (1 байт)	x		0
VER (1 байт)	x		1
CMD_TYPE (1 байт)	CODE_CMD_PASSWORD_SET = 0x0D		2
CMD_IDENTIFICATION (1 байт)	x		3
LENGTH_DATA (Младший байт)	0x08		4
LENGTH_DATA (Старший байт)	0x00		5
DATA [0] (Password Младший байт)	x		6
DATA [1]	x		7
DATA [2]	x		8
DATA [3]	x		9
DATA [4]	x		10
DATA [5]	x		11
DATA [6]	x		12
DATA [7] (Password Старший байт)	x		13

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE.

4.6.9 Команда передачи данных CODE_CMD_ERROR_GET

Команда передачи данных CODE_CMD_ERROR_GET предназначена для чтения из памяти контроллера информации о количестве включений рабочего режима контроллера и статистики по ошибкам.

В контроллер:

Поле	Значение		Порядок данных в пакете
XOR (1 байт)	x		0
VER (1 байт)	x		1
CMD_TYPE (1 байт)	CODE_CMD_ERROR_GET = 0x0E		2

CMD_IDENTIFICATION (1 байт)	x	3
LENGTH_DATA (Младший байт)	0	4
LENGTH_DATA (Старший байт)	0	5
Data	-	-

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_ERROR_GET.

Из контроллера:

Поле	Значение	Порядок данных в пакете
XOR (1 байт)	x	0
VER (1 байт)	x	1
CMD_TYPE (1 байт)	CODE_CMD_ERROR_GET = 0x0E	2
CMD_IDENTIFICATION (1 байт)	x	3
LENGTH_DATA (Младший байт)	0x44 (=17*4)	4
LENGTH_DATA (Старший байт)	0	5
Data[0]	x	6
.....		
Data[67]	x	73

Поле DATA содержит 17 последовательных значений 4-х байтовых переменных, представляющих из себя счётчики событий:

N_STARTS – количество раз, когда были запитаны обмотки шагового двигателя.
ERROR_XT – количество внутренних ошибок запуска тактового генератора
ERROR_TIME_OUT – количество ошибок превышения времени выполнения основного цикла программы.
ERROR_INIT_POWERSTEP01 – количество ошибок инициализации чипа PowerSTEP01.
ERROR_INIT_WIZNET – количество ошибок инициализации чипа W5500.
ERROR_INIT_FRAM – количество ошибок инициализации чипа памяти FRAM.
ERROR_SOCKET – количество ошибок соединений по Ethernet
ERROR_FRAM – количество ошибок обмена с чипом памяти FRAM.
ERROR_INTERRUPT – количество ошибок обработки прерывания.
ERROR_EXTERN_5V – количество перегрузок по току, внутреннего выходного источника питания в 5В.
ERROR_EXTERN_VDD – количество выходов за диапазон питающего напряжения Контроллера.
ERROR_THERMAL_POWERSTEP01 – количество перегревов чипа PowerSTEP01
ERROR_THERMAL_BRAKE – количество перегревов тормозного резистора.
ERROR_COMMAND_POWERSTEP01 – количество ошибок при передаче управляющих команд в чип PowerSTEP01
ERROR_UVLO_POWERSTEP01 – количество ошибок
ERROR_STALL_POWERSTEP01 – количество ошибок
ERROR_WORK_PROGRAM – количество ошибок выполнения программы управления.

5. Структура COMMANDS_RETURN_DATA_Type

В ответ на команды, в поле CMD_TYPE которых содержится значение CODE_CMD_RESPONSE или CODE_CMD_POWERSTEP01, контроллер возвращает ответ, в области данных DATA которого содержится структура COMMANDS_RETURN_DATA_Type:

```
typedef struct
{ powerSTEP_STATUS_TypeDef
  uint8_t
  uint32_t
}COMMANDS_RETURN_DATA_Type;

STATUS_POWERSTEP01;
ERROR_OR_COMMAND;
RETURN_DATA;
```

STATUS_POWERSTEP01 – битовое поле длиной 16 бит, содержащее флаги состояния, отвечающие за текущее состояние схемы управления шаговым двигателем. Важная информация, поэтому включена в каждый ответ данного формата.

ERROR_OR_COMMAND – код статуса выполнения команды или код ошибки. Длина - 1 байт;

RETURN_DATA – поле данных длиной 4 байта.

5.1 Назначение битовых полей STATUS_POWERSTEP01

Статус состояния процесса управления шаговым двигателем содержится в структуре

powerSTEP_STATUS_TypeDef:

```
typedef struct {
    uint16_t    HiZ           : 1;
    uint16_t    BUSY          : 1;
    uint16_t    SW_F          : 1;
    uint16_t    SW_EVN        : 1;
    uint16_t    DIR           : 1;
    uint16_t    MOT_STATUS    : 2;
    uint16_t    CMD_ERROR     : 1;
    uint16_t    RESERVE       : 8;
} powerSTEP_STATUS_TypeDef;
```

HiZ – Z-состояние обмоток, если 1- обмотки шагового двигателя обесточены, 0 – обмотки шагового двигателя запитаны.

BUSY – ожидание, если 1- блок готов к выполнению следующей команды, 0 – выполняется предыдущая команда.

SW_F – если 1- функция SW включена, 0 – функция SW выключена.

SW_EVN – флаг события SW если 1- событие наступило, 0 – событие не наступило.

DIR – направление вращения, если 1- основное направление, 0 – обратное направление.

MOT_STATUS – статус текущего действия, если 0 – двигатель остановлен, 1 – ускорение, 2 - торможение, 3 - равномерное вращение шагового двигателя.

CMD_ERROR – ошибка выполнения команды, если 1- ошибка выполнения команды, 0 – без ошибок.

5.2 Список возможных значений поля ERROR_OR_COMMAND

Числовые значения начинаются с 0 и последовательно инкрементируются. Список возможных вариантов значений поля ERROR_OR_COMMAND:

0x00 - OK	- без ошибок
0x01 - OK_ACCESS	- признак получения доступа к управлению Контроллером
0x02 - ERROR_ACCESS	- признак ошибки получения доступа к управлению Контроллером
0x03 - ERROR_ACCESS_TIMEOUT	- признак того, что не истёк таймаут для повторной авторизации (1 сек)
0x04 - ERROR_XOR	- ошибка контрольной суммы команды
0x05 - ERROR_NO_COMMAND	- признак того, что такой команды не существует
0x06 - ERROR_LEN	- ошибочная длина пакета
0x07 - ERROR_RANGE	- выход за допустимый диапазон значений
0x08 - ERROR_WRITE	- ошибка записи
0x09 - ERROR_READ	- ошибка чтения
0x0A - ERROR_PROGRAMS	- для внутреннего пользования
0x0B - ERROR_WRITE_SETUP	- для внутреннего пользования
0x0C - NO_NEXT	- для внутреннего пользования
0x0D - END_PROGRAMS	- конец программы
0x0E - COMMAND_GET_STATUS_IN_EVENT	- в поле данных RETURN_DATA содержится битовая карта входных сигналов
0x0F - COMMAND_GET_MODE	- в поле данных RETURN_DATA содержится битовая карта текущих настроек Контроллера
0x10 - COMMAND_GET_ABS_POS	- в поле данных RETURN_DATA содержится текущее положение шагового двигателя в шагах



- 0x11 - **COMMAND_GET_EL_POS** - в поле данных RETURN_DATA содержится текущее электрическое положение двигателя
- 0x12 - **COMMAND_GET_SPEED** - в поле данных RETURN_DATA содержится текущая скорость двигателя
- 0x13 - **COMMAND_GET_MIN_SPEED** в поле данных RETURN_DATA содержится текущая установленная минимальная скорость двигателя
- 0x14 - **COMMAND_GET_MAX_SPEED** в поле данных RETURN_DATA содержится текущая установленная максимальная скорость двигателя
- 0x15 - **COMMAND_GET_STACK** в поле данных RETURN_DATA содержится информация о номере текущей выполняемой программы и номер выполняемой команды
- 0x16 - **STATUS_RELE_SET** – реле включено
- 0x17 - **STATUS_RELE_CLR** – реле выключено

6. Структура исполнительных команды управления SMSD_CMD_Type

Структура исполнительных команды управления SMSD_CMD_Type:

```
typedef struct
{ uint32_t      RESERVE   :3;
  uint32_t      ACTION    :1;
  uint32_t      COMMAND   :6;
  uint32_t      DATA     :22;
}SMSD_CMD_Type;
```

- RESERVE** – 3 бита, не используется;
- ACTION** – 1 бит = 0 – для внутреннего использования;
- COMMAND** – 6 бит – код исполнительной команды;
- DATA** – 22 бита – параметр команды, если исполнительная команда не требует параметра, значение данного поля = 0x00 (22 бита заполняются 0 и отправляются не изменяя длины поля).
- Размер структуры – 4 байта.
- Структура SMSD_CMD_Type используется в пакетах, содержащих команды передачи данных CMD_TYPE = CODE_CMD_POWERSTEP01, CODE_CMD_POWERSTEP01_W_MEM0...MEM3, CODE_CMD_POWERSTEP01_R_MEM0...MEM3.

Числовые значения поля COMMAND начинаются с 0 и последовательно инкрементируются. Список возможных вариантов значений поля:

- 0x00 CMD_PowerSTEP01_END,
- 0x01 CMD_PowerSTEP01_GET_SPEED,
- 0x02 CMD_PowerSTEP01_STATUS_IN_EVENT,
- 0x03 CMD_PowerSTEP01_SET_MODE,
- 0x04 CMD_PowerSTEP01_GET_MODE,
- 0x05 CMD_PowerSTEP01_SET_MIN_SPEED,
- 0x06 CMD_PowerSTEP01_SET_MAX_SPEED,
- 0x07 CMD_PowerSTEP01_SET_ACC,
- 0x08 CMD_PowerSTEP01_SET_DEC,
- 0x09 CMD_PowerSTEP01_SET_FS_SPEED,
- 0x0A CMD_PowerSTEP01_SET_MASK_EVENT
- 0x0B CMD_PowerSTEP01_GET_ABS_POS,
- 0x0C CMD_PowerSTEP01_GET_EL_POS,
- 0x0D CMD_PowerSTEP01_GET_STATUS_AND_CLR,
- 0x0E CMD_PowerSTEP01_RUN_F,
- 0x0F CMD_PowerSTEP01_RUN_R,
- 0x10 CMD_PowerSTEP01_MOVE_F,
- 0x11 CMD_PowerSTEP01_MOVE_R,
- 0x12 CMD_PowerSTEP01_GO_TO_F,
- 0x13 CMD_PowerSTEP01_GO_TO_R,



```

0x14 CMD_PowerSTEP01_GO_UNTIL_F,
0x15 CMD_PowerSTEP01_GO_UNTIL_R,
0x16 CMD_PowerSTEP01_SCAN_ZERO_F,
0x17 CMD_PowerSTEP01_SCAN_ZERO_R,
0x18 CMD_PowerSTEP01_SCAN_LABEL_F,
0x19 CMD_PowerSTEP01_SCAN_LABEL_R,
0x1A CMD_PowerSTEP01_GO_ZERO,
0x1B CMD_PowerSTEP01_GO_LABEL,
0x1C CMD_PowerSTEP01_GO_TO,
0x1D CMD_PowerSTEP01_RESET_POS,
0x1E CMD_PowerSTEP01_RESET_POWERSTEP01,
0x1F CMD_PowerSTEP01_SOFT_STOP,
0x20 CMD_PowerSTEP01_HARD_STOP,
0x21 CMD_PowerSTEP01_SOFT_HI_Z,
0x22 CMD_PowerSTEP01_HARD_HI_Z,
0x23 CMD_PowerSTEP01_SET_WAIT,
0x24 CMD_PowerSTEP01_SET_RELE,
0x25 CMD_PowerSTEP01_CLR_RELE,
0x26 CMD_PowerSTEP01_GET_RELE,
0x27 CMD_PowerSTEP01_WAIT_IN0,
0x28 CMD_PowerSTEP01_WAIT_IN1,
0x29 CMD_PowerSTEP01_GOTO_PROGRAM,
0x2A CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN0,
0x2B CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN1,
0x2C CMD_PowerSTEP01_LOOP_PROGRAM,
0x2D CMD_PowerSTEP01_CALL_PROGRAM,
0x2E CMD_PowerSTEP01_RETURN_PROGRAM,
0x2F CMD_PowerSTEP01_START_PROGRAM_MEM0,
0x30 CMD_PowerSTEP01_START_PROGRAM_MEM1,
0x31 CMD_PowerSTEP01_START_PROGRAM_MEM2,
0x32 CMD_PowerSTEP01_START_PROGRAM_MEM3,
0x33 CMD_PowerSTEP01_STOP_PROGRAM_MEM,
0x34 CMD_PowerSTEP01_STEP_CLOCK,
0x35 CMD_PowerSTEP01_STOP_USB,
0x36 CMD_PowerSTEP01_GET_MIN_SPEED,
0x37 CMD_PowerSTEP01_GET_MAX_SPEED,
0x38 CMD_PowerSTEP01_GET_STACK,
0x39 CMD_PowerSTEP01_GOTO_PROGRAM_IF_ZERO,
0x3A CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN_ZERO,
0x3B CMD_PowerSTEP01_WAIT_CONTINUE,
0x3C CMD_PowerSTEP01_SET_WAIT_2,
0x3D CMD_PowerSTEP01_SCAN_MARK2_F,
0x3E CMD_PowerSTEP01_SCAN_MARK2_R

```

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Byte [2]			Byte [1]			Byte [0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data (параметр команды)						Команда (код команды CMD PowerSTEP01)						Action		Reserve		

6.1 Исполнительная команда CMD_PowerSTEP01_END

Исполнительная команда CMD_PowerSTEP01_END = 0x00 предназначена для обозначения конца программы. Используется в конце программы, записываемой в память контроллера.



6.2 Исполнительная команда CMD_PowerSTEP01_GET_SPEED

Исполнительная команда CMD_PowerSTEP01_GET_SPEED = 0x01 предназначена для чтения текущего значения скорости.

Важное замечание: для корректного ответа контроллера о текущей скорости перед запросом CMD_PowerSTEP01_GET_SPEED должна быть установлена минимальная скорость двигателя - команда CMD_PowerSTEP01_SET_MIN_SPEED = 0x00. В противном случае контроллер может выдавать неверные значения для низких скоростей и в случае остановки двигателя.

Ниже дан пример пакета передачи данных для запроса текущей скорости:

В контроллер:

Поле	Значение	Порядок данных в пакете
XOR (1 байт)	x	0
VER (1 байт)	x	1
CMD_TYPE (1 байт)	CODE_CMD_POWERSTEP01 = 0x02	2
CMD_IDENTIFICATION (1 байт)	x	3
LENGTH_DATA (Младший байт)	Sizeof (SMSD_CMD_Type)=0x04	0x04
LENGTH_DATA (Старший байт)		0x00
DATA [0] (SMSD_CMD_Type Младший байт)	0x10	6
DATA [1]	0x00	7
DATA [2]	0x00	8
DATA [3] (SMSD_CMD_Type Старший байт)	0x00	9

Поле DATA пакета содержит структуру SMSD_CMD_Type, поле command которой содержит код команды запроса скорости CMD_PowerSTEP01_GET_SPEED.

Битовая раскладка структуры SMSD_CMD_Type:

Поле DATA	DATA[3]=0x00			DATA[2]=0x00			DATA[1]=0x00			DATA[0]=0x10							
	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = 0x00						Command CMD_PowerSTEP01_GET_SPEED = 0x01						Action	Reserve			
Значение	0			0			0	0	0	0	0	0	1	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = COMMAND_GET_SPEED, RETURN_DATA - значение текущей скорости двигателя.

Из контроллера:

Поле	Значение		Порядок данных в пакете
XOR (1 байт)	x		0
VER (1 байт)	x		1
CMD_TYPE (1 байт)	CODE_CMD_RESPONSE = 0x01		2
CMD_IDENTIFICATION (1 байт)	x		3
LENGTH_DATA (Младший байт)	Sizeof (COMMANDS_RETURN_DATA_Type	0x07	4
LENGTH_DATA (Старший байт)		0x00	5
DATA [0] - Младший байт (COMMANDS_RETURN_DATA)	x		6
DATA [1]	x		7
DATA [2] = ERROR_OR_COMMAND	= COMMAND_GET_SPEED		8
DATA [3] = RETURN_DATA[0]	(Младший байт значения текущей скорости)		9
DATA [4] = RETURN_DATA[1]	x		10
DATA [5] = RETURN_DATA[2]	x		11
DATA [6] - Старший байт (COMMANDS_RETURN_DATA) = RETURN_DATA[3]	(Старший байт значения текущей скорости)		12

6.3 Исполнительная команда CMD_PowerSTEP01_STATUS_IN_EVENT

Исполнительная команда CMD_PowerSTEP01_STATUS_IN_EVENT = 0x02 предназначена для чтения текущего состояния входных сигналов.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]						
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1 0
Назначение	Data = 0x00						Command CMD_PowerSTEP01_ STATUS_IN_EVENT = 0x02				Action		Reserve			
Значение	0			0			0	0	0	0	0	1	0	0	0	0 0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = COMMAND_GET_STATUS_IN_EVENT, RETURN_DATA – битовую карту состояний входных сигналов:

	Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0
RETURN_DATA[0]	INT_7	INT_6	INT_5	INT_4	INT_3	INT_2	INT_1	INT_0
RETURN_DATA[1]	Mask_7	Mask_6	Mask_5	Mask_4	Mask_3	Mask_2	Mask_1	Mask_0
RETURN_DATA[2]	Wait_7	Wait_6	Wait_5	Wait_4	Wait_3	Wait_2	Wait_1	Wait_0
RETURN_DATA[3]	Not use							

INT_X – Событие на входном сигнале X;
Mask_X – Маскирование входного сигнала X;
Wait_X – Ожидание входного сигнала X.

6.4 Исполнительная команда CMD_PowerSTEP01_SET_MODE

Исполнительная команда CMD_PowerSTEP01_SET_MODE = 0x03 предназначена для установки параметров управления двигателем.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Byte [2]			Byte [1]			Byte [0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Поле Data структуры SMSD_CMD_Type						CommandCMD_PowerSTEP01_SET_M ODE = 0x03								Action	Reserve	
Значение	Зависит от значения поля Data						0	0	0	0	0	1	1	0	0	0	0

Битовая раскладка поля Data структуры SMSD_CMD_Type:

Байт[3] – биты 7..0								Байт[2] – биты 7..0								Байт[1] биты 7..2					
21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			STOP_CURRENT		WORK_CURRENT								MICROSTEPPING			MOTOR_TYPE				CURRENT_OR_VOLTAGE	

CURRENT_OR_VOLTAGE - тип управления двигателем:

0 – напряжение (вольтовый режим),

1 – ток (токовый режим)

MOTOR_TYPE – модель двигателя для вольтового режима управления:

Значение		Макс. ток фазы, А	Сопро- тивление фазы, Ом	Индуктив- ность фазы, мГн	Угло- вой шаг	Модель
SMSD-4.2LAN	SMSD-8.0LAN					
0	0	-	-	-	-	нет
1	1	1.33	2.1	2.5	1.8	FL42STH33-1334 1.8 deg
2	2	1.33	2.1	4.2	0.9	FL42STH33-1334 0.9 deg
3	3	1.2	3.3	3.4	0.9	FL42STH38-1206 0.9 deg
4	4	1.68	1.65	3.2	1.8	FL42STH38-1684 1.8 deg
5	5	1.68	1.64	3.2	0.9	FL42STH38-1684 0.9 deg
6	6	1.2	3.3	2.8	0.8	FL42STH47-1206 1.8 deg
7	7	1.68	1.65	2.8	1.8	FL42STH47-1684 1.8 deg
8	8	1.68	1.65	4.1	0.9	FL42STH47-1684 0.9 deg
9	9	1.2	6	7	1.8	FL42STH60-1206 1.8 deg
10	10	1.2	12.1	36.7	0.9	FL42STH60-1206 0.9 deg
11	11	1.56	1.8	3.6	1.8	FL57ST41-1564 1.8 deg
12	12	1.0	16.7	46.5	1.8	FL57ST76-1006 1.8 deg
13	13	1.5	3.6	6	1.8	FL57ST76-1506 1.8 deg
14	14	1.0	5.7	5.4	1.8	FL57STH41-1006 1.8 deg
15	15	1.0	5.7	8	0.9	FL57STH41-1006 0.9 deg
16	16	2.8	0.7	1.4	1.8	FL57STH41-2804 1.8 deg
17	17	2.8	0.7	2.2	0.9	FL57STH41-2804 0.9 deg
18	18	1.0	6.6	8.6	1.8	FL57STH51-1006 1.8 deg
19	19	2.8	0.83	2.2	1.8	FL57STH51-2804 1.8 deg
20	20	2.8	0.9	3.7	0.9	FL57STH51-2804 0.9 deg
21	21	1.0	7.4	10	1.8	FL57STH56-1006 1.8 deg
22	22	2.0	1.8	2.5	1.8	FL57STH56-2006 1.8 deg
23	23	2.8	0.9	2.5	1.8	FL57STH56-2804 1.8 deg
24	24	1.0	8.6	14	1.8	FL57STH76-1006 1.8 deg
25	25	2.8	1.13	3.6	1.8	FL57STH76-2804 1.8 deg
26	26	2.8	1.13	5.6	0.9	FL57STH76-2804 0.9 deg



27	27	2.0	1.2	4.6	1.8	FL60STH65-2008 1.8 deg параллельное соединение обмоток
28	28	2.0	4.8	18.4	1.8	FL60STH65-2008 1.8 deg последовательное соединение обмоток
29	29	2.0	1.5	6.8	1.8	FL60STH86-2008 1.8 deg параллельное соединение обмоток
30	30	2.0	6	7.2	1.8	FL60STH86-2008 1.8 deg последовательное соединение обмоток
31	31	2.8	0.7	3.9	1.8	FL86STH65-2808 1.8 deg параллельное соединение обмоток
32	32	2.8	2.8	15.6	1.8	FL86STH65-2808 1.8 deg последовательное соединение обмоток
33	33	4.2	0,375	3.4	1.8	FL86STH80-4208 1.8 deg параллельное соединение обмоток
34	34	4.2	1.5	13.6	1.8	FL86STH80-4208 1.8 deg последовательное соединение обмоток
35	35	4.2	0.45	6	1.8	FL86STH118-4208 1.8 deg параллельное соединение обмоток
36	36	4.2	1.8	24	1.8	FL86STH118-4208 1.8 deg последовательное соединение обмоток
37	37	4.2	0,625	8	1.8	FL86STH156-4208 1.8 deg параллельное соединение обмоток
38	38	4.2	2.5	32	1.8	FL86STH156-4208 1.8 deg последовательное соединение обмоток
-	39	6.0	0.6	6.5	1.8	FL86STH118-6004 1.8 deg
-	40	6.2	0.75	9	1.8	FL86STH156-6204 1.8 deg
-	41	5.5	0.9	12	1.8	FL110STH99-5504 1.8 deg
-	42	6.5	0.8	15	1.8	FL110STH150-6504 1.8 deg
-	43	8	0.67	12	1.8	FL110STH201-8004 1.8 deg
39	44	0.3	32	40	1.8	ДШ3934-0,3А
40	45	0.67	8.5	7.5	1.8	ДШ2851-0,7А
41	46	1.68	2.3	3.4	1.8	ДШ4248-1,7А
42	47	3.0	1.0	3.4	1.8	ДШ5776-3,0А
43	48	3.0	1.45	6.5	1.8	ДШ57112-3,0А
44	49	3.0	1.2	6.4	1.8	ДШ8665-3,0А
45	50	4.5	0.36	3.0	1.8	ДШ8682-4,5А
-	51	6.0	0.6	5.7	1.8	ДШ86118-6,0А
-	52	6.2	0.7	8.5	1.8	ДШ86156-6,2А
-	53	8.0	0.8	16	1.8	ДШ110201-8,0А
-	54	6.0	0.8	8.7	1.8	ДШ130280-6,0А

MICROSTEPPING – дробление шага:

- 0 - 1
- 1 - 1/2
- 2 - 1/4
- 3 - 1/8
- 4 - 1/16
- 5 - 1/32
- 6 - 1/64
- 7 - 1/128

WORK CURRENT – рабочий ток (используется в токовом режиме управления двигателем). Значение рабочего тока определяется по формуле: $0.1A \cdot \text{Значение параметра}$; $1 \leq \text{Значение} \leq 80$. Значения тока могут быть следующие:

1 - 0.1A	15 - 1.5A	29 - 2.9A	43 - 4.3A	57 - 5.7A	71 - 7.1A
2 - 0.2A	16 - 1.6A	30 - 3.0A	44 - 4.4A	58 - 5.8A	72 - 7.2A
3 - 0.3A	17 - 1.7A	31 - 3.1A	45 - 4.5A	59 - 5.9A	73 - 7.3A
4 - 0.4A	18 - 1.8A	32 - 3.2A	46 - 4.6A	60 - 6.0A	74 - 7.4A
5 - 0.5A	19 - 1.9A	33 - 3.3A	47 - 4.7A	61 - 6.1A	75 - 7.5A
6 - 0.6A	20 - 2.0A	34 - 3.4A	48 - 4.8A	62 - 6.2A	76 - 7.6A
7 - 0.7A	21 - 2.1A	35 - 3.5A	49 - 4.9A	63 - 6.3A	77 - 7.7A
8 - 0.8A	22 - 2.2A	36 - 3.6A	50 - 5.0A	64 - 6.4A	78 - 7.8A
9 - 0.9A	23 - 2.3A	37 - 3.7A	51 - 5.1A	65 - 6.5A	79 - 7.9A
10 - 1.0A	24 - 2.4A	38 - 3.8A	52 - 5.2A	66 - 6.6A	80 - 8.0A
11 - 1.1A	25 - 2.5A	39 - 3.9A	53 - 5.3A	67 - 6.7A	
12 - 1.2A	26 - 2.6A	40 - 4.0A	54 - 5.4A	68 - 6.8A	
13 - 1.3A	27 - 2.7A	41 - 4.1A	55 - 5.5A	69 - 6.9A	
14 - 1.4A	28 - 2.8A	42 - 4.2A	56 - 5.6A	70 - 7.0A	

Допустимый диапазон значений для блоков SMSD-4.2LAN: 1 – 42; для блоков SMSD-8.0LAN: 1 – 80.

STOP CURRENT – ток удержания – устанавливается в процентах от значения рабочего тока:

- 0 - 25%
- 1 - 50%
- 2 - 75%
- 3 - 100%

В ответ контроллер отправляет пакет с кодом команды `CMD_TYPE = CODE_CMD_RESPONSE`, поле `DATA` которого содержит структуру `COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK`.

6.5 Исполнительная команда `CMD_PowerSTEP01_GET_MODE`

Исполнительная команда `CMD_PowerSTEP01_GET_MODE = 0x04` предназначена для чтения настроек управления двигателем.

Битовая раскладка структуры `SMSD_CMD_Type`:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = 0x00						Command CMD_PowerSTEP01_ GET_MODE = 0x04						Action	Reserve			
Значение	0			0			0	0	0	0	1	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды `CMD_TYPE = CODE_CMD_RESPONSE`, поле `DATA` которого содержит структуру `COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = COMMAND_GET_MODE`, `RETURN_DATA` содержит информацию о настройках управления двигателем.

RETURN_DATA[3]		RETURN_DATA[2]					RETURN_DATA[1]			RETURN_DATA[0]				
0..7		5..7	4	3	2	1	0	2..7		1	0	7	1..6	0
Не используется		PROGRAM_N		STOP_CURRENT		WORK_CURRENT		MICRO-STEPPING		MOTOR_TYPE		CURRENT_OR_VOLTAGE		

Назначения полей STOP_CURRENT, WORK_CURRENT, MICROSTEPPING, MOTOR_TYPE, CURRENT_OR_VOLTAGE такие же, как для команды установки настроек CMD_PowerSTEP01_SET_MODE.

Поле PROGRAM_N содержит номер программы для старта внешними сигналами.

6.6 Исполнительная команда CMD_PowerSTEP01_SET_MIN_SPEED

Исполнительная команда CMD_PowerSTEP01_SET_MIN_SPEED = 0x05 предназначена для установки минимальной скорости вращения двигателя. Поле DATA должно содержать значение скорости в диапазоне от 0 до 950 шагов/сек.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Значение минимальной скорости							Command CMD_PowerSTEP01_ SET_MIN_SPEED = 0x05							Action	Reserve		
Значение	Зависит от значения поля Data							0	0	0	1	0	1	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.7 Исполнительная команда CMD_PowerSTEP01_SET_MAX_SPEED

Исполнительная команда CMD_PowerSTEP01_SET_MAX_SPEED = 0x06 предназначена для установки максимальной скорости шагового двигателя. Поле DATA должно содержать значение скорости в диапазоне от 16 до 15600 шагов/сек.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Значение максимальной скорости							Command CMD_PowerSTEP01_ SET_MAX_SPEED = 0x06							Action	Reserve		
Значение	Зависит от значения поля Data							0	0	0	1	1	0	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.8 Исполнительная команда CMD_PowerSTEP01_SET_ACC

Исполнительная команда CMD_PowerSTEP01_SET_ACC = 0x07 предназначена для установки значения ускорения двигателя. Поле DATA должно содержать значение ускорения в диапазоне от 15 до 59000 шагов/сек².

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Ускорение							Command CMD_PowerSTEP01_SET_ACC = 0x07							Action	Reserve		
Значение	Зависит от значения поля Data							0	0	0	1	1	1	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.9 Исполнительная команда CMD_PowerSTEP01_SET_DEC

Исполнительная команда CMD_PowerSTEP01_SET_DEC = 0x08 предназначена для установки значения замедления шагового двигателя. Поле DATA должно содержать значение замедления в диапазоне от 15 до 59000 шагов/сек².

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Замедление						Command CMD_PowerSTEP01_SET_DEC = 0x08							Action	Reserve			
Значение	Зависит от значения поля Data						0	0	01	0	0	0	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.10 Исполнительная команда CMD_PowerSTEP01_SET_FS_SPEED

Исполнительная команда CMD_PowerSTEP01_SET_FS_SPEED = 0x09 предназначена для установки скорости перехода на полношаговый режим работы. Поле DATA должно содержать скорость перехода в полношаговый режим в диапазоне от 15 до 15600 шагов/сек.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Значение скорости перехода в полношаговый режим							Command CMD_PowerSTEP01_SET_FS_SPE ED = 0x09							Action	Reserve		
Значение	Зависит от значения поля Data							0	0	01	0	0	1	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.11 Исполнительная команда CMD_PowerSTEP01_SET_MASK_EVENT

Исполнительная команда CMD_PowerSTEP01_SET_MASK_EVENT = 0x0A предназначена для маскирования входных сигналов. В случае, если значение маски сигнала установлено 1 – контроллер обрабатывает сигналы на соответствующем физическом входе. Если значение маски сигнала установлено 0 – контроллер не обрабатывает сигналы на соответствующем физическом входе.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = Маска сигналов						Command CMD_PowerSTEP01_ SET_MASK_EVENT = 0x0A						Action		Reserve		
Значение	Зависит от значения поля Data						0	0	01	0	1	0	0	0	0	0	0

Битовая раскладка поля Data:

	Байт[3] bits 7..0								Байт[2] bits 7..0								Байт[1] bits 7..2					
Бит	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Mask_7	Mask_6	Mask_5	Mask_4	Mask_3	Mask_2	Mask_1	Mask_0

Mask_X – Маскирование сигнала на входе X.

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.12 Исполнительная команда CMD_PowerSTEP01_GET_ABS_POS

Исполнительная команда CMD_PowerSTEP01_GET_ABS_POS = 0x0B предназначена для чтения положения двигателя.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = 0x00						Command CMD_PowerSTEP01_ GET_ABS_POS = 0x0B						Action		Reserve		
Значение	0						0	0	01	0	1	1	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = COMMAND_GET_ABS_POS, RETURN_DATA содержит значение текущего положения двигателя в диапазоне – $(2^{21}) \dots + (2^{21}-1)$.

6.13 Исполнительная команда CMD_PowerSTEP01_GET_EL_POS

Исполнительная команда CMD_PowerSTEP01_GET_EL_POS = 0x0C предназначена для чтения электрического положения ротора двигателя.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = 0x00						Command CMD_PowerSTEP01_ GET_EL_POS = 0x0C							Action	Reserve			
Значение	0						0	0	01	1	0	0	0	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = COMMAND_GET_EL_POS, RETURN_DATA содержит информацию о текущем электрическом положении ротора: биты 8,7 – текущий шаг, bits 6..0 – текущий микрошаг в пределах полного шага (измеряется как 1/128 от величина полного шага).

RETURN_DATA[3]								RETURN_DATA[2]								RETURN_DATA[1]								RETURN_DATA[0]							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Не используется																Текущий шаг								Текущий микрошаг							

6.14 Исполнительная команда CMD_PowerSTEP01_GET_STATUS_AND_CLR

Исполнительная команда CMD_PowerSTEP01_GET_STATUS_AND_CLR = 0x0D предназначена для чтения текущего статуса контроллера и сброса всех флагов ошибок.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = 0x00						Command CMD_PowerSTEP01_ GET_STATUS_AND_CLR = 0x0D							Action	Reserve			
Значение	0						0	0	01	1	0	1	0	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.15 Исполнительная команда CMD_PowerSTEP01_RUN_F

Исполнительная команда CMD_PowerSTEP01_RUN_F = 0x0E предназначена для старта непрерывного вращения двигателя в прямом направлении на указанной скорости. Поле DATA содержит значение скорости вращения в диапазоне от 15 до 15600 шагов/сек.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = Значение скорости вращения						Command CMD_PowerSTEP01_ RUN_F = 0x0E							Action	Reserve		
Значение	Зависит от значения поля Data						0	0	01	1	1	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.16 Исполнительная команда CMD_PowerSTEP01_RUN_R

Исполнительная команда CMD_PowerSTEP01_RUN_F = 0x0E предназначена для старта непрерывного вращения двигателя в обратном направлении на указанной скорости. Поле DATA содержит значение скорости вращения в диапазоне от 15 до 15600 шагов/сек.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = Значение скорости вращения						Command CMD_PowerSTEP01_ RUN_R = 0x0F							Action	Reserve		
Значение	Зависит от значения поля Data						0	0	01	1	1	1	1	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.17 Исполнительная команда CMD_PowerSTEP01_MOVE_F

Исполнительная команда CMD_PowerSTEP01_MOVE_F = 0x10 предназначена для перемещения двигателя в прямом направлении на указанную величину. Поле DATA должно содержать величину перемещения в диапазоне $-(2^{21}) \dots +(2^{21}-1)$. Скорость перемещения определяется предварительно заданными параметрами минимальной скорости, максимальной скорости, ускорения и замедления. Перед отправкой данной команды двигатель должен быть остановлен (поле Mot_Status структуры powerSTEP_STATUS_Type = 0).

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

В командах задания перемещения единицы измерения параметра - микрошаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Перемещение						Command CMD_PowerSTEP01_ MOVE_F = 0x10							Action	Reserve			
Значение	Зависит от значения поля Data						0	1	0	0	0	0	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.18 Исполнительная команда CMD_PowerSTEP01_MOVE_R

Исполнительная команда CMD_PowerSTEP01_MOVE_F = 0x10 предназначена для перемещения двигателя в обратном направлении на указанную величину. Поле DATA должно содержать величину перемещения в диапазоне $-(2^{21}) \dots +(2^{21}-1)$. Скорость перемещения определяется предварительно заданными параметрами минимальной скорости, максимальной скорости, ускорения и замедления. Перед отправкой данной команды двигатель должен быть остановлен (поле Mot_Status структуры powerSTEP_STATUS_Type = 0).

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

В командах задания перемещения единицы измерения параметра - микрошаги в секунду.



Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Перемещение						Command CMD_PowerSTEP01_ MOVE_R = 0x11							Action	Reserve			
Значение	Зависит от значения поля Data						0	1	0	0	0	0	1	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.19 Исполнительная команда CMD_PowerSTEP01_GO_TO_F

Исполнительная команда CMD_PowerSTEP01_GO_TO_F = 0x12 предназначена для перемещения в заданную позицию в прямом направлении. Поле DATA должно содержать требуемое положение двигателя в диапазоне $-(2^{21}) \dots +(2^{21}-1)$. Скорость перемещения определяется предварительно заданными параметрами минимальной скорости, максимальной скорости, ускорения и замедления.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду. В командах задания перемещения единицы измерения параметра - микрошаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = Требуемое положение двигателя						Command CMD_PowerSTEP01_ GO_TO_F = 0x12							Action	Reserve		
Значение	Зависит от значения поля Data						0	1	0	0	1	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.20 Исполнительная команда CMD_PowerSTEP01_GO_TO_R

Исполнительная команда CMD_PowerSTEP01_GO_TO_R = 0x13 для перемещения в заданную позицию в обратном направлении. Поле DATA должно содержать требуемое положение двигателя в диапазоне $-(2^{21}) \dots +(2^{21}-1)$. Скорость перемещения определяется предварительно заданными параметрами минимальной скорости, максимальной скорости, ускорения и замедления.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду. В командах задания перемещения единицы измерения параметра - микрошаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]									
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0		
Назначение	Data = Требуемое положение двигателя							Command CMD_PowerSTEP01_ GO_TO_R = 0x13							Action	Reserve			
Значение	Зависит от значения поля Data							0	1	0	0	1	1	0	0	0	0		

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.21 Исполнительная команда CMD_PowerSTEP01_GO_UNTIL_F

Исполнительная команда CMD_PowerSTEP01_GO_UNTIL_F = 0x14 предназначена для старта вращения двигателя в прямом направлении на максимальной скорости до получения сигнала на вход, номер которого задан в поле DATA. После получения сигнала двигатель останавливается с заданным замедлением. При отработке команды учитывается заданная маска сигнала. Маскирование сигналов может быть изменено командой CMD_PowerSTEP01_SET_MASK_EVENT.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = Номер сигнала						Command CMD_PowerSTEP01_ GO_UNTIL_F = 0x14							Action	Reserve		
Значение	Зависит от значения поля Data						0	1	0	1	0	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.22 Исполнительная команда CMD_PowerSTEP01_GO_UNTIL_R

Исполнительная команда CMD_PowerSTEP01_GO_UNTIL_R = 0x15 предназначена для старта вращения двигателя в обратном направлении на максимальной скорости до получения сигнала на вход, номер которого задан в поле DATA. После получения сигнала двигатель останавливается с заданным замедлением. При отработке команды учитывается заданная маска сигнала. Маскирование сигналов может быть изменено командой CMD_PowerSTEP01_SET_MASK_EVENT.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = Номер сигнала						Command CMD_PowerSTEP01_ GO_UNTIL_R = 0x15							Action	Reserve		
Значение	Зависит от значения поля Data						0	1	0	1	0	1	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.23 Исполнительная команда CMD_PowerSTEP01_SCAN_ZERO_F

Исполнительная команда CMD_PowerSTEP01_SCAN_ZERO_F = 0x16 предназначена для поиска нулевого положения в прямом направлении с заданной скоростью. Движение продолжается до поступления сигнала на вход SET_ZERO. При поступлении сигнала двигатель останавливается, текущее положение принимается за нулевое. Поле DATA должно содержать скорость движения при поиске нулевого положения.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Скорость поиска нулевого положения							Command CMD_PowerSTEP01_SCAN_ZERO_F = 0x16							Action	Reserve		
Значение	Зависит от значения поля Data							0	1	0	1	1	0	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.24 Исполнительная команда CMD_PowerSTEP01_SCAN_ZERO_R

Исполнительная команда CMD_PowerSTEP01_SCAN_ZERO_R = 0x17 предназначена для поиска нулевого положения в обратном направлении с заданной скоростью. Движение продолжается до поступления сигнала на вход SET_ZERO. При поступлении сигнала двигатель останавливается, текущее положение принимается за нулевое. Поле DATA должно содержать скорость движения при поиске нулевого положения.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]						
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1 0
Назначение	Data = Скорость поиска нулевого положения						Command CMD_PowerSTEP01_SCAN_ZERO_F = 0x17							Action	Reserve	
Значение	Зависит от значения поля Data						0	1	0	1	1	1	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.25 Исполнительная команда CMD_PowerSTEP01_SCAN_LABEL_F

Исполнительная команда CMD_PowerSTEP01_SCAN_LABEL_F = 0x18 предназначена для поиска метки положения в прямом направлении. Движение продолжается до поступления сигнала на вход IN1. При поступлении сигнала двигатель останавливается, текущее положение запоминается как метка. Поле DATA определяет скорость движения при поиске метки.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]						
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1 0
Назначение	Data = Скорость поиска метки						Command CMD_PowerSTEP01_SCAN_LABEL_F = 0x18							Action	Reserve	
Значение	Зависит от значения поля Data						0	1	1	0	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.26 Исполнительная команда CMD_PowerSTEP01_SCAN_LABEL_R

Исполнительная команда CMD_PowerSTEP01_SCAN_LABEL_R = 0x19 предназначена для поиска метки положения в обратном направлении. Движение продолжается до поступления сигнала на вход IN1. При поступлении сигнала двигатель останавливается, текущее положение запоминается как метка. Поле DATA определяет скорость движения при поиске метки.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду.



Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Скорость поиска метки						Command CMD_PowerSTEP01_ SCAN_LABEL_R = 0x19							Action	Reserve			
Значение	Зависит от значения поля Data						0	1	1	0	0	1	0	0	0	0		

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.27 Исполнительная команда CMD_PowerSTEP01_GO_ZERO

Исполнительная команда CMD_PowerSTEP01_GO_ZERO = 0x1A предназначена для перемещения в нулевое положение.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_GO_ZERO = 0x1A							Action	Reserve		
Значение	0						0	1	1	0	1	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.28 Исполнительная команда CMD_PowerSTEP01_GO_LABEL

Исполнительная команда CMD_PowerSTEP01_GO_LABEL = 0x1B предназначена для перемещения в положение, которое было отмечено как метка.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	0						Command CMD_PowerSTEP01_GO_LABEL = 0x1B							Action	Reserve			
Значение	0						0	1	1	0	1	1	0	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.29 Исполнительная команда CMD_PowerSTEP01_GO_TO

Исполнительная команда CMD_PowerSTEP01_GO_TO = 0x1C предназначена для перемещения в заданное положение по кратчайшему пути.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Заданное положение						Command CMD_PowerSTEP01_GO_TO = 0x1C							Action	Reserve			
Значение	Зависит от значения поля Data						0	1	1	1	0	0	0	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.30 Исполнительная команда CMD_PowerSTEP01_RESET_POS

Исполнительная команда CMD_PowerSTEP01_RESET_POS = 0x1D предназначена для обнуления счетчика текущего положения. После выполнения команды текущее положение принимается как нулевое.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_ RESET_POS = 0x1D							Action	Reserve		
Значение	0						0	1	1	1	0	1	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.31 Исполнительная команда CMD_PowerSTEP01_RESET_POWERSTEP01

Исполнительная команда CMD_PowerSTEP01_RESET_POWERSTEP01 = 0x1E используется для осуществления полного аппаратного и программного сброса модуля управления шаговым двигателем, но не контроллера в целом.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_ RESET_POWERSTEP01 = 0x1E							Action	Reserve		
Значение	0						0	1	1	1	1	0	0	0	0		

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.32 Исполнительная команда CMD_PowerSTEP01_SOFT_STOP

Исполнительная команда CMD_PowerSTEP01_SOFT_STOP = 0x1F используется для плавной остановки двигателя с заданным ускорением. После остановки двигатель удерживает положение с заданным током удержания.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_ SOFT_STOP = 0x1F							Action	Reserve		
Значение	0						0	1	1	1	1	1	1	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.33 Исполнительная команда CMD_PowerSTEP01_HARD_STOP

Исполнительная команда CMD_PowerSTEP01_HARD_STOP = 0x20 используется для резкой остановки шагового двигателя. После остановки двигатель удерживает положение с заданным током удержания.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	0						Command CMD_PowerSTEP01_ HARD_STOP = 0x20							Action	Reserve			
Значение	0						1	0	0	0	0	0	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.34 Исполнительная команда CMD_PowerSTEP01_SOFT_HI_Z

Исполнительная команда CMD_PowerSTEP01_SOFT_HI_Z = 0x21 используется для плавной остановки шагового двигателя с заданным ускорением. После остановки питания с обмоток двигателя снимается.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_SOFT_HI_Z = 0x21							Action	Reserve		
Значение	0						1	0	0	0	0	0	1	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.35 Исполнительная команда CMD_PowerSTEP01_HARD_HI_Z

Исполнительная команда CMD_PowerSTEP01_HARD_HI_Z = 0x22 используется для резкой остановки и обесточивания обмоток двигателя.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_HARD_HI_Z = 0x22							Action	Reserve		
Значение	0						1	0	0	0	0	1	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.36 Исполнительная команда CMD_PowerSTEP01_SET_WAIT

Исполнительная команда CMD_PowerSTEP01_SET_WAIT = 0x23 предназначена для задания паузы. Поле DATA должно содержать время ожидания паузы в диапазоне от 0 до 3600000мс.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = время ожидания							Command CMD_PowerSTEP01_ SET_WAIT = 0x23							Action	Reserve		
Значение	Зависит от значения поля Data							1	0	0	0	1	1	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.37 Исполнительная команда CMD_PowerSTEP01_SET_RELE

Исполнительная команда CMD_PowerSTEP01_SET_RELE = 0x24 предназначена для включения реле контроллера.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_SET_RELE = 0x24							Action	Reserve		
Значение	0						1	0	0	1	0	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = STATUS_RELE_SET.

6.38 Исполнительная команда CMD_PowerSTEP01_CLR_RELE

Исполнительная команда CMD_PowerSTEP01_CLR_RELE = 0x25 предназначена для выключения реле контроллера.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_CLR_RELE = 0x25							Action	Reserve		
Значение	0						1	0	0	1	0	1	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = STATUS_RELE_CLR.

6.39 Исполнительная команда CMD_PowerSTEP01_GET_RELE

Исполнительная команда CMD_PowerSTEP01_GET_RELE = 0x26 предназначена для запроса состояния реле контроллера.

Битовая раскладка структуры SMSD_CMD_Type:

Битовый разбиение структуры CMD01_CMD_Type.																		
	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	0							Command CMD_PowerSTEP01_GET_RELE = 0x26							Action	Reserve		
Значение	0							1	0	0	1	1	0	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: значение ERROR_OR_COMMAND зависит от текущего состояния реле - STATUS_RELE_SET (реле включено) или STATUS_RELE_CLR (реле выключено).

6.40 Исполнительная команда CMD_PowerSTEP01_WAIT_IN0

Исполнительная команда CMD_PowerSTEP01_WAIT_IN0 = 0x27 предназначена для ожидания поступления сигнала на вход IN0.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_WAIT_IN0=0x27							Action	Reserve		
Значение	0						1	0	0	1	1	1	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.41 Исполнительная команда CMD_PowerSTEP01_WAIT_IN1

Исполнительная команда CMD_PowerSTEP01_WAIT_IN1 = 0x28 предназначена для ожидания поступления сигнала на вход IN1.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_WAIT_IN1= 0x28							Action	Reserve		
Значение	0						1	0	1	0	0	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.42 Исполнительная команда CMD_PowerSTEP01_GOTO_PROGRAM

Исполнительная команда CMD_PowerSTEP01_GOTO_PROGRAM = 0x29 предназначена для безусловного перехода к заданной команде заданной программы. Поле DATA должно содержать информацию о требуемом номере программы и порядковом номере команды в программе: биты 0..7 поля Data определяют номер команды, биты 8,9 поля DATA определяют номер программы.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = Номер программы и команды						Command CMD_PowerSTEP01_ GOTO_PROGRAM = 0x29							Action	Reserve		
Значение	Зависит от значения поля Data						1	0	1	0	0	1	0	0	0	0	0

Битовая раскладка поля Data:

	Байт[3] биты 7..0								Байт[2] биты 7..0								Байт[1] биты 7..2							
Номер бита поля Data	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	0	0	0	0	0	0	Номер программы				Номер команды							

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.43 Исполнительная команда CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN0

Исполнительная команда CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN0 = 0x2A предназначена для перехода к заданной команде заданной программы, если на входе IN0 присутствует сигнал. Поле DATA должно содержать информацию о требуемом номере программы и порядковом номере команды в программе: биты 0..7 поля Data определяют номер команды, биты 8,9 поля DATA определяют номер программы.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = Номер программы и команды						Command CMD_PowerSTEP01_ GOTO_PROGRAM_IF_IN0 = 0x2A							Action	Reserve		
Значение	Зависит от значения поля Data						1	0	1	0	1	0	0	0	0	0	0

Битовая раскладка поля Data:

	Байт[3] биты 7..0								Байт[2] биты 7..0								Байт[1] биты 7..2							
Номер бита поля Data	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	0	0	0	0	0	0	Номер программы			Номер команды								

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.44 Исполнительная команда CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN1

Исполнительная команда CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN1 = 0x2B предназначена для перехода к заданной команде заданной программы, если на входе IN1 присутствует сигнал. Поле DATA должно содержать информацию о требуемом номере программы и порядковом номере команды в программе: биты 0..7 поля Data определяют номер команды, биты 8,9 поля DATA определяют номер программы.

Битовая раскладка структуры SMSD_CMD_Type:

Выводная раскладка структуры CMD5B_CMD7_группы:																		
	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Номер программы и команды							Command CMD_PowerSTEP01_ GOTO_PROGRAM_IF_IN1 = 0x2B							Action	Reserve		
Значение	Зависит от значения поля Data							1	0	1	0	1	1	0	0	0	0	

Битовая раскладка поля Data:

	Байт[3] биты 7..0								Байт[2] биты 7..0								Байт[1] биты 7..2							
Номер бита поля Data	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	0	0	0	0	0	0	Номер программы			Номер команды								

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.45 Исполнительная команда CMD_PowerSTEP01_LOOP_PROGRAM

Исполнительная команда CMD_PowerSTEP01_LOOP_PROGRAM = 0x2C используется для создания циклов – контроллер повторяет в цикле заданное число раз заданное количество команд (начиная с команды, следующей за CMD_PowerSTEP01_LOOP_PROGRAM). Поле DATA содержит количество циклов (от 1 до 1023) и количество команд в цикле (от 1 до 1023): 0..9 поля Data – количество команд, биты 10..19 поля Data – количество циклов.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Количество команд и циклов							Command CMD_PowerSTEP01_ LOOP_PROGRAM = 0x2C							Action	Reserve		
Значение	Зависит от значения поля Data							1	0	1	1	0	0	0	0	0	0	

Битовая раскладка поля Data:

	Байт[3] биты 7..0								Байт[2] биты 7..0								Байт[1] биты 7..2							
Номер бита поля Data	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
	0	0	Количество циклов										Количество команд в цикле											

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.46 Исполнительная команда CMD_PowerSTEP01_CALL_PROGRAM

Исполнительная команда CMD_PowerSTEP01_CALL_PROGRAM = 0x2D предназначена для вызова подпрограммы. Поле DATA содержит информацию о номере программы и порядковом номере команды в программе, с которой начинается подпрограмма: биты 0..7 поля DATA – порядковый номер команды, биты 8,9 поля DATA – номер программы. Для возврата в основную программу подпрограмма должна содержать команду возврата - CMD_PowerSTEP01_RETURN_PROGRAM. Подпрограмма выполняется до тех пор, пока не дойдет до команды CMD_PowerSTEP01_RETURN_PROGRAM, затем возвращает управление основной вызвавшей ее программе.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = Номера команды и программы						Command CMD_PowerSTEP01_CALL_PROG RAM = 0x2D								Action	Reserve	
Значение	Зависит от значения поля Data						1	0	1	1	0	1	0	0	0	0	0

Битовая раскладка поля Data:

	Байт[3] биты 7..0								Байт[2] биты 7..0								Байт[1] биты 7..2							
Номер бита поля Data	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	0	0	0	0	0	0	Номер программы			Номер команды								

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.47 Исполнительная команда CMD_PowerSTEP01_RETURN_PROGRAM

Исполнительная команда CMD_PowerSTEP01_RETURN_PROGRAM = 0x2E используется для возврата из подпрограммы в основную программу. Вызов подпрограммы осуществляется командой CMD_PowerSTEP01_CALL_PROGRAM. В случае, если перед вызовом команды возврата CMD_PowerSTEP01_RETURN_PROGRAM не была вызвана подпрограмма, контроллер сгенерирует ошибку.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_ RETURN_PROGRAM = 0x2E							Action	Reserve		
Значение	0						1	0	1	1	1	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.48 Исполнительная команда CMD_PowerSTEP01_START_PROGRAM_MEM0

Исполнительная команда CMD_PowerSTEP01_START_PROGRAM_MEM0 = 0x2F используется для старта программы, записанной в область памяти 0 контроллера.

Команды CMD_PowerSTEP01_START_PROGRAM_MEM1 = 0x30, CMD_PowerSTEP01_START_PROGRAM_MEM2 = 0x31, CMD_PowerSTEP01_START_PROGRAM_MEM3 = 0x32 используются для старта программ, записанных области памяти 1, 2 и 3 соответственно.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	0						Command CMD_PowerSTEP01_ START_PROGRAM_MEM0= 0x2F							Action	Reserve			
Значение	0						1	0	1	1	1	1	0	0	0	0		

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.49 Исполнительная команда CMD_PowerSTEP01_STOP_PROGRAM_MEM

Исполнительная команда CMD_PowerSTEP01_STOP_PROGRAM_MEM = 0x33 используется для остановки выполнения программы.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_ STOP_PROGRAM_MEM = 0x33							Action	Reserve		
Значение	0						1	1	0	0	1	1	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.50 Исполнительная команда CMD_PowerSTEP01_STEP_CLOCK

Исполнительная команда CMD_PowerSTEP01_STEP_CLOCK = 0x34 предназначена для изменения режима управления двигателем на импульсные сигналами EN, STEP, DIR.

Устаревшая команда, не используется в последних версиях контроллера.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	0						Command CMD_PowerSTEP01_ STEP_CLOCK = 0x34							Action	Reserve			
Значение	0						1	1	0	1	0	0	0	0	0	0	0	0

6.51 Исполнительная команда CMD_PowerSTEP01_STOP_USB

Исполнительная команда CMD_PowerSTEP01_STOP_USB = 0x35 предназначена для остановки работы микросхемы USB.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_ STOP_USB = 0x35							Action	Reserve		
Значение	0						1	1	0	1	0	1	0	0	0	0	0

6.52 Исполнительная команда CMD_PowerSTEP01_GET_MIN_SPEED

Исполнительная команда CMD_PowerSTEP01_GET_MIN_SPEED = 0x36 предназначена для чтения текущего значения установленной минимальной скорости. Данная команда добавлена в версию протокола 02. В версии 1 эта команда отсутствует.

Поле DATA пакета содержит структуру SMSD_CMD_Type, поле command которой содержит код команды запроса текущего значения установленной минимальной скорости CMD_PowerSTEP01_GET_MIN_SPEED.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_GET_MIN_ SPEED = 0x36							Action	Reserve		
Значение	0						1	1	0	1	1	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = COMMAND_GET_MIN_SPEED, RETURN_DATA - значение текущей установленной минимальной скорости двигателя.

6.53 Исполнительная команда CMD_PowerSTEP01_GET_MAX_SPEED

Исполнительная команда CMD_PowerSTEP01_GET_MAX_SPEED = 0x37 предназначена для чтения текущего значения установленной максимальной скорости. Данная команда добавлена в версию протокола 02. В версии 1 эта команда отсутствует.

Поле DATA пакета содержит структуру SMSD_CMD_Type, поле command которой содержит код команды запроса текущего значения установленной максимальной скорости CMD_PowerSTEP01_GET_MAX_SPEED.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_GET_MAX_ SPEED = 0x37								Action	Reserve	
Значение	0						1	1	0	1	1	1	1	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = COMMAND_GET_MAX_SPEED, RETURN_DATA - значение текущей установленной максимальной скорости двигателя.

6.54 Исполнительная команда CMD_PowerSTEP01_GET_STACK

Исполнительная команда CMD_PowerSTEP01_GET_STACK = 0x38 предназначена для чтения информации о выполняемой в данный момент программе. Данная команда добавлена в версию протокола 02. В версии 1 эта команда отсутствует.

Поле DATA пакета содержит структуру SMSD_CMD_Type, поле command которой содержит код команды запроса номера исполняемой программы и номер текущей команды CMD_PowerSTEP01_GET_STACK.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_GET_STACK = 0x38								Action	Reserve	
Значение	0						1	1	1	0	0	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = COMMAND_GET_STACK, RETURN_DATA - содержит номер текущей выполняемой команды (первые 8 бит) и номер программы (2 бита).

Битовая раскладка поля RETURN_DATA:

	Байт[3] биты 7..0								Байт[2] биты 7..0								Байт[1] биты 7..2							
Номер бита поля Data	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	0	0	0	0	0	0	Номер программы			Номер команды								

6.55 Исполнительная команда CMD_PowerSTEP01_GOTO_PROGRAM_IF_ZERO

Исполнительная команда CMD_PowerSTEP01_GOTO_PROGRAM_IF_ZERO = 0x39 предназначена для перехода к заданной команде заданной программы, если значение текущей позиции равно 0. Поле DATA должно содержать информацию о требуемом номере программы и порядковом номере команды в программе: биты 0..7 поля Data определяют номер команды, биты 8,9 поля DATA определяют номер программы.

Данная команда добавлена в версию протокола 02. В версии 1 эта команда отсутствует.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Номер программы и команды							Command CMD_PowerSTEP01_GOTO_PROG RAM_IF_ZERO = 0x39							Action	Reserve		
Значение	Зависит от значения поля Data							1	1	1	0	0	1	0	0	0	0	

Битовая раскладка поля Data:

	Байт[3] биты 7..0								Байт[2] биты 7..0								Байт[1] биты 7..2							
Номер бита поля Data	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	0	0	0	0	0	0	Номер программы			Номер команды								

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.56 Исполнительная команда CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN_ZERO

Исполнительная команда CMD_PowerSTEP01_GOTO_PROGRAM_IF_IN_ZERO = 0x3A предназначена для перехода к заданной команде заданной программы, в случае, если на входе SET_ZERO присутствует сигнал. Поле DATA должно содержать информацию о требуемом номере программы и порядковом номере команды в программе: биты 0..7 поля Data определяют номер команды, биты 8,9 поля DATA определяют номер программы.

Данная команда добавлена в версию протокола 02. В версии 1 эта команда отсутствует.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Номер программы и команды							Command CMD_PowerSTEP01_GOTO_PROG RAM_IF_IN_ZERO = 0x3A							Action	Reserve		
Значение	Зависит от значения поля Data							1	1	1	0	1	0	0	0	0	0	

Битовая раскладка поля Data:

	Байт[3] биты 7..0								Байт[2] биты 7..0								Байт[1] биты 7..2							
Номер бита поля Data	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
	0	0	0	0	0	0	0	0	0	0	0	0	Номер программы			Номер команды								

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.57 Исполнительная команда CMD_PowerSTEP01_WAIT_CONTINUE

Исполнительная команда CMD_PowerSTEP01_WAIT_CONTINUE = 0x3B предназначена для ожидания прихода синхросигнала на вход CONTINUE, необходимого для синхронизации программ в нескольких блоках.

Данная команда добавлена в версию протокола 02. В версии 1 эта команда отсутствует.



Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	0						Command CMD_PowerSTEP01_WAIT_CONTI NUE = 0x3B							Action	Reserve		
Значение	0						1	1	1	0	1	1	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.58 Исполнительная команда CMD_PowerSTEP01_SET_WAIT_2

Исполнительная команда CMD_PowerSTEP01_SET_WAIT_2 = 0x3C предназначена для задания паузы. Поле DATA должно содержать время ожидания паузы в диапазоне от 0 до 3600000мс. В отличие от аналогичной команды CMD_PowerSTEP01_SET_WAIT, выполнение данной команды может быть прервано поступлением сигнала на вход IN0, IN1 или SET_ZERO.

Данная команда добавлена в версию протокола 02. В версии 1 эта команда отсутствует.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = время ожидания						Command CMD_PowerSTEP01_ SET_WAIT_2 = 0x3C							Action	Reserve		
Значение	Зависит от значения поля Data						1	0	0	0	1	1	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

6.59 Исполнительная команда CMD_PowerSTEP01_SCAN_MARK2_F

Исполнительная команда CMD_PowerSTEP01_SCAN_MARK2_F = 0x3D предназначена для поиска метки положения в прямом направлении. Движение продолжается до поступления сигнала на вход IN1. При поступлении сигнала двигатель останавливается с заданным значением торможения, текущее положение запоминается как метка. Поле DATA определяет скорость движения при поиске метки.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду. Данная команда добавлена в версию протокола 02. В версии 1 эта команда отсутствует.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]								
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0	
Назначение	Data = Скорость поиска метки							Command CMD_PowerSTEP01_SCAN_MARK 2_F = 0x3D							Action	Reserve		
Значение	Зависит от значения поля Data							1	1	1	0	1	1	0	0	0	0	

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.



6.60 Исполнительная команда CMD_PowerSTEP01_SCAN_MARK2_R

Исполнительная команда CMD_PowerSTEP01_SCAN_MARK2_R = 0x3E предназначена для поиска метки положения в обратном направлении. Движение продолжается до поступления сигнала на вход IN1. При поступлении сигнала двигатель останавливается с заданным значением торможения, текущее положение запоминается как метка. Поле DATA определяет скорость движения при поиске метки.

Внимание: в командах установки скорости единицы измерения параметра - полные шаги в секунду. Данная команда добавлена в версию протокола 02. В версии 1 эта команда отсутствует.

Битовая раскладка структуры SMSD_CMD_Type:

	Байт[3]			Байт[2]			Байт[1]			Байт[0]							
Бит	7	...	0	7	...	0	7..2	1	0	7	6	5	4	3	2	1	0
Назначение	Data = Скорость поиска метки						Command CMD_PowerSTEP01_SCAN_MARK 2_R = 0x3E							Action	Reserve		
Значение	Зависит от значения поля Data						1	1	1	1	1	0	0	0	0	0	0

В ответ контроллер отправляет пакет с кодом команды CMD_TYPE = CODE_CMD_RESPONSE, поле DATA которого содержит структуру COMMANDS_RETURN_DATA_Type: ERROR_OR_COMMAND = OK.

7. Структура SMSD_LAN_Config_Type

Сетевые настройки Контроллера содержатся в структуре SMSD_LAN_Config_Type:

```
typedef struct
{
    uint8_t mac[6];
    uint8_t ip[4];
    uint8_t sn[4];
    uint8_t gw[4];
    uint8_t dns[4];
    uint16_t Port;
    dhcp_mode dhcp;
} SMSD_LAN_Config_Type;
```



Значения по умолчанию:

```
{
    .mac= {0x00, 0xf8, 0xdc, 0x3f, 0x00, 0x00},
    .ip = {192, 168, 1, 2},
    .sn = {255, 255, 0, 0},
    .gw = {192, 168, 1, 1},
    .dns= {0, 0, 0, 0},
    .Port = 5000,
    .dhcp = 1
};
```

8. Отличия между Ethernet и USB в передаче потока данных

Поток данных, передаваемый по физическому каналу USB (имитация COM-port), представляет собой данные Ethernet, в начале и конце которого установлены маркеры начала и конца пакета, а уникальные символы заменены на парные по заданному алгоритму:

0xFA – маркер «начало пакета»

0xFB – маркер «конец пакета»

Если в потоке данных встречаются уникальные байты: 0xFA, 0xFB, 0xFE, они преобразуются в пары байтов 0xFE 0xXX, где 0xXX = байт ^ 0x80

Байт 0xFA внутри пакета замещается парой байтов 0xFE 0x7A.

Байт 0xFB внутри пакета замещается парой байтов 0xFE 0x7B.

Байт 0xFE внутри пакета замещается парой байтов 0xFE 0x7E.

Редакция от 03.06.2024