NSM lab

SMC-5000MA

Контроллер шагового двигателя

руководство пользователя

ОСНОВНЫЕ ХАРАКТЕРИСТИКИ

- *управление биполярными шаговыми двигателями*
- *максимальное напряжение питания – 35 В*
- *максимальный ток питания двигателя – 2.5 А*
- *ШИМ-стабилизация тока обмоток двигателя*
- *поддержка микрошагового режима до 1/8 шага*
- *скорость вращения от 1 до 32 000 микрошагов/сек*
- *подключение двух концевых выключателей (NO или NC)*
- *программное разрешение/запрещение концевых выключателей*
- *подключение датчика базового положения*
- *возможность подключения квадратурного энкодера*
- *разгон с заданным ускорением*
- *программируемый ток фаз для разгона, движения и удержания*
- *поддержка медленной, быстрой и смешанной скорости спада тока*
- *перемещение на абсолютную координату или на заданное число шагов*
- *абсолютная координата ±2 000 000 000 микрошагов*
- *чтение и установка абсолютной координаты*
- *задание скорости начала разгона*
- *сохранение параметров в энергонезависимой памяти*
- *местное управление с помощью кнопок*
- *программируемые функции кнопок (вращение/шаг/скорость)*
- *возможность управления с помощью логических сигналов*
- *светодиодная индикация режимов работы*
- *контроль напряжения питания*
- *количество пользовательских каналов цифрового ввода-вывода – 8*
- *тип цифровых выходов – открытый сток*
- *выходной ток цифрового выхода – до 100 мА*
- *напряжение на цифровых выходах – до 30 В*
- *пороговое напряжение цифровых входов – 1.7 В*
- *управление по интерфейсу RS-485*
- *включение в цепочку по RS-485 до 64 контроллеров*
- *протокол обмена – WAKE*
- *один источник питания +10…35 В*
- *тестовое ПО (Win98/ME/2000/XP)*
- *библиотека функций управления (DLL)*

ОПИСАНИЕ УСТРОЙСТВА

Контроллер шагового двигателя SMC-5000MA предназначен для работы с биполярными шаговыми двигателями. Основой устройства является микросхема A3977 фирмы «Allegro», которая содержит транслятор уровней, два ЦАП для задания тока фаз с помощью ШИМ, а также два мостовых выходных каскада на полевых транзисторах. Управление устройством обеспечивает микроконтроллер ATmega16 фирмы «Atmel». Связь с компьютером осуществляется по интерфейсу RS-485 через переходник RS-232 – RS-485 или USB – RS-485. Скорость обмена фиксирована и равна 19200 бод. Контроллер имеет дополнительный разъем

RS-485, который позволяет организовывать сеть контроллеров (до 64 контроллеров). Вместе с контроллером поставляется библиотека в виде DLL, которая содержит все необходимые для управления устройством функции, а также тестовое ПО. Функции DLL могут быть вызваны из среды LabVIEW или из программы пользователя, написанной на любом языке программирования.

НАЗНАЧЕНИЕ КОНТАКТОВ РАЗЪЕМОВ

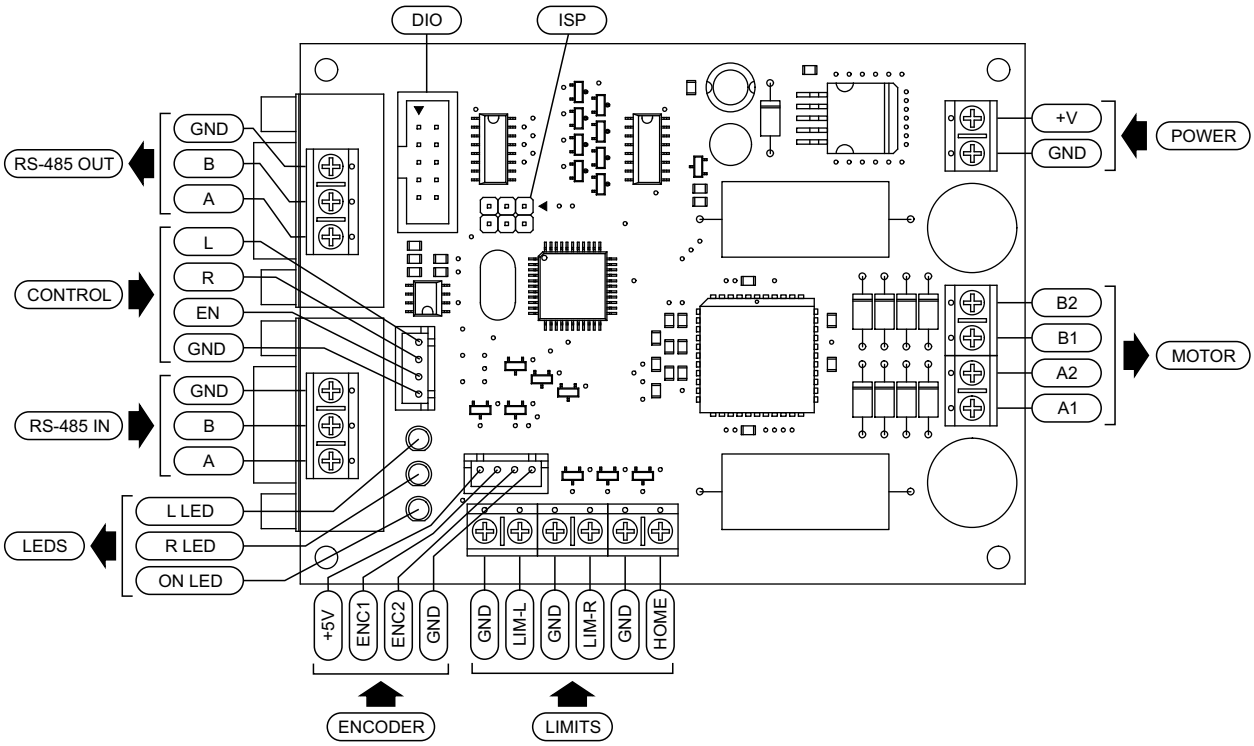

Рис. 1. Назначение контактов разъемов.

Для подключения шагового двигателя плата имеет группу из 4-х клеммников «MOTOR». Контроллер рассчитан на подключение биполярного шагового двигателя с двумя обмотками. Клеммы A1 и A2 служат для подключения начала и конца обмотки фазы A, клеммы B1 и B2 – начала и конца обмотки фазы B. Униполярные шаговые двигатели с 4-мя раздельными обмотками (например, ДШИ-200) могут быть подключены в биполярном режиме, для этого обмотки должны быть соединены попарно параллельно.

Напряжение питания подключается с помощью группы из 2-х клеммников «POWER». Напряжение питания может лежать в пределах 10…35 В. Контроллер обеспечивает ШИМ-стабилизацию тока в обмотках двигателя, поэтому напряжение питания должно быть не менее номинального напряжения питания используемого двигателя.

Для подключения концевых выключателей служит группа из 6-ти клеммников «LIMITS». Концевой выключатель, ограничивающий движение шагового привода назад, подключается к клеммам LIM-L и GND. Концевой выключатель, ограничивающий движение шагового привода вперед – к клеммам LIM-R и GND. Могут использоваться концевые

выключатели как с нормально-замкнутыми, так и с нормально-разомкнутыми контактами. Кроме концевых выключателей к этой же группе клеммников можно подключить датчик базового положения привода. Для этого предназначены клеммы HOME и GND. Входы концевых выключателей и датчика базового положения имеют подтягивающие резисторы на +5 В. Вместо механических выключателей допускается применение электронных датчиков положения (магнитных, оптических) с выходом в формате ТТЛ.

Контроллер поддерживает работу с квадратурным энкодером, который может служить датчиком положения привода. Для подключения энкодера служит разъем «ENCODER». На разъем поступает напряжение питания +5 В, максимально допустимый ток – до 25 мА. Квадратурные сигналы с выхода энкодера подаются на контакты ENC1 и ENC2. Эти сигналы должны быть в формате ТТЛ. Входы для подключения энкодера имеют подтягивающие резисторы на +5 В.

Местное управление контроллером осуществляется через разъем «CONTROL». Этот разъем имеет контакт разрешения работы двигателя (EN), задания направления вперед (R) и задания направления назад (L). Все сигналы имеют подтягивающие резисторы на +5 В. Активный уровень всех сигналов может выбираться программно. Этот же разъем может быть использован для управления контроллером с помощью внешних сигналов, которые должны иметь формат ТТЛ. Функции сигналов управления могут программироваться с помощью специальной команды, поступающей от компьютера (см. описание команд ниже).

Для управления контроллером от компьютера плата имеет два разъема интерфейса RS-485. Оба разъема равнозначны, один из них может использоваться как вход, другой – как выход. Благодаря этому контроллеры можно объединять в цепочку (до 64 штук), которая через преобразователь интерфейсов может быть подключена к одному COM-порту (или USB-порту) компьютера. Используются разъемы типа D-SUB-9F (розетки), назначение контактов показано в таблице 1.

Таблица 1. Назначение контактов разъемов RS-485.

Номер контакта	Назначение
2	сигнал «A» интерфейса RS-485
3	сигнал «B» интерфейса RS-485
5	общий

Каждый из разъемов RS-485 на печатной плате контроллера продублирован группой из 3-х клеммников. В случае необходимости, клеммники могут быть установлены вместо разъемов. Назначение клемм приведено на рис. 1.

Разъем «ISP» предназначен для программирования управляющего микроконтроллера на этапе производства и в процессе эксплуатации не используется.

Контроллер имеет в своем составе порт цифрового ввода-вывода, который может использоваться для считывания состояния различных датчиков или управления исполнительными механизмами. Порт имеет 8 двунаправленных линий, каждую из которых можно отдельно использовать на ввод или на вывод. Порт использует 10-контактный разъем «DIO» типа IDC-10. Назначение контактов разъема показано в таблице 2.

Таблица 2. Назначение контактов разъема DIO.

Номер контакта	Назначение
1	вход-выход D0
2	вход-выход D1
3	вход-выход D2
4	вход-выход D3
5	вход-выход D4
6	вход-выход D5
7	вход-выход D6
8	вход-выход D7
9	общий
10	общий

РЕЖИМЫ РАБОТЫ

Контроллер обеспечивает работу биполярных шаговых двигателей в полношаговом (FS, full step), полушаговом (HS, half step), а также микрошаговом режиме. В микрошаговом режиме возможно дробление шага на 4 (FS/4) или на 8 (FS/8). Выбор нужного режима работы осуществляется с помощью команды C_SetMode.

Разные режимы работы обеспечиваются путем задания требуемых соотношений токов фаз для каждого шага (или микрошага). Таблица используемых значений тока приведена в приложении 1.

Все значения координат, скоростей и ускорений задаются с использованием микрошага FS/8. При работе с более крупным шагом контроллер способен отрабатывать значения, кратные 2-м, 4-м или 8-ми.

СКОРОСТЬ И УСКОРЕНИЕ

Контроллер позволяет устанавливать скорость от 0 до 32000 микрошагов в секунду (1 микрошаг = 1/8 полного шага). При работе контроллера в режиме FS/8 это значение непосредственно определяет скорость в микрошагах в секунду. Если контроллер переключить в режим FS/4, реальное число шагов в секунду будет вдвое меньше, при этом частота вращения вала двигателя останется той же. Аналогично и для режимов HS и FS.

Контроллер обеспечивает разгон с постоянным ускорением (трапециидальный профиль скорости). Если задать значение ускорения 0, то вместо трапециидального профиля скорости двигатель будет иметь прямоугольный профиль, т.е. двигатель сразу будет переходить на заданную скорость. Однако при этом возможна потеря координаты ввиду пропуска шагов и уменьшение максимальной рабочей скорости. Использовать прямоугольный профиль скорости допустимо только при работе на малых скоростях.

СКОРОСТЬ НАЧАЛА РАЗГОНА

Контроллер позволяет задавать скорость начала разгона. Задание скорости начала разгона позволяет реализовать для двигателя смешанный профиль скорости. Если двигатель

начинает вращение с нулевой скорости, то скорость скачком достигает скорости начала разгона, а дальше двигатель начинает разгоняться с заданным ускорением. При торможении все происходит наоборот: двигатель выполняет торможение с заданным ускорением, пока скорость не снизится до скорости начала разгона. Затем двигатель сразу останавливается. Задание минимальной скорости разгона обычно используется для уменьшения вибраций при разгоне и торможении, которые имеют место при работе двигателя на низких скоростях. Минимальная скорость не накладывает ограничение снизу на устанавливаемую скорость. Просто для скоростей, меньших скорости начала разгона, будет реализован прямоугольный профиль скорости.

УПРАВЛЕНИЕ РАБОЧИМ ТОКОМ

Контроллер обеспечивает ШИМ-стабилизацию тока обмоток двигателя. Поэтому ток не зависит от напряжения питания и определяется только заданным значением. Значение тока задается программно. Контроллер позволяет индивидуально задавать рабочий ток, ток разгона и ток удержания. Рабочий ток включается при вращении двигателя с постоянной скоростью (когда ускорение равно нулю). Во время разгона и торможения (когда ускорение не равно нулю) включается ток разгона. Требуемое значение тока разгона обычно большее, чем значение рабочего тока. Когда двигатель остановлен (скорость равна нулю), включается ток удержания. Требуемое значение тока удержания обычно меньше, чем значение рабочего тока. Ток удержания необходим для того, чтобы при остановленном двигателе не потерять текущую координату. Особенно это актуально при работе в полушаговом или микрошаговом режиме, когда при обесточивании двигателя координата теряется (ротор двигателя поворачивается на некоторый угол, чтобы достичь положения равновесия).

ПОЗИЦИОНИРОВАНИЕ

Шаговые двигатели позволяют осуществить позиционирование без применения датчиков положения и обратной связи. Контроллер SMC-5000MA поддерживает обработку относительной и абсолютной координаты. Значение координаты всегда выражается в микрошагах FS/8. Поэтому при работе с более крупным шагом возможно лишь более грубое позиционирование. Контроллер имеет счетчик абсолютной координаты емкостью ±2 000 000 000 микрошагов. При выполнении команд позиционирования возможно перемещение на абсолютную или относительную координату. При переходе на абсолютную координату начинается вращение двигателя в требуемом направлении (направление вычисляется исходя из разности текущей и требуемой координаты), которое завершается при достижении требуемой координаты. При переходе на относительную координату начинается вращение двигателя в направлении, определяемом знаком заданной относительной координаты, которое завершается после осуществления заданного количества шагов.

При позиционировании осуществляется разгон и торможение согласно заданным значениям скорости, скорости начала разгона и ускорения. Для предотвращения потери координаты в режиме позиционирования должен быть включен ток удержания.

БАЗОВАЯ ПОЗИЦИЯ

Контроллер позволяет подключить датчик базовой позиции. В качестве датчика может быть оптопара, датчик Холла или обычный механический выключатель. Программно можно выбрать желаемый активный уровень и действие при достижении базовой позиции: плавное торможение или мгновенная остановка двигателя. Программное обеспечение контроллера содержит специальную функцию базирования, которая позволяет определить координату базовой позиции в микрошагах.

Для выполнения процедуры базирования необходимо выполнить команду C_Home. В качестве параметра эта команда требует направление поиска базы. Во время базирования разгон и торможение осуществляется согласно заданным значениям скорости, скорости начала разгона и ускорения. При срабатывании датчика базовой позиции двигатель остановится, а координата базовой позиции в микрошагах будет сохранена в контроллере и может быть считана. Если считанную координату базовой позиции потом вычесть из текущей позиции, то получим координату относительно базы. Выполнение процедуры базирования можно контролировать с помощью команды C_GetStat. Эта команда позволяет узнать, успешно ли завершилась процедура, или она была прервана, например, срабатыванием концевого выключателя. Для повышения точности базирования грубый поиск базы можно делать с высокой скоростью и с плавным разгоном и торможением. После поиска базы команду C_Home можно выполнить еще раз, поменяв направление поиска, снизив скорость и изменив режим работы датчика базового положения на остановку без ускорения. После этого двигатель будет остановлен точно в базовой позиции.

КОНЦЕВЫЕ ВЫКЛЮЧАТЕЛИ

Контроллер позволяет подключить 2 концевых выключателя. Включение и отключение обработки концевых выключателей осуществляется с помощью команды C_SetLim. Дополнительно можно задать тип концевых выключателей – нормально-разомкнутые или нормально-замкнутые.

При срабатывании одного из концевых выключателей двигатель выполняет торможение с заданным ускорением (или мгновенную остановку, если задан такой режим работы), и движение в данном направлении прекращается. Дальнейшее движение возможно только в обратном направлении.

Если одновременно сработали оба концевых выключателя, то движение становится невозможным ни в одном из направлений. Такая ситуация должна быть исключена конструктивно.

Вместо механических концевых выключателей можно использовать любые датчики положения с выходными сигналами в формате ТТЛ. Такие датчики могут быть непосредственно подключены к клеммам «LIMITS».

Нужно отметить, что в случае использования плавного торможения на концевых выключателях, необходимо обеспечить возможность достаточного выбега двигателя в зоне срабатывания концевого выключателя (т.е. после срабатывания концевого выключателя двигатель еще некоторое время вращается, это не должно приводить к механическим поломкам).

УПРАВЛЕНИЕ С МЕСТНОЙ КЛАВИАТУРЫ

К разъему «CONTROL» может быть подключена клавиатура местного управления. В стандартной конфигурации клавиатура содержит одну кнопку с фиксацией и две кнопки без фиксации. Кнопка с фиксацией подает сигнал низкого уровня на контакт EN и выполняет функцию разрешения работы двигателя «EN». Две других кнопки «L» и «R» подают сигналы низкого уровня на контакты R и L соответственно.

Всего имеется 7 режимов работы местного управления:

1. Местное управление выключено. При этом двигатель не реагирует ни на одну из кнопок.
2. Кнопка «L» включает вращение назад, кнопка «R» включает вращение вперед. Движение происходит согласно заданным значениям скорости, скорости начала разгона и ускорения.
3. Кнопка «L» включает вращение назад, кнопка «R» включает вращение вперед. Движение происходит на минимальной скорости.
4. Кнопка «L» осуществляет один шаг двигателя, кнопка «R» определяет направление шага: нажатое состояние соответствует направлению вперед, отжатое – назад. Для предотвращения потери координаты следует использовать ток удержания.
5. Кнопка «L» осуществляет один шаг двигателя назад, кнопка «R» – один шаг вперед. Для предотвращения потери координаты следует использовать ток удержания.
6. Вращение двигателя вперед, кнопка «L» осуществляет уменьшение скорости, кнопка «R» – увеличение.
7. Вращение двигателя назад, кнопка «L» осуществляет уменьшение скорости, кнопка «R» – увеличение.

Режим работы местного управления задается с компьютера и может быть записан в энергонезависимую память контроллера. При включении питания режим восстанавливается, как и все параметры движения: скорость, скорость начала разгона, ускорение.

При управлении контроллером от кнопок осуществляется подавление дребезга с постоянной времени 20 мс. При управлении логическими сигналами подавление дребезга может быть программно отключено. При отключенном подавлении дребезга минимальная длительность сигналов управления составляет 1 мс, а минимальный период – 100 мс. Активный уровень управляющих сигналов может выбираться программно.

ИНДИКАЦИЯ РЕЖИМОВ РАБОТЫ

На печатной плате контроллера расположены три светодиода. Зеленый светодиод «ON» индицирует наличие напряжения питания.

Красные светодиоды «L» и «R» индицируют состояние концевых выключателей и направление вращения двигателя. Если сработал концевой выключатель, соответствующий светодиод мигает. Сработавшим считается выключатель, который обеспечивает на входе контроллера тот логический уровень, который запрограммирован в данный момент как

активный. Если концевой выключатель программно отключен, индикация не производится. При вращении двигателя в одном из направлений соответствующий светодиод горит постоянно.

ЦИФРОВОЙ ВВОД-ВЫВОД

Контроллер имеет 8 линий цифрового ввода-вывода. Каждая линия ввода-вывода является двунаправленной и может использоваться как для ввода, так и для вывода цифровой информации. Схемотехника одного из каналов цифрового ввода-вывода показана на рис. 2.

Цифровые выходы с открытым стоком обеспечивают выходной ток до 100 мА на канал при напряжении до 30 В. Встроенная схема защиты ограничивает ток каждого выхода на уровне 250 мА. Для того чтобы настроить любой из каналов на ввод, достаточно просто записать в этот канал единицу. При этом выходной транзистор будет закрыт, и состояние линии ввода будет полностью определяться внешним сигналом. Пороговое напряжение линий цифрового ввода составляет примерно 1.7 В.

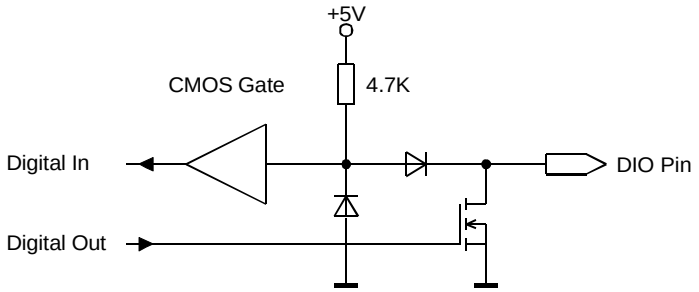

Рис. 2. Схемотехника каналов цифрового ввода-вывода.

ОБРАБОТКА ЭНКОДЕРА

Контроллер SMC-5000MA поддерживает работу с квадратурным энкодером, который может служить датчиком положения привода. В контроллере реализован 1x-квадратурный декодер, который увеличивает или уменьшает координату энкодера на единицу при повороте вала энкодера на один полный период повторения выходных сигналов. Декодер использует промежуточные состояния энкодера для подавления дребезга.

Контроллер имеет счетчик абсолютной координаты энкодера емкостью ±2 000 000 000 единиц, который можно считать или загрузить начальным значением с помощью специальных команд. При выполнении команд позиционирования возможно перемещение на абсолютную или относительную координату энкодера. При переходе на абсолютную координату начинается вращение двигателя в требуемом направлении (направление вычисляется исходя из разности текущей и требуемой координаты), которое завершается при достижении требуемой координаты. При переходе на относительную координату энкодера начинается вращение двигателя в направлении, определяемом знаком заданной относительной координаты, которое завершается после осуществления заданного количества шагов энкодера. При позиционировании по энкодеру старт и остановка двигателя всегда производятся без ускорения.

АДРЕСАЦИЯ КОНТРОЛЛЕРОВ

Контроллеры можно объединять в сеть, используя дополнительный разъем RS-485. Перед объединением контроллеров в сеть каждому из них должен быть присвоен уникальный адрес, который сохраняется в энергонезависимой памяти. Сделать это можно с помощью команды C_SetAddr. Присваивать адреса контроллерам нужно по очереди, по одному подключая их к компьютеру. Подробное описание команды C_SetAddr приведено ниже.

КОМАНДЫ КОНТРОЛЛЕРА ШАГОВОГО ДВИГАТЕЛЯ.

Команды передаются в виде пакетов согласно протоколу WAKE. Инициатором обмена всегда выступает PC. В ответ на каждую команду контроллер передает пакет, который содержит тот же номер команды, а в качестве первого байта данных передается код ошибки (за исключением команд C_Echo и C_Info). Код ошибки 00h означает успешное выполнение команд, любой отличный код – наличие ошибки (см. описание кодов ошибок). В поле данных каждой команды передаются параметры. Для разных команд число параметров разное, некоторые команды могут не иметь параметров вообще.

C_Nop – нет операции. Используется для внутренних целей и никогда не передается в контроллер или PC.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
00h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
00h	0	-							

C_Err – контроллер передает эту команду в PC в качестве ответа на любую команду, если произошла ошибка приема пакета. Для этой команды Error Code всегда равен Err_Tx.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
01h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
01h	1	Error Code							

C_Echo – команда запроса возврата пакета. Пакет может содержать до 32 байт произвольных данных. В ответ на эту команду контроллер передает пакет в неизменном виде обратно. Команда используется для проверки связи с контроллером.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
02h	X	Byte1							
		…							
		ByteN							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
02h	X	Byte1							
		…							
		ByteN							

C_Info – запрос информации о типе, версии firmware и серийном номере контроллера. В ответ передается пакет, содержащий 16 байт данных, которые представляют собой строку в коде ASCII: SMC-5000MA V1.0, где SMC-5000MA – тип устройства, V1.0 – версия firmware 1.0. В качестве разделителей используются пробелы (код 20h). Строка заканчивается байтом 00h.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
03h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
03h	16	String: “SMC-5000MA V1.0”, 00h							

C_SetAddr – запись адреса подчиненного устройства.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
04h	3	Signature (low byte)							
		Signature (high byte)							
		Address							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
04h	1	Error Code							

Параметр Signature представляет собой ключ, который необходимо передать устройству для получения прав изменения адреса. Ключ является неизменным и равен BEDAh.

Параметр Address представляет собой значение адреса подчиненного устройства (контроллера) в сети. Адрес может принимать значения 0…127. Значение адреса, равное 0, совпадает с адресом для коллективного вызова и может использоваться лишь в случае наличия всего одного подчиненного устройства в сети. Если подключено несколько устройств, адрес может принимать значения 1…127. Каждое устройство должно иметь уникальный адрес, иначе передаваемые по сети данные будут искажены. Для того чтобы назначить устройству адрес, устройство нужно подключить отдельно, т.е. без других устройств в сети, и выполнить команду C_SetAddr. При этом обращаться к устройству можно по ранее заданному адресу или по адресу 0, который является адресом коллективного вызова. В последнем случае новый адрес будет установлен, если даже заданный ранее адрес неизвестен.

Команда сохраняет переданный адрес в энергонезависимой памяти, что требует дополнительно 10 мс.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды, Err_Pa – в случае неверного значения параметров.

C_GetAddr – чтение адреса подчиненного устройства.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
05h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
05h	2	Error Code							
		Address							

Команда всегда возвращает код ошибки Error Code, равный Err_No.

Команда возвращает значение адреса подчиненного устройства (модуля) Address. Для того чтобы узнать неизвестный адрес устройства, его нужно подключить отдельно, т.е. без других устройств в сети и выполнить команду C_GetAddr. Обращаться к устройству нужно по адресу 0, который является адресом коллективного вызова.

C_SetDio – управление выходами.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
06h	1	O7	O6	O5	O4	O3	O2	O1	O0

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
06h	1	Error Code							

Команда имеет параметр, каждый бит которого отвечает за одну из линий цифрового вывода DIO0…DIO7. Значение каждого бита устанавливается на линиях DIO.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

C_GetDio – чтение входов.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
07h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
07h	2	Error Code							
		I7	I6	I5	I4	I3	I2	I1	I0

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

Команда возвращает байт, каждый бит которого отвечает за одну из линий цифрового ввода DIO0…DIO7. Значение каждого бита считывается с линий DIO.

C_GetAdc – чтение напряжения питания.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
08h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
08h	3	Error Code							
		Adc (low byte)							
		Adc (high byte)							

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

Команда возвращает значение напряжения питания контроллера в виде 2-х байтового числа без знака, которое может принимать значения 0…400, что соответствует напряжению питания 0…40.0 В.

C_SetMode – установка параметров движения.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
09h	2	MS							
		PFD							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
09h	1	Error Code							

Параметр MS определяет величину шага:

- MS = 1 – полношаговый режим (с перекрытием фаз)
- MS = 2 – полушаговый режим (деление шага на 2)
- MS = 4 – микрошаговый режим с делением нага на 4
- MS = 8 – микрошаговый режим с делением нага на 8

Параметр PFD определяет скорость спада тока в обмотках двигателя:

- PFD = 0 – медленный спад тока
- PFD = 1 – быстрый спад тока
- PFD = 2 – смешанный спад тока

При работе в полношаговом режиме рекомендуется использовать медленный спад тока. В микрошаговом режиме возможно использование смешанного спада тока, что улучшает динамическую точность позиционирования. Быстрый спад тока можно использовать для улучшения динамических характеристик двигателя на высоких скоростях, но при этом увеличивается нагрев двигателя и ухудшается точность позиционирования.

Переключение параметров движения может производиться как в состоянии останова, так и при вращении двигателя. Новые параметры движения вступают в силу при выполнении следующего шага после выполнения команды.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального завершения команды и Err_Pa – в случае неверного значения параметров.

C_SetIh – задание тока удержания.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Ah	2	Ih (low byte)							
		Ih (high byte)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Ah	1	Error Code							

Команда имеет параметр Ih, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 0…2500 мА. Значение тока Ih используется только в режиме удержания (когда скорость равна нулю).

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Pa – в случае неверного значения параметра.

C_SetIa – задание тока разгона и торможения.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Bh	2	Ia (low byte)							
		Ia (high byte)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Bh	1	Error Code							

Команда имеет параметр Ia, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 0…2500 мА. При работе двигателя с разными параметрами шага ток фаз зависит от угла поворота ротора (см. таблицу в приложении 1). Значение тока Ia используется только при разгоне и торможении двигателя (когда ускорение не равно нулю).

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Pa – в случае неверного значения параметра.

C_SetIw – задание рабочего тока.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Ch	2	Iw (low byte)							
		Iw (high byte)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Ch	1	Error Code							

Команда имеет параметр Iw, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 0…2500 мА. При работе двигателя с

разными параметрами шага ток фаз зависит от угла поворота ротора (см. таблицу в приложении 1). Значение тока Iw используется только при вращении двигателя с постоянной скоростью (когда ускорение равно нулю).

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Pa – в случае неверного значения параметра.

C_GetIc – чтение тока двигателя.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Dh	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Dh	3	Error Code							
		Ic (low byte)							
		Ic (high byte)							

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

Команда возвращает значение текущего тока двигателя Ic в виде 2-х байтового числа без знака, которое может принимать значения 0…2500 мА.

C_SetAw – задание ускорения.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Eh	2	Aw (low byte)							
		Aw (high byte)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Eh	1	Error Code							

Команда имеет параметр Aw, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 0…32000 микрошагов/сек^2 (1 микрошаг = 1/8 полного шага). Изменение ускорения можно производить как в состоянии останова, так и при вращении двигателя. При задании Aw = 0 вместо трапециидального профиля скорости двигатель будет иметь прямоугольный профиль, т.е. двигатель сразу будет переходить на заданную скорость.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Pa – в случае неверного значения параметра.

C_GetAc – чтение текущего ускорения.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Fh	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
0Fh	3	Error Code							
		Ac (low byte)							
		Ac (high byte)							

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

Команда возвращает значение текущего ускорения Ac в виде 2-х байтового числа со знаком, которое может принимать значения ±32000 микрошагов/сек^2 (1 микрошаг = 1/8 полного шага). Если двигатель остановлен или вращается с постоянной скоростью, команда возвращает 0.

C_SetVm – задание скорости начала разгона.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
10h	4	Vm (low byte)							
		Vm (high byte)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
10h	1	Error Code							

Команда имеет параметр Vm, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 1…32000 микрошагов/сек (1 микрошаг = 1/8 полного шага). При разгоне двигатель скачком перейдет на скорость Vm, а затем будет выполнять разгон до скорости V. Изменение скорости начала разгона можно производить как в состоянии останова, так и при вращении двигателя.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Pa – в случае неверного значения параметра.

C_SetVw – задание рабочей скорости.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
11h	2	Vw (low byte)							
		Vw (high byte)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
11h	1	Error Code							

Команда имеет параметр Vw, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 0…32000 микрошагов/сек (1 микрошаг = 1/8 полного шага). Изменение скорости можно производить как в состоянии останова, так и при вращении двигателя. Если команда выполняется в состоянии останова,

двигатель продолжает находиться в состоянии останова до выполнения одной из команд старта.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального завершения команды и Err_Pa – в случае неверного значения параметра.

C_GetVc – чтение текущей скорости.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
12h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
12h	3	Error Code							
		Vc (low byte)							
		Vc (high byte)							

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

Команда возвращает значение текущей скорости Vc в виде 2-х байтового числа со знаком, которое может принимать значения ±32000 микрошагов/сек (1 микрошаг = 1/8 полного шага).

C_SetNc – установка текущей координаты.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
13h	4	Nc (byte 1)							
		Nc (byte 2)							
		Nc (byte 3)							
		Nc (byte 4)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
13h	1	Error Code							

Команда имеет параметр Nc, который представляет собой четырехбайтовое знаковое число (байт 1 – младший, байт 4 – старший). Параметр может принимать значения ±2000000000 и определяет координату в микрошагах (1 микрошаг = 1/8 полного шага). Обычно команда используется для сброса (обнуления) текущей координаты.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Pa – в случае неверного значения параметра.

C_GetNc – чтение текущей координаты.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
14h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
14h	5	Error Code							
		Nc (byte 1)							
		Nc (byte 2)							
		Nc (byte 3)							
		Nc (byte 4)							

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

Команда возвращает значение текущей координаты Nc в виде 4-х байтового числа со знаком, которое может принимать значения ±2000000000 и представляет собой координату в микрошагах (1 микрошаг = 1/8 полного шага).

C_GetNb – чтение координаты базовой позиции (HOME).

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
15h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
15h	5	Error Code							
		Nb (byte 1)							
		Nb (byte 2)							
		Nb (byte 3)							
		Nb (byte 4)							

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

Команда возвращает значение координаты базовой позиции Nb в виде 4-х байтового числа со знаком (байт 1 – младший, байт 4 – старший), которое может принимать значения ±2000000000 и представляет собой координату базовой позиции (HOME) в микрошагах (1 микрошаг = 1/8 полного шага). Эта координата сохраняется контроллером в тот момент, когда срабатывает датчик базовой позиции.

C_SetNe – установка текущей координаты энкодера.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
16h	4	Ne (byte 1)							
		Ne (byte 2)							
		Ne (byte 3)							
		Ne (byte 4)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
16h	1	Error Code							

Команда имеет параметр Ne, который представляет собой четырехбайтовое знаковое число (байт 1 – младший, байт 4 – старший). Параметр может принимать значения от ±2000000000 и определяет координату в шагах энкодера. Обычно команда используется для сброса (обнуления) координаты энкодера.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Pa – в случае неверного значения параметра.

C_GetNe – чтение текущей координаты энкодера.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
17h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
17h	5	Error Code							
		Ne (byte 1)							
		Ne (byte 2)							
		Ne (byte 3)							
		Ne (byte 4)							

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

Команда возвращает значение текущей координаты энкодера Ne в виде 4-х байтового числа со знаком, которое может принимать значения ±2000000000 и представляет собой координату в шагах энкодера.

C_StartV – старт двигателя на заданной скорости.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
18h	2	V (low byte)							
		V (high byte)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
18h	1	Error Code							

Команда имеет параметр V, который представляет собой двухбайтовое число со знаком. Параметр может принимать значения ±32000 микрошагов/сек (1 микрошаг = 1/8 полного шага). При выполнении команды двигатель начинает вращаться с заданным ускорением, пока не достигнет скорости V. Направление вращения определяется знаком скорости, положительные значения соответствуют движению вперед (или вправо – R), отрицательные – движению назад (или влево – L). С нулевым значением V команда может использоваться для остановки двигателя с заданным ускорением.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального завершения команды и Err_Pa – в случае неверного значения параметра.

C_StartD – старт двигателя в заданном направлении.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
19h	1	Dir							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
19h	1	Error Code							

Команда имеет параметр Dir, который представляет собой число со знаком. При выполнении команды двигатель начинает вращаться с заданным ускорением, пока не достигнет рабочей скорости Vw. Направление вращения определяется знаком параметра Dir, любые положительные значения соответствуют движению вперед (или вправо – R), любые отрицательные – движению назад (или влево – L). С нулевым значением Dir команда может использоваться для остановки двигателя с заданным ускорением.

Команда возвращает код ошибки Error Code, который всегда равен Err_No.

C_StartN – старт двигателя на заданную абсолютную координату.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Ah	4	N (byte 1)							
		N (byte 2)							
		N (byte 3)							
		N (byte 4)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Ah	1	Error Code							

Команда имеет параметр N, который представляет собой четырехбайтовое знаковое число (байт 1 – младший, байт 4 – старший). Параметр может принимать значения ±2000000000 и определяет координату в микрошагах (1 микрошаг = 1/8 полного шага), на которую выполнит перемещение двигатель. Направление вращения определяется значениями текущей и заданной координат: если текущая координата больше заданной, то двигатель будет выполнять вращение назад, если меньше заданной – вперед. После выполнения перемещения на заданную координату двигатель останавливается. Разгон и торможение двигателя производится с заданным ускорением. Двигатель останавливается также при выполнении команды C_StartV или C_StartD с нулевым значением параметра, C_Stop или при срабатывании соответствующего концевого выключателя (если концевые выключатели программно разрешены). Определить момент завершения команды можно с помощью команды C_GetStat.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Pa – в случае неверного значения параметра.

C_StartdN – старт двигателя на заданную относительную координату.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Bh	4	dN (byte 1)							
		dN (byte 2)							
		dN (byte 3)							
		dN (byte 4)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Bh	1	Error Code							

Команда имеет параметр dN, который представляет собой четырехбайтовое знаковое число (байт 1 – младший, байт 4 – старший). Параметр может принимать значения ±2000000000 и определяет, на какое количество микрошагов (1 микрошаг = 1/8 полного шага) выполнит перемещение двигатель. Направление вращения определяется знаком параметра: отрицательным значениям соответствует вращение назад, положительным – вперед. После выполнения перемещения на заданное количество микрошагов двигатель останавливается. Разгон и торможение двигателя производится с заданным ускорением. Двигатель останавливается также при выполнении команды C_StartV или C_StartD с нулевым значением параметра, C_Stop или при срабатывании соответствующего концевого выключателя (если концевые выключатели программно разрешены). Определить момент завершения команды можно с помощью команды C_GetStat.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Pa – в случае неверного значения параметра.

C_StartNe – старт двигателя на заданную абсолютную координату энкодера.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Ch	4	Ne (byte 1)							
		Ne (byte 2)							
		Ne (byte 3)							
		Ne (byte 4)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Ch	1	Error Code							

Команда имеет параметр N, который представляет собой четырехбайтовое знаковое число (байт 1 – младший, байт 4 – старший). Параметр может принимать значения ±2000000000 и определяет координату в шагах энкодера, на которую выполнит перемещение двигатель. Направление вращения определяется значениями текущей и заданной координат: если текущая координата больше заданной, то двигатель будет выполнять вращение назад, если меньше заданной – вперед. После выполнения перемещения на заданную координату двигатель останавливается. Старт и остановка двигателя производится без ускорения. Двигатель останавливается также при выполнении команды C_StartV или C_StartD с нулевым значением параметра, C_Stop или при срабатывании соответствующего концевого выключателя (если концевые выключатели программно разрешены). Определить момент завершения команды можно с помощью команды C_GetStat.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Pa – в случае неверного значения параметра.

C_StartdNe – старт двигателя на заданную относительную координату энкодера.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Dh	4	dNe (byte 1)							
		dNe (byte 2)							
		dNe (byte 3)							
		dNe (byte 4)							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Dh	1	Error Code							

Команда имеет параметр dNe, который представляет собой четырехбайтовое знаковое число (байт 1 – младший, байт 4 – старший). Параметр может принимать значения ±2000000000 и определяет, на какое количество шагов энкодера выполнит перемещение двигатель. Направление вращения определяется знаком параметра: отрицательным значениям соответствует вращение назад, положительным – вперед. После выполнения перемещения на заданное количество шагов энкодера двигатель останавливается. Старт и остановка двигателя производится без ускорения. Двигатель останавливается также при выполнении команды C_StartV или C_StartD с нулевым значением параметра, C_Stop или при срабатывании соответствующего концевого

выключателя (если концевые выключатели программно разрешены). Определить момент завершения команды можно с помощью команды C_GetStat.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Pa – в случае неверного значения параметра.

C_Stop – остановка двигателя.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Eh	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Eh	1	Error Code							

Команда не имеет параметров. При выполнении команды двигатель останавливается без ускорения. Для остановки двигателя с заданным ускорением нужно использовать команду C_StartV или C_StartD с нулевым значением параметра.

Команда всегда возвращает код ошибки Error Code, который равен Err_No.

C_SetLim – установка режима работы концевых выключателей.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Fh	4	LimModeL							
		LimModeR							
		HomeMode							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Fh	1	Error Code							

Параметр LimModeL определяет режим работы концевого выключателя, ограничивающего движение привода назад, LimModeR – концевого выключателя, ограничивающего движение привода вперед, HomeMode – датчика базового положения. Назначение отдельных битов параметров приведено ниже:

D7	D6	D5	D4	D3	D2	D1	D0	
-	**-**	**-**	**-**	**-**	**AC**	**AL**	**EN**	**Назначение бита**
							0	концевой выключатель отключен
							1	концевой выключатель обрабатывается
						0		низкий активный уровень
						1		высокий активный уровень
					0			остановка с заданным ускорением
					1			мгновенная остановка
X	X	X	X	X				не используются

Если бит EN = 1, то концевой выключатель задействован. Если EN = 0, то концевой выключатель не используется (ограничение перемещения привода в данном направлении не происходит).

Если бит AL = 1, то контроллер работает с нормально-замкнутым (NC, или normal close) концевым выключателем. Если AL = 0, то контроллер работает с нормально-разомкнутым (NO, или normal open) концевым выключателем. Рекомендуется использовать нормально-замкнутые концевые выключатели, так как в этом случае в случае обрыва цепи концевого выключателя двигатель будет автоматически блокироваться.

Если бит AC = 1, то при срабатывании концевого выключателя или датчика базовой позиции производится мгновенная остановка двигателя. Если AC = 0, то остановка двигателя производится с заданным ускорением. Нужно отметить, что в случае мгновенной остановки работающего на высокой скорости двигателя координата может быть потеряна в результате пропуска шагов.

Любой из концевых выключателей или датчик базового положения может иметь логический ТТЛ-выход. Если он имеет НИЗКИЙ активный уровень, нужно устанавливать тип NO, если ВЫСОКИЙ – NC.

Заданный командой C_SetLim режим работы сохраняется в ОЗУ, поэтому при выключении и следующем включении питания восстановится старый режим. Для того чтобы новый режим работы записался в энергонезависимую память, нужно выполнить команду сохранения параметров C_SavePar.

Команда всегда возвращает код ошибки Error Code, который равен Err_No.

C_GetLim – чтение текущего состояния концевых выключателей.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
20h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
20h	2	Error Code							
		LimState							

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

Команда возвращает текущее состояние концевых выключателей и датчика базового положения в виде одного байта LimState. Назначение отдельных битов приведено ниже:

D7	D6	D5	D4	D3	D2	D1	D0	
-	**-**	**-**	**-**	**-**	**AC**	**AL**	**EN**	**Назначение бита**
							0	концевой выключатель L не сработал
							1	концевой выключатель L сработал
						0		концевой выключатель R не сработал
						1		концевой выключатель R сработал
					0			датчик базовой позиции сработал
					1			датчик базовой позиции не сработал
X	X	X	X	X				не используются

Срабатыванием концевых выключателей и датчика считаются такие логические уровни, которые совпадают с установленным с помощью команды C_SetLim активными уровнями.

C_SetCtrl – установка режима работы сигналов управления.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
21h	1	Ctrl							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
21h	1	Error Code							

Команда имеет параметр Ctrl, назначение отдельных битов которого приведено ниже:

D7	D6	D5	D4	D3	D2	D1	D0	
-	**-**	**DB**	**AL**	**M3**	**M2**	**M1**	**M0**	**Назначение бита**
				X	X	X	X	режим работы сигналов управления
			0					низкий активный уровень
			1					высокий активный уровень
		0						подавление дребезга выключено
		1						подавление дребезга включено
X	X							не используются

Сигнал «EN» разрешает работы двигателя. Биты M3…M0 задают режим работы сигналов управления «L» и «R». Возможны следующие режимы:

- M3…M0 = 0 – местное управление выключено. При этом двигатель не реагирует ни на один из сигналов управления.
- M3…M0 = 1 – сигнал «L» включает вращение назад, сигнал «R» включает вращение вперед. Движение происходит согласно заданным значениям скорости, скорости начала разгона и ускорения.
- M3…M0 = 2 – сигнал «L» включает вращение назад, сигнал «R» включает вращение вперед. Движение происходит на минимальной скорости.
- M3…M0 = 3 – сигнал «L» осуществляет один шаг двигателя, сигнал «R» определяет направление шага: активное состояние соответствует направлению вперед, пассивное – назад. Для предотвращения потери координаты следует использовать ток удержания.

- M3…M0 = 4 – сигнал «L» осуществляет один шаг двигателя назад, сигнал «R» – один шаг вперед. Для предотвращения потери координаты следует использовать ток удержания.
- M3…M0 = 5 – вращение двигателя вперед, сигнал «L» осуществляет уменьшение скорости, сигнал «R» – увеличение.
- M3…M0 = 6 – вращение двигателя назад, сигнал «L» осуществляет уменьшение скорости, сигнал «R» – увеличение.

Бит AL определяет активный уровень сигналов управления. Если AL = 0, то устанавливается низкий активный уровень. Это позволяет использовать для формирования сигналов управления кнопки с нормально-разомкнутыми контактами. При AL = 1 устанавливается высокий активный уровень сигналов управления.

Бит DB разрешает процедуру подавления дребезга. Когда для формирования сигналов управления используются кнопки, подавление дребезга должно быть включено (DB = 1). При этом минимальная длительность сигналов управления составляет 20 мс. При управлении логическими уровнями для повышения быстродействия подавление дребезга может быть отключено (DB = 0), тогда минимальная длительность сигналов управления будет 1 мс. В любом режиме минимальный период повторения сигналов управления не должен быть меньше 100 мс.

Заданный командой C_SetCtrl режим работы сохраняется в ОЗУ, поэтому при выключении и следующем включении питания восстановится старый режим. Для того чтобы новый режим работы записался в энергонезависимую память, нужно выполнить команду сохранения параметров C_SavePar.

Команда всегда возвращает код ошибки Error Code, который равен Err_No.

C_GetCtrl – чтение текущего состояния сигналов управления.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
22h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
22h	2	Error Code							
		CtrlState							

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

Команда возвращает текущее состояние сигналов управления в виде одного байта CtrlState. Назначение отдельных битов приведено ниже:

D7	D6	D5	D4	D3	D2	D1	D0	
-	-	-	-	-	**BE**	**BR**	**BL**	**Назначение бита**
							0	сигнал «L» не активен
							1	сигнал «L» активен
						0		сигнал «R» не активен
						1		сигнал «R» активен
					0			сигнал «EN» не активен
					1			сигнал «EN» активен
X	X	X	X	X				не используются

C_GetStat – чтение состояния контроллера.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
23h	0	-							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
23h	2	Error Code							
		State							

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

Команда возвращает текущее состояние контроллера в виде одного байта Stat, который может принимать следующие значения:

- Stat = 0 – привод остановлен.
- Stat = 1 – предыдущая команда успешно выполнилась.
- Stat = 2 – движение было прервано концевым выключателем.
- Stat = 3 – идет вращение.
- Stat = 4 – идет позиционирование.
- Stat = 5 – идет позиционирование по энкодеру.
- Stat = 6 – идет поиск базовой позиции.

Команда может быть использована для определения момента окончания выполнения позиционирования или базирования.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err_No.

C_GetPar – чтение установленных значений параметров.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
24h	0								

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
24h	19	Error Code							
		MS							
		PFD							
		Ih (low byte)							
		Ih (high byte)							
		Ia (low byte)							
		Ia (high byte)							
		Iw (low byte)							
		Iw (high byte)							
		Aw (low byte)							
		Aw (high byte)							
		Vm (low byte)							
		Vm (high byte)							
		Vw (low byte)							
		Vw (high byte)							
		LimModeL							
		LimModeR							
		HomeMode							
		Ctrl							

Команда не имеет параметров.

Команда возвращает набор параметров, которые были ранее установлены командами C_SetMode, C_SetIh, C_SetIa, C_SetIw, C_SetAw, C_SetVm, C_SetVw, C_SetLim, C_SetCtrl или считаны при включении питания из EEPROM. Назначение и диапазон значений параметров см. в описании соответствующих команд.

Команда всегда возвращает код ошибки Error Code, который равен Err_No.

C_SavePar – сохранение параметров в энергонезависимой памяти.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
25h	0								

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
25h	1	Error Code							

Команда не имеет параметров. Команда производит сохранение в энергонезависимой памяти значений следующих параметров:

- параметров движения (MS и PFD)
- тока удержания (Ih)
- тока разгона и торможения (Ia)
- рабочего тока (Iw)
- рабочего ускорения (Aw)
- скорости начала разгона (Vm)
- рабочей скорости (Vw)
- режима работы концевых выключателей (LimModeL, LimModeR, HomeMode)
- режима работы сигналов управления (Ctrl)
- текущей координаты (Nc)
- текущей координаты энкодера (Ne)
- координаты базовой позиции (Nb)

Сохранение параметров можно выполнять только тогда, когда двигатель остановлен. Выполнение команды может занимать время до 1 с.

Команда возвращает код ошибки Error Code, который может принимать значение Err_No в случае нормального выполнения команды и Err_Bu – в случае попытки выполнения команды во время движения.

C_Home – поиск базы.

TX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
26h	1	Dir							

RX

CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
26h	1	Error Code							

Команда имеет параметр Dir, который представляет собой число со знаком. Направление поиска базы определяется знаком параметра Dir, любые положительные значения соответствуют движению вперед (или вправо – R), любые отрицательные – движению назад (или влево – L). С нулевым значением Dir команда не выполняет никаких действий.

Поиск базы ведется на заданном значении рабочей скорости Vw. Командой C_SetLim может быть задана мгновенная остановка в базовой позиции или плавное торможение. Если во время выполнения команды базирования поступит команда C_StartV или C_StartD с нулевым параметром, C_Stop, сработает концевой выключатель, то выполнение команды прервется. Определить момент завершения команды поиска базы можно с помощью команды C_GetStat.

Команда всегда возвращает код ошибки Error Code, который равен Err_No.

Коды стандартных ошибок, определенных для протокола WAKE приведены в таблице 3:

Таблица 3. Коды ошибок.

Name	Error Code	Название ошибки
Err_No	00h	Нормальное завершение команды
Err_Tx	01h	Ошибка обмена с устройством
Err_Bu	02h	Устройство занято
Err_Re	03h	Устройство не готово
Err_Pa	04h	Ошибка значений параметров

ОПИСАНИЕ ФУНКЦИЙ DLL

Вместе с контроллером поставляется библиотека Smc32.dll, реализующая все функции управления контроллером. В ней каждая из команд протокола реализована в виде отдельной функции. Кроме того, библиотека содержит дополнительные функции, предназначенные для настройки порта. Назначение и диапазон допустимых значений параметров для всех функций см. в разделе описания системы команд. Список функций библиотеки Smc32.dll приведен ниже:

bool SMC_AccessPort(DWORD PortNum);

Проверяет доступность порта с номером ***PortNum***. Если порт доступен, возвращает ***true.***

bool SMC_OpenPort(DWORD PortNum);

Открывает порт с номером ***PortNum***. В случае успешного выполнения возвращает ***true***.

bool SMC_SetBaud(DWORD Baud);

Устанавливает скорость порта ***Baud***. В случае успешного выполнения возвращает ***true***.

bool SMC_SetRxTo(DWORD RxTo);

Устанавливает значение таймаута по приему ***RxTo*** в миллисекундах. В случае успешного выполнения возвращает ***true***.

bool WINAPI SMC_SetupPort(DWORD RtsMode, DWORD DtrMode);

Устанавливает режим сигнала RTS ***RtsMode*** и DTR ***DtrMode***. Для работы с контроллером SMC-5000 через переходник RS-232 – RS-485 требуется параметр ***RtsMode*** установить в RTS_CONTROL_TOGGLE (0x03), а ***DtrMode*** – в DTR_CONTROL_ENABLE (0x01). В случае успешного выполнения возвращает ***true***.

bool SMC_ClosePort(void);

Закрывает порт. В случае успешного выполнения возвращает ***true***.

void SMC_GetLastError(LPCSTR &lpcStr);

Возвращает строку с текстовым описанием последней ошибки.

bool SMC_GetInfo(int wna, LPCSTR &lpcStr);

Возвращает строку с именем устройства. Для SMC-5000 эта строка имеет вид «SMC-5000MA V1.0». В случае корректного ответа устройства возвращает ***true***.

bool SMC_SetAddr(int wna, int addr);

Устанавливает адрес устройства. Параметр ***wna*** может быть равен текущему адресу устройства или адресу коллективного вызова. Параметр ***addr*** представляет собой новый адрес устройства. В случае корректного ответа устройства возвращает ***true***.

bool SMC_GetAddr(int wna, int &addr);

Возвращает текущее значение адреса устройства. Параметр ***wna*** может быть равен текущему адресу устройства или адресу коллективного вызова. Параметр ***addr*** представляет собой текущий адрес устройства. В случае корректного ответа устройства возвращает ***true***.

Во всех последующих функциях параметр ***wna*** может быть равен адресу коллективного вызова (если используется одиночное устройство) или текущему адресу устройства (если устройство используется в сети). В случае корректного ответа устройства функции возвращают ***true***.

bool SMC_SetDio(int wna, int d);

Осуществляет управление линиями цифрового ввода-вывода. При использовании каких-то разрядов для ввода данных необходимо предварительно записать в эти разряды логические единицы.

bool SMC_GetDio(int wna, int &d);

Возвращает текущее состояние линий цифрового ввода-вывода.

bool SMC_GetAdc(int wna, int &v);

Возвращает текущее напряжение питания контроллера. Параметр ***v*** может принимать значения 0…400, что соответствует напряжению 0.0…40.0 В.

bool SMC_SetMode(int wna, int sm, int fm);

Устанавливает режим работы двигателя. Параметр ***sm*** определяет режим микрошага:

sm = 1 – полношаговый режим.
sm = 2 – полушаговый режим.
sm = 4 – деление шага на 4.
sm = 8 – деление шага на 8.

Параметр ***fm*** определяет скорость спада тока:

fm = 0 – медленный спад.
fm = 1 – быстрый спад.
fm = 3 – смешанный спад.

bool SMC_SetIh(int wna, int ih);

Устанавливает ток удержания. Параметр ***ih*** может принимать значения 0…2500 мА.

bool SMC_SetIa(int wna, int ia);

Устанавливает ток ускорения. Параметр ***ia*** может принимать значения 0…2500 мА.

bool SMC_SetIw(int wna, int iw);

Устанавливает рабочий ток. Параметр ***iw*** может принимать значения 0…2500 мА.

bool SMC_GetIc(int wna, int &ic);

Возвращает текущий ток двигателя. Значение ***ic*** может лежать в диапазоне 0…2500 мА.

bool SMC_SetAw(int wna, int aw);

Устанавливает рабочее ускорение. Параметр ***aw*** может принимать значения 0…32000 микрошагов/с^2.

bool SMC_GetAc(int wna, int &ac);

Возвращает текущее ускорение. Значение ***ac*** может лежать в диапазоне 0…32000 микрошагов/с^2.

bool SMC_SetVm(int wna, int vm);

Устанавливает минимальную скорость (скорость начала разгона). Параметр ***vm*** может принимать значения 0…32000 микрошагов/с.

bool SMC_SetVw(int wna, int vw);

Устанавливает рабочую скорость. Параметр ***vw*** может принимать значения 0…32000 микрошагов/с.

bool SMC_GetVc(int wna, int &vc);

Возвращает текущую скорость. Значение ***vc*** может лежать в диапазоне 0…32000 микрошагов/с.

bool SMC_SetNc(int wna, int nc);

Устанавливает текущую координату. Параметр ***nc*** может принимать значения ±2000000000 микрошагов.

bool SMC_GetNc(int wna, int &nc);

Возвращает текущую координату. Значение ***nc*** может лежать в диапазоне ±2000000000 микрошагов.

bool SMC_GetNb(int wna, int &nb);

Возвращает координату базовой позиции. Значение ***nb*** может лежать в диапазоне ±2000000000 микрошагов.

bool SMC_SetNe(int wna, int ne);

Устанавливает текущую координату энкодера. Параметр ***ne*** может принимать значения ±2000000000 микрошагов.

bool SMC_GetNe(int wna, int &ne);

Возвращает текущую координату энкодера. Значение ***ne*** может лежать в диапазоне ±2000000000 микрошагов.

bool SMC_StartV(int wna, int v);

Осуществляет старт двигателя на заданной скорости. Значение ***vc*** может лежать в диапазоне ±32000 микрошагов/с. Знак скорости определяет направление вращения.

bool SMC_StartD(int wna, int d);

Осуществляет старт двигателя в заданном направлении. Если ***d*** = 1 – вращение вперед, ***d*** = –1 – вращение назад, ***d*** = 0 – останов. Движение осуществляется на рабочей скорости.

bool SMC_StartN(int wna, int n);

Осуществляет старт двигателя на абсолютную позицию. Параметр ***n*** может принимать значения ±2000000000 микрошагов.

bool SMC_StartdN(int wna, int dn);

Осуществляет старт двигателя на относительную позицию. Параметр ***dn*** может принимать значения ±2000000000 микрошагов.

bool SMC_StartNe(int wna, int ne);

Осуществляет старт двигателя на абсолютную позицию энкодера. Параметр ***ne*** может принимать значения ±2000000000 микрошагов.

bool SMC_StartdNe(int wna, int dne);

Осуществляет старт двигателя на относительную позицию энкодера. Параметр ***dne*** может принимать значения ±2000000000 микрошагов.

bool SMC_Stop(int wna);

Останавливает двигатель без ускорения.

bool SMC_SetLim(int wna, int ml, int mr, int mh);

Устанавливает режим работы левого концевого выключателя (параметр ***ml***), правого концевого выключателя (параметр ***mr***) и датчика базового положения (параметр ***mh***). Каждый из параметров представляет собой набор битовых полей.

mx.0 = 0 –выключатель или датчик не используется.
mx.0 = 1 –выключатель или датчик используется.
mx.1 = 0 – низкий активный уровень.
mx.1 = 1 – высокий активный уровень.
mx.2 = 0 – остановка двигателя с ускорением.
mx.2 = 1 – мгновенная остановка.

bool SMC_GetLim(int wna, int &lm);

Возвращает текущее состояние концевых выключателей и датчика базового положения. Параметр ***lm*** представляет собой набор битовых полей.

lm.0 = 1 – сработал левый концевой выключатель.
lm.1 = 1 – сработал правый концевой выключатель.
lm.2 = 1 – сработал датчик базового положения.

bool SMC_SetCtrl(int wna, int cm);

Устанавливает режим работы сигналов местного управления «L», «R» и «EN». Сигнал «EN» во всех режимах осуществляет разрешение работы двигателя. Параметр ***cm*** представляет собой набор битовых полей.

cm.3:0 = 0 – сигнал «L» запрещен, «R» запрещен.
cm.3:0 = 1 – сигнал «L» вращение назад, «R» вращение вперед (на Vw).
cm.3:0 = 2 – сигнал «L» вращение назад, «R» вращение вперед (на Vm).
cm.3:0 = 3 – сигнал «L» шаг, «R» направление.
cm.3:0 = 4 – сигнал «L» шаг назад, «R» шаг вперед.
cm.3:0 = 5 – сигнал «L» уменьшение скорости, «R» увеличение скорости (вперед).
cm.3:0 = 6 – сигнал «L» уменьшение скорости, «R» увеличение скорости (назад).
cm.4 = 0 – низкий активный уровень для всех сигналов.
cm.4 = 1 – высокий активный уровень для всех сигналов.
cm.5 = 0 – подавление дребезга запрещено.
cm.5 = 1 – подавление дребезга разрешено.

bool SMC_GetCtrl(int wna, int &cs);

Возвращает текущее состояние сигналов местного управления «L», «R» и «EN». Параметр ***cs*** представляет собой набор битовых полей.

cs.0 = 1 – сигнал «L» активен.
cs.1 = 1 – сигнал «R» активен.
cs.2 = 1 – сигнал «EN» активен.

bool SMC_GetStat(int wna, int &st);

Возвращает текущее состояние контроллера.

st = 0 – двигатель остановлен.
st = 1 – команда выполнена успешно.
st = 2 – сработал концевой выключатель.
st = 3 – двигатель вращается.
st = 4 – идет позиционирование.
st = 5 – идет позиционирование по энкодеру.
st = 6 – идет поиск базовой позиции.

bool SMC_GetPar(int wna, int &sm, int &fm, int &ih, int &ia, int &iw,
int &aw, int &vm, int &vw, int &ml, int &mr, int &mh, int &cm);

Возвращает набор параметров, которые были ранее записаны в контроллер с помощью соответствующих команд или были считаны из энергонезависимой памяти контроллера после включения питания.

bool SMC_SavePar(int wna);

Сохраняет набор параметров в энергонезависимой памяти контроллера. Выполнять команду можно только при остановленном двигателе.

bool SMC_Home(int wna, int d);

Осуществляет старт поиска базовой позиции. Если ***d*** = 1 производится поиск вперед, если ***d*** = –1 – поиск назад, если ***d*** = 0 – останов. Движение осуществляется на рабочей скорости. При достижении базовой позиции контроллер запоминает ее координату, и двигатель останавливается. Остановка может осуществляться мгновенно или с заданным ускорением в зависимости от выбранного режима датчика базовой позиции.

ОПИСАНИЕ УПРАВЛЯЮЩЕЙ ПРОГРАММЫ

Для управления контроллером служит программа SMC.exe. Она позволяет использовать все функции контроллера, а также задавать и считывать все программируемые параметры движения.

Главное меню содержит два пункта: «Порт» и «Помощь». Меню «Порт» позволяет выбрать нужный COM-порт, который используется для подключения контроллера. Это может быть реальный последовательный порт компьютера, когда контроллер подключается через переходник RS-232 – RS-485 или виртуальный, когда контроллер подключается через переходник USB – RS-485. Для работы с таким переходником требуется специальный драйвер, который поставляется вместе с переходником. Меню «Помощь» содержит единственный пункт «О программе...», который позволяет открыть окно с информацией о производителе.

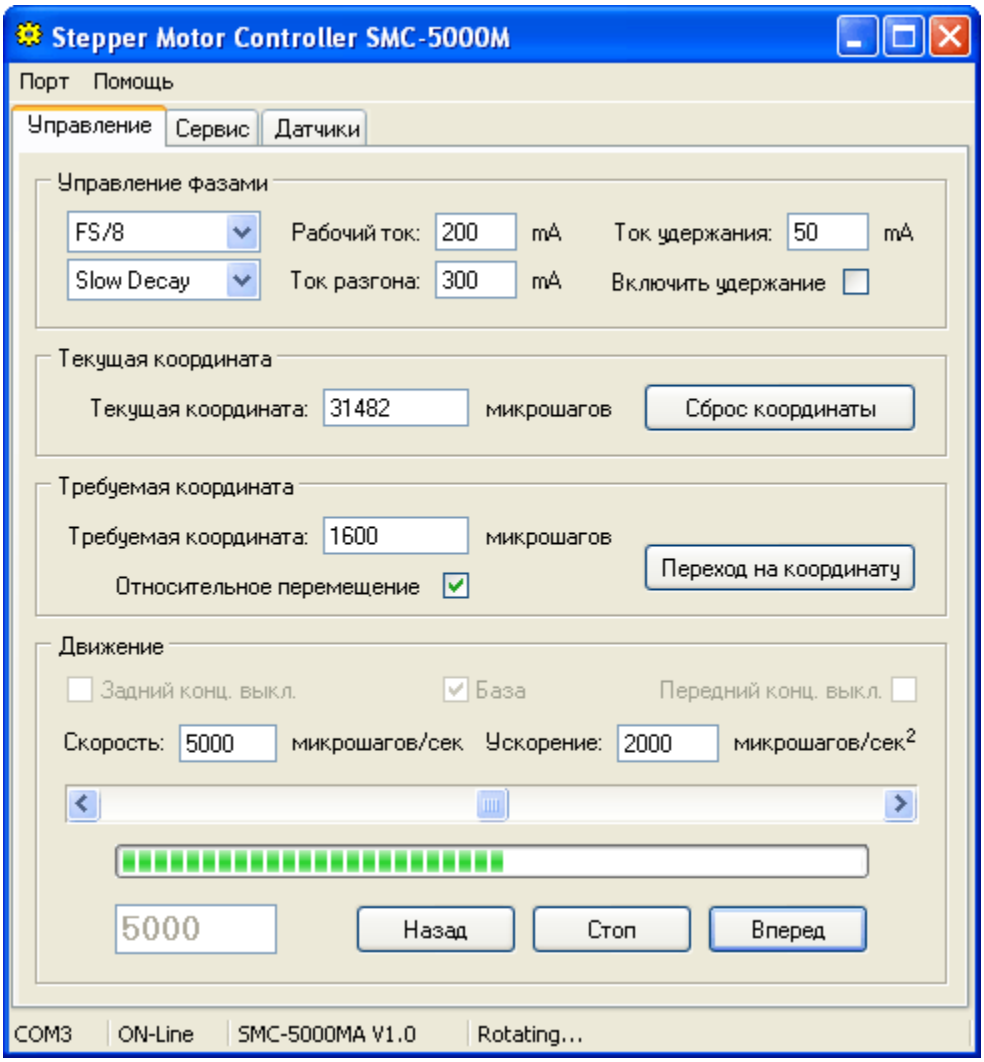

Рис. 3. Закладка «Управление» управляющей программы.

Внизу основного окна программы расположена строка состояния, в которой выводится имя используемого порта, состояние связи (ON-Line или OFF-Line), тип подключенного контроллера и текущее состояние контроллера.

На закладке «Управление» имеется группа «Управление фазами», внутри которой расположены выпадающие списки режима дробления шага, режима спада тока, а также поля для ввода рабочего тока, тока разгона и тока удержания. Ток удержания включается специальным флажком.

Группа «Текущая координата» содержит поле только для чтения, в котором отображается координата, и кнопку сброса текущей координаты.

Группа «Требуемая координата» содержит поле для ввода желаемой координаты и кнопку перехода на эту координату. Для выбора режима перехода на относительную координату служит флажок «Относительное перемещение».

Группа «Движение» содержит все необходимые органы управления движением шагового двигателя. Флажки «Задний конц. выкл.», «База» и «Передний конц. выкл.» показывают текущее состояние заднего концевого выключателя, датчика базовой позиции и переднего концевого выключателя соответственно. Для задания рабочей скорости есть поле ввода «Скорость». Скорость можно задать и с помощью скроллера. Максимальная скорость, которую можно задавать, ограничивается параметром VMAX, содержащимся в файле SMC.ini. По умолчанию (при отсутствии файла SMC.ini) этот параметр устанавливается равным 30000. При необходимости значение этого параметра можно изменить, воспользовавшись любым текстовым редактором. Ускорение задается в соответствующем поле ввода. Ниже скроллера расположен индикатор текущей скорости двигателя, которая выводится в виде горизонтальной шкалы и в цифровом виде. Для запуска двигателя в заданном направлении или для его остановки служат кнопки «Назад», «Вперед» и «Стоп».

На закладке «Сервис» расположена группа «Адресация», которая содержит поле ввода адреса контроллера и кнопку «Присвоить адрес». Для того чтобы объединить контроллеры в сеть, им нужно присвоить индивидуальные адреса. Сделать это можно, подключая контроллеры по очереди. При нажатии кнопки «Присвоить адрес» новый адрес передается в контроллер с использованием адреса коллективного вызова, что предусматривает наличие только одного подключенного контроллера в момент присвоения адреса.

Группа «Скорость начала разгона» содержит поле для ввода минимальной скорости.

Группа «Энкодер» содержит поле только для чтения, куда выводится текущая координата энкодера. Текущую координату можно обнулить с помощью кнопки «Сброс координаты». В поле «Требуемая координата» можно задать желаемую координату энкодера, перейти на которую можно с помощью кнопки «Переход на координату». Для выбора режима перехода на относительную координату служит флажок «Относительное перемещение».

Группа «Местное управление» позволяет задать режим работы сигналов местного управления. Для этого есть выпадающий список режимов, в котором через запятую перечислены функции сигналов «L», «R» и «EN» для каждого из режимов. Активный уровень для всех сигналов управления можно выбрать с помощью флажка «Высокий активный уровень». С помощью флажка «Подавление дребезга» можно включить или отключить функцию подавления дребезга. Эта функция должна быть включена, когда для управления используются механические переключатели.

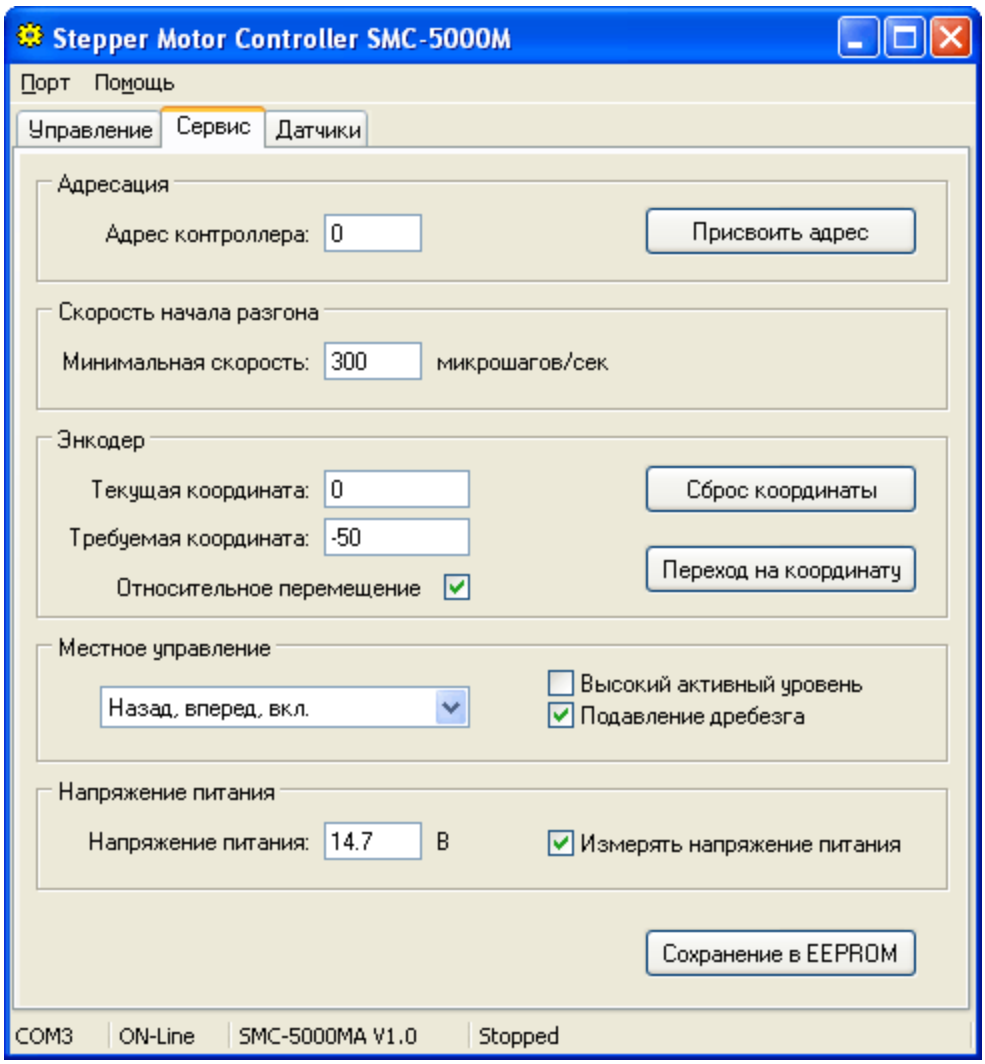

Рис. 4. Закладка «Сервис» управляющей программы.

Группа «Напряжение питания» содержит поле только для чтения, куда выводится текущее напряжение питания контроллера. Функцию измерения напряжения питания можно отключить специальным флажком.

Внизу на этой закладке расположена кнопка «Сохранение в EEPROM», с помощью которой можно все текущие параметры записать в энергонезависимую память контроллера. Выполнять эту функцию можно только при остановленном двигателе.

На закладке «Датчики» расположена группа «Концевые выключатели». Она содержит флажки, с помощью которых можно разрешить работу переднего и заднего концевого выключателя, задать тип выключателя (нормально-разомкнутые или нормально-замкнутые контакты), а также задать способ остановки при срабатывании концевого выключателя (мгновенная остановка или плавное торможение).

Группа «Датчик базового положения» позволяет разрешить работу датчика, задать тип датчика (нормально-разомкнутые или нормально-замкнутые контакты), а также задать способ

остановки при срабатывании датчика (мгновенная остановка или плавное торможение). Для осуществления процедуры поиска базы нужно нажать кнопку «Базирование», задав предварительно направление с помощью флажка «Направление поиска базы – вперед». Когда операция поиска базы завершится, в поле только для чтения «Координата» будет выведена координата базовой позиции в микрошагах.

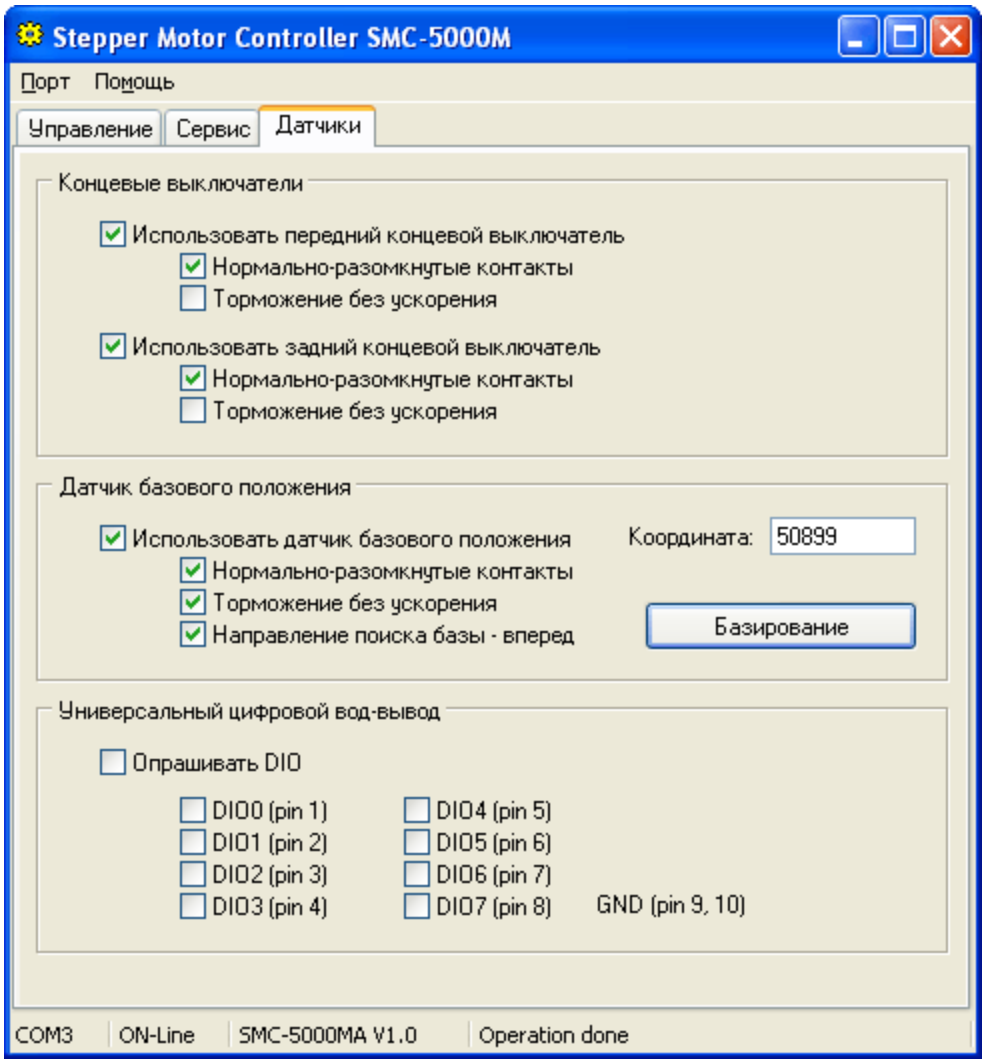

Рис. 5. Закладка «Датчики» управляющей программы.

Группа «Универсальный цифровой ввод-вывод» позволяет осуществлять управление и просматривать текущее состояние каждой из восьми линий цифрового ввода-вывода. Все линии являются двунаправленными, тип выходов – открытый сток. Для того чтобы использовать какую-то из линий на ввод, необходимо предварительно записать в этот разряд логическую единицу. Отмеченные флажки соответствуют логическому нулю.

Приложение 1. Последовательности задания тока фаз для разных режимов работы.

FS	HS	FS/4	FS/8	Ток фазы 1, (%)	Ток фазы 2, (%)	Угол, (°)
	1	1	1	100.00	0.00	0.0
			2	98.08	19.51	11.3
		2	3	92.39	38.27	22.5
			4	83.15	55.56	33.8
1	2	3	5	70.71	70.71	45.0
			6	55.56	83.15	56.3
		4	7	38.27	92.39	67.5
			8	19.51	98.08	78.8
	3	5	9	0.00	100.00	90.0
			10	-19.51	98.08	101.3
		6	11	-38.27	92.39	112.5
			12	-55.56	83.15	123.8
2	4	7	13	-70.71	70.71	135.0
			14	-83.15	55.56	146.3
		8	15	-92.39	38.27	157.5
			16	-98.08	19.51	168.8
	5	9	17	-100.00	0.00	180.0
			18	-98.08	-19.51	191.3
		10	19	-92.39	-38.27	202.5
			20	-83.15	-55.56	213.8
3	6	11	21	-70.71	-70.71	225.0
			22	-55.56	-83.15	236.3
		12	23	-38.27	-92.39	247.5
			24	-19.51	-98.08	258.8
	7	13	25	0.00	-100.00	270.0
			26	19.51	-98.08	281.3
		14	27	38.27	-92.39	292.5
			28	55.56	-83.15	303.8
4	8	15	29	70.71	-70.71	315.0
			30	83.15	-55.56	326.3
		16	31	92.39	-38.27	337.5
			32	98.08	-19.51	348.8