

# SMC-4000A

---

## Контроллер шагового двигателя

wake 

---

*руководство пользователя*

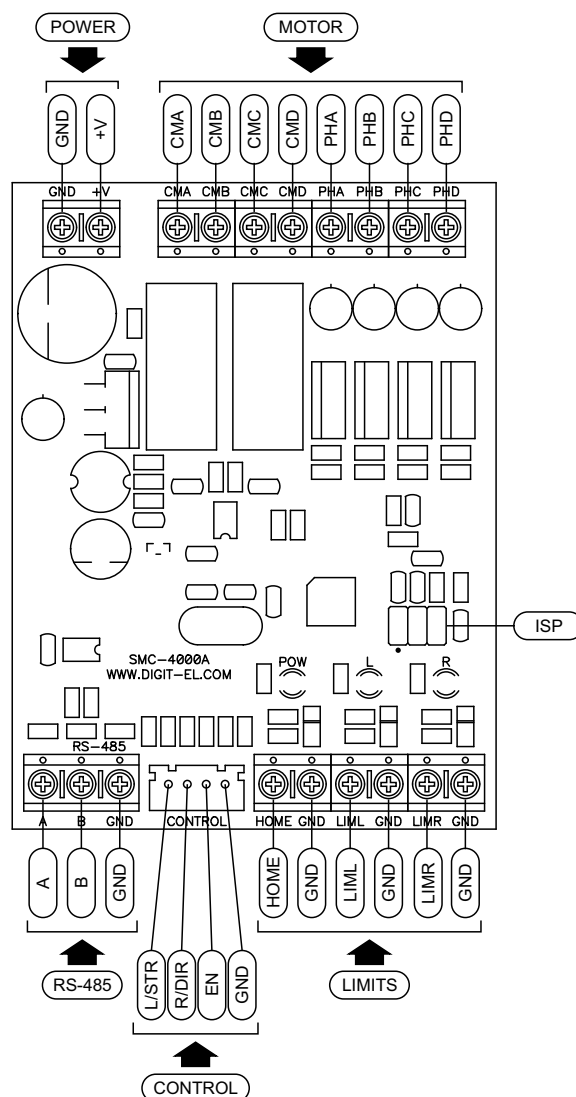
## ОСНОВНЫЕ ХАРАКТЕРИСТИКИ

- управление униполярными шаговыми двигателями
- максимальное напряжение питания – 35 В
- максимальный ток питания двигателя – 4 А
- ШИМ-стабилизация тока обмоток двигателя
- поддержка полношагового и полушагового режима
- скорость вращения от 1 до 30 000 полушагов/сек
- подключение двух концевых выключателей (NO или NC)
- программное разрешение/запрещение концевых выключателей
- подключение датчика базового положения
- разгон с заданным ускорением
- программируемый ток фаз для разгона, движения и удержания
- перемещение на абсолютную координату или на заданное число шагов
- абсолютная координата  $\pm 2\,000\,000\,000$  полушагов
- 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
- управление по интерфейсу RS-485
- включение в цепочку по RS-485 до 64 контроллеров
- протокол обмена – WAKE
- один источник питания +10...35 В
- крепление на DIN-рейку
- тестовое ПО (Win98/ME/2000/XP)
- библиотека функций управления (DLL)

## ОПИСАНИЕ УСТРОЙСТВА

Контроллер шагового двигателя SMC-4000A предназначен для работы с униполярными шаговыми двигателями. Управление устройством обеспечивает микроконтроллер ATmega8 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). Как опция, возможна установка интерфейса RS-232 или USB. В этом случае организация сети контроллеров невозможна. 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.**

Напряжение питания подключается к клеммникам «POWER» (+V и GND). 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 шагового двигателя плата имеет группу из 8-ми клеммников «MOTOR». Контроллер рассчитан на подключение униполярного шагового двигателя с четырьмя обмотками, или двигателя с двумя обмотками, которые имеют отводы от середины. Начала обмоток подключают к клеммам PHA...PHD (выходы транзисторных ключей), концы обмоток (или отводы от середины) подключают к клеммам CMA...CMD (напряжение питания +V). Например, двигатель типа ДШИ-200 подключается следующим образом (вывод двигателя – клемма): НА – PHA, НБ – PHB, НВ – PHC, НГ – PHD, КА – CMA, КБ – CMB, KB – CMC, КГ – CMD.

Для подключения концевых выключателей служит группа из 6-ти клеммников «LIMITS». Концевой выключатель, ограничивающий движение шагового привода назад, подключается к клеммам LIML и GND. Концевой выключатель, ограничивающий движение шагового привода вперед – к клеммам LIMR 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.

Местное управление контроллером осуществляется через разъем «CONTROL». Этот разъем имеет следующие контакты (слева направо): задание направления вращения назад (L/STR), задание направления вращения вперед (R/DIR), разрешение работы двигателя (EN), земля (GND). Все сигналы имеют подтягивающие резисторы на уровень +5 В. 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 интерфейс RS-485 (клеммник «RS-485»). Контроллеры можно объединять в цепочку (до 64 штук), которая через преобразователь интерфейсов может быть подключена к одному COM-порту (или USB-порту) компьютера.

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

## **РЕЖИМЫ РАБОТЫ**

Контроллер обеспечивает работу биполярных шаговых двигателей в полношаговом (FS, full step) и полушаговом (HS, half step) режиме.

Все значения координат, скоростей и ускорений задаются с использованием полушага HS. При работе с полным шагом контроллер способен обрабатывать значения, кратные 2-м.

## **СКОРОСТЬ И УСКОРЕНИЕ**

Контроллер позволяет устанавливать скорость от 0 до 30000 полушагов в секунду. При работе контроллера в режиме HS это значение непосредственно определяет скорость в полушагах в секунду. Если контроллер переключить в режим FS, реальное число шагов в секунду будет вдвое меньшим, при этом частота вращения вала двигателя останется той же.

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 больше, 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 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-4000A поддерживает обработку относительной и абсолютной координаты. Значение координаты всегда выражается в полушагах HS. Поэтому при работе с полным шагом возможно лишь более грубое позиционирование. Контроллер имеет счетчик абсолютной координаты емкостью  $\pm 2\,000\,000\,000$  полушагов. 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 полушагах.

## **КОНЦЕВЫЕ ВЫКЛЮЧАТЕЛИ**

Контроллер позволяет подключить 2 концевых выключателя. Включение и отключение обработки концевых выключателей осуществляется программно. 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.

## **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. Светодиод «POWER» индицирует наличие напряжения питания.

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.

### **АДРЕСАЦИЯ КОНТРОЛЛЕРОВ**

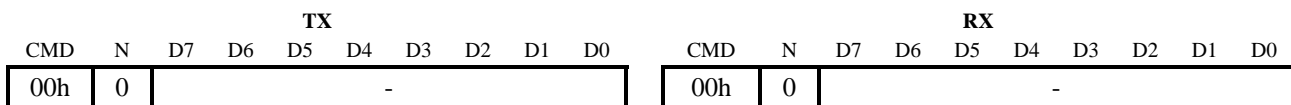
Контроллеры можно объединять в сеть, используя дополнительный разъем RS-485. Перед объединением контроллеров в сеть каждому из них должен быть присвоен уникальный адрес, который сохраняется в энергонезависимой памяти.

Для подключения к компьютеру используется переходник RS-232 – RS-485 или USB – RS-485. Контроллер может комплектоваться переходниками нескольких типов: от простейшего, собранного в вилке разъема, до промышленного, с гальванической развязкой сигналов. Выбор переходника определяется протяженностью сети RS-485 и электромагнитной обстановкой в месте установки контроллера шагового двигателя.

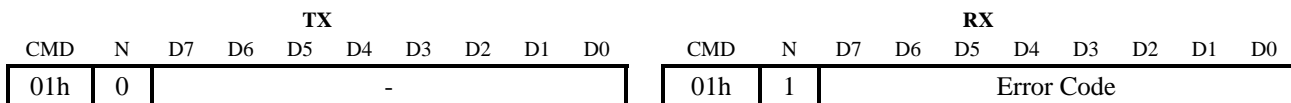
**КОМАНДЫ КОНТРОЛЛЕРА ШАГОВОГО ДВИГАТЕЛЯ.**

Команды передаются в виде пакетов согласно протоколу WAKE. Скорость обмена – 19200 бод, длина слова данных – 8 бит, бит четности не используется, 1 стоп-бит. Инициатором обмена всегда выступает РС. 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.

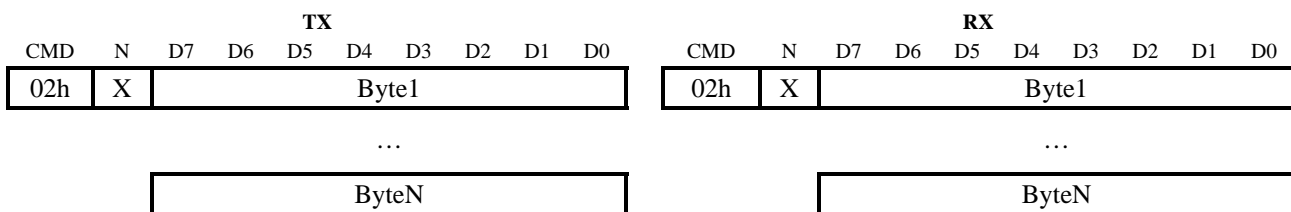
C\_Nop – нет операции. Используется для внутренних целей и никогда не передается в контроллер или РС.



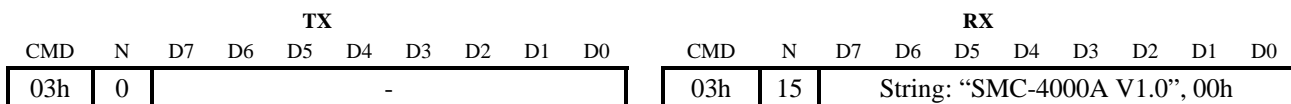
C\_Err – контроллер передает эту команду в РС в качестве ответа на любую команду, если произошла ошибка приема пакета. Для этой команды Error Code всегда равен Err\_Tx.



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.



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-4000A V1.0, где SMC-4000A – тип устройства, V1.0 – версия firmware 1.0. В качестве разделителей используются пробелы (код 20h). Строка заканчивается байтом 00h.





C\_GetAdc – чтение напряжения питания.

TX										RX										
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	
08h	0										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										RX										
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	
09h	1	SM									09h	1	Error Code							

Параметр SM определяет величину шага:

- SM = 0 – полношаговый режим (с перекрытием фаз)
- SM = 1 – полушаговый режим (деление шага на 2)

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.

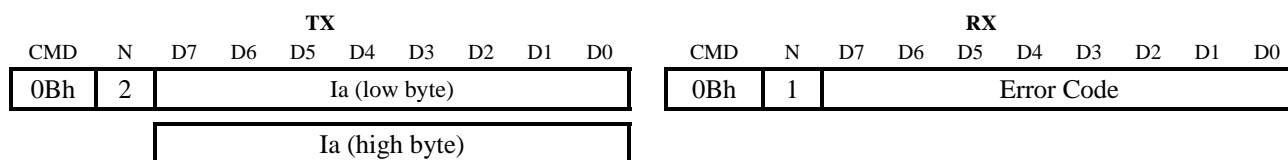
C\_SetIh – задание тока удержания.

TX										RX										
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	
0Ah	2	Ih (low byte)									0Ah	1	Error Code							
										Ih (high byte)										

Команда имеет параметр Ih, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 0...4000 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.

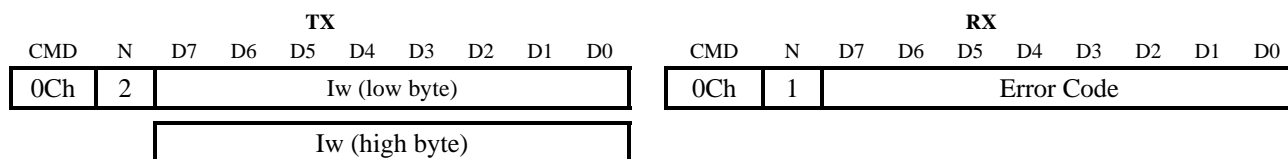
C\_SetIa – задание тока разгона и торможения.



Команда имеет параметр Ia, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 0...4000 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.



Команда имеет параметр Iw, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 0...4000 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										RX											
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0		
0Dh	0	-									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...4000 мА.

C\_SetAw – задание ускорения.

TX										RX											
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0		
0Eh	2	Aw (low byte)									0Eh	1	Error Code								
										Aw (high byte)											

Команда имеет параметр Aw, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 0...30000 полушагов/сек<sup>2</sup> (1 полушаг = 1/2 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.

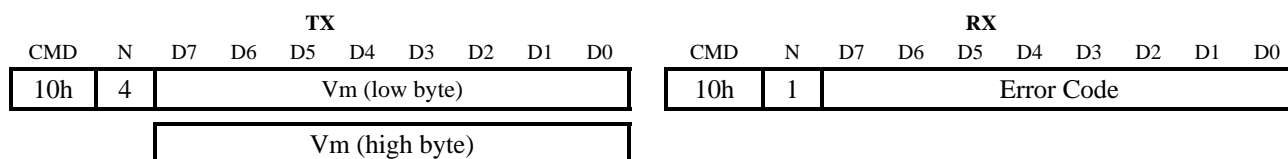
TX										RX											
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0		
0Fh	0	-									0Fh	3	Error Code								
										Ac (low byte)											
										Ac (high byte)											

Команда не имеет параметров.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err\_No.

Команда возвращает значение текущего ускорения  $A_c$  в виде 2-х байтового числа со знаком, которое может принимать значения  $\pm 30000$  полушагов/сек<sup>2</sup> (1 полушаг = 1/2 полного шага). Если двигатель остановлен или вращается с постоянной скоростью, команда возвращает 0.

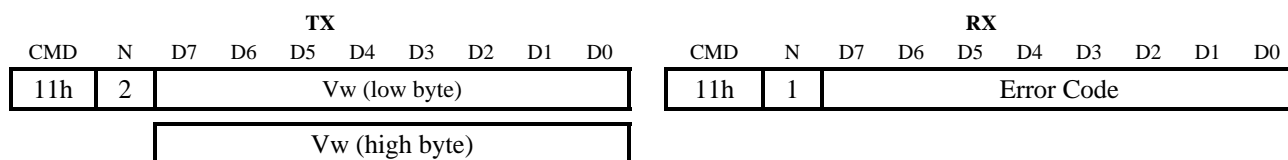
C\_SetVm – задание скорости начала разгона.



Команда имеет параметр Vm, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 1...30000 полушагов/сек (1 полушаг = 1/2 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.



Команда имеет параметр Vw, который представляет собой двухбайтовое число без знака. Параметр может принимать значения 0...30000 полушагов/сек (1 полушаг = 1/2 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										RX										
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	
12h	0										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  $\pm 30000$  полушагов/сек (1 полушаг = 1/2 полного шага).

C\_SetNc – установка текущей координаты.

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

Команда имеет параметр Nc, который представляет собой четырехбайтовое знаковое число (байт 1 – младший, байт 4 – старший). Параметр может принимать значения  $\pm 2000000000$  и определяет координату в полушагах (1 полушаг = 1/2 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										RX											
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0		
14h	0	-									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  $\pm 2000000000$  и представляет собой координату в полушагах (1 полушаг = 1/2 полного шага).

C\_GetNb – чтение координаты базовой позиции (HOME).

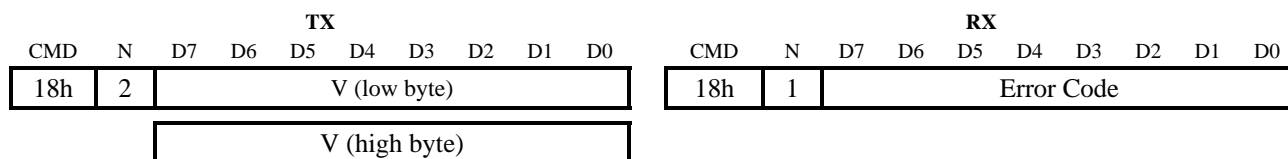
TX										RX											
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0		
15h	0	-									15h	5	Error Code								
										Nb (byte 1)											
										Nb (byte 2)											
										Nb (byte 3)											
										Nb (byte 4)											

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  $\pm 2000000000$  и представляет собой координату базовой позиции (HOME) в полушагах (1 полушаг = 1/2 полного шага). Эта координата сохраняется контроллером в тот момент, когда срабатывает датчик базовой позиции.

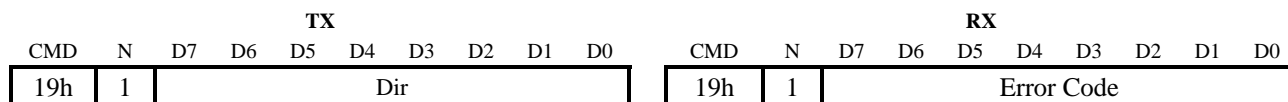
C\_StartV – старт двигателя на заданной скорости.



Команда имеет параметр V, который представляет собой двухбайтовое число со знаком. Параметр может принимать значения  $\pm 32000$  полушагов/сек (1 полушаг = 1/2 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.

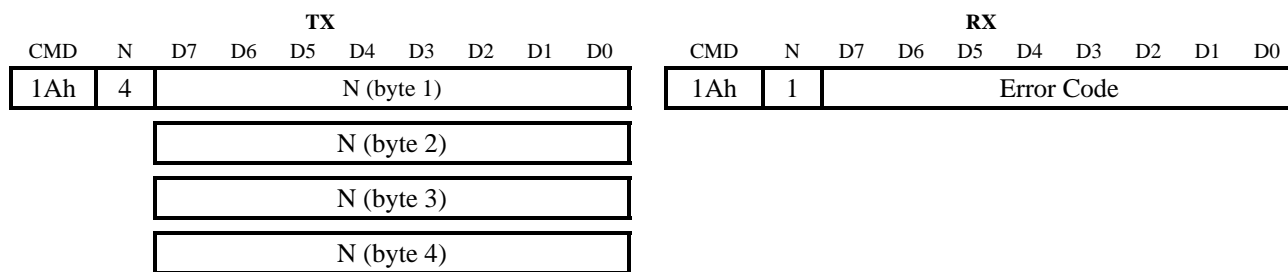
C\_StartD – старт двигателя в заданном направлении.



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.



Команда имеет параметр N, который представляет собой четырехбайтовое знаковое число (байт 1 – младший, байт 4 – старший). Параметр может принимать значения

$\pm 2000000000$  и определяет координату в полушагах (1 полушаг = 1/2 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								RX									
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
1Bh	4	dN (byte 1)								1Bh	1	Error Code							
		dN (byte 2)																	
		dN (byte 3)																	
		dN (byte 4)																	

Команда имеет параметр dN, который представляет собой четырехбайтовое знаковое число (байт 1 – младший, байт 4 – старший). Параметр может принимать значения  $\pm 2000000000$  и определяет, на какое количество полушагов (1 полушаг = 1/2 полного шага) выполнит перемещение двигатель. Направление вращения определяется знаком параметра: отрицательным значениям соответствует вращение назад, положительным – вперед. После выполнения перемещения на заданное количество полушагов 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\_Stop – остановка двигателя.

TX										RX											
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0		
1Eh	0	-									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										RX											
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0		
1Fh	4	LimModeL									1Fh	1	Error Code								
		LimModeR																			
		HomeMode																			

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	Назначение бита
-	-	-	-	-	AC	AL	EN	
							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

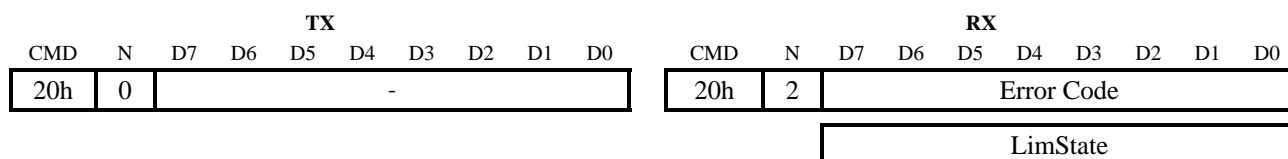
двигателя производится с заданным ускорением. Нужно отметить, что в случае мгновенной остановки работающего на высокой скорости двигателя координата может быть потеряна в результате пропуска шагов.

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.



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	Назначение бита
-	-	-	-	-	AC	AL	EN	
							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								RX											
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
21h	1	Ctrl								21h	1	Error Code							

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

D7	D6	D5	D4	D3	D2	D1	D0	Назначение бита
-	-	DB	AL	M3	M2	M1	M0	
				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										RX											
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0		
22h	0										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	Назначение бита	
-	-	-	-	-	BE	BR	BL	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										RX											
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0		
23h	0										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, которое может быть использовано для определения момента окончания выполнения позиционирования или базирования. Состояние возвращается в виде одного байта Stat, который может принимать следующие значения:

- Stat = 0 – привод остановлен.
- Stat = 1 – предыдущая команда успешно выполнена.
- Stat = 2 – движение было прервано конечным выключателем.
- Stat = 3 – идет вращение.
- Stat = 4 – идет позиционирование.
- Stat = 6 – идет поиск базовой позиции.

Команда возвращает код ошибки Error Code, который всегда имеет значение Err\_No.

C\_GetPar – чтение установленных значений параметров.

TX										RX									
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0
24h	0									24h	19	Error Code							
										SM									
										FM									
										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. Параметр FM всегда равен нулю.

Команда всегда возвращает код ошибки Error Code, который равен Err\_No.

C\_SavePar – сохранение параметров в энергонезависимой памяти.

TX										RX										
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	
25h	0									25h	1									Error Code

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

- параметра движения (SM)
- 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)
- 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\_Vu – в случае попытки выполнения команды во время движения.

C\_Home – поиск базы.

TX										RX										
CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	CMD	N	D7	D6	D5	D4	D3	D2	D1	D0	
26h	1								Dir	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-4000A эта строка имеет вид «SMC-4000A 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\_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);***

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

*sm* = 0 – полношаговый режим.

*sm* = 1 – полушаговый режим.

***bool SMC\_SetIh(int wna, int ih);***

Устанавливает ток удержания. Параметр *ih* может принимать значения 0...4000 мА.

***bool SMC\_SetIa(int wna, int ia);***

Устанавливает ток ускорения. Параметр *ia* может принимать значения 0...4000 мА.

***bool SMC\_SetIw(int wna, int iw);***

Устанавливает рабочий ток. Параметр *iw* может принимать значения 0...4000 мА.

***bool SMC\_GetIc(int wna, int &ic);***

Возвращает текущий ток двигателя. Значение *ic* может лежать в диапазоне 0...4000 мА.

***bool SMC\_SetAw(int wna, int aw);***

Устанавливает рабочее ускорение. Параметр *aw* может принимать значения 0...30000 полушагов/с<sup>2</sup>.

***bool SMC\_GetAc(int wna, int &ac);***

Возвращает текущее ускорение. Значение *ac* может лежать в диапазоне 0...30000 полушагов/с<sup>2</sup>.

***bool SMC\_SetVm(int wna, int vm);***

Устанавливает минимальную скорость (скорость начала разгона). Параметр *vm* может принимать значения 0...30000 полушагов/с.

***bool SMC\_SetVw(int wna, int vw);***

Устанавливает рабочую скорость. Параметр *vw* может принимать значения 0...30000 полушагов/с.

***bool SMC\_GetVc(int wna, int &vc);***

Возвращает текущую скорость. Значение *vc* может лежать в диапазоне 0...30000 полушагов/с.

***bool SMC\_SetNc(int wna, int nc);***

Устанавливает текущую координату. Параметр *nc* может принимать значения  $\pm 2000000000$  полушагов.

***bool SMC\_GetNc(int wna, int &nc);***

Возвращает текущую координату. Значение ***nc*** может лежать в диапазоне  $\pm 2000000000$  полушагов.

***bool SMC\_GetNb(int wna, int &nb);***

Возвращает координату базовой позиции. Значение ***nb*** может лежать в диапазоне  $\pm 2000000000$  полушагов.

***bool SMC\_StartV(int wna, int v);***

Осуществляет старт двигателя на заданной скорости. Значение ***vc*** может лежать в диапазоне  $\pm 30000$  полушагов/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  $\pm 2000000000$  полушагов.

***bool SMC\_StartdN(int wna, int dn);***

Осуществляет старт двигателя на относительную позицию. Параметр ***dn*** может принимать значения  $\pm 2000000000$  полушагов.

***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  $V_w$ ).

*cm.3:0* = 2 – сигнал «L» вращение назад, «R» вращение вперед (на  $V_m$ ).

*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, = 1 – высокий активный уровень.

*cm.5* = 0 – подавление дребезга запрещено, = 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* = 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. Параметр *fm* всегда равен нулю.

***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 SMC4000.exe. Она позволяет использовать все функции контроллера, а также задавать и считывать все программируемые параметры движения.

Программа использует динамическую библиотеку smc32.dll.

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.

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.

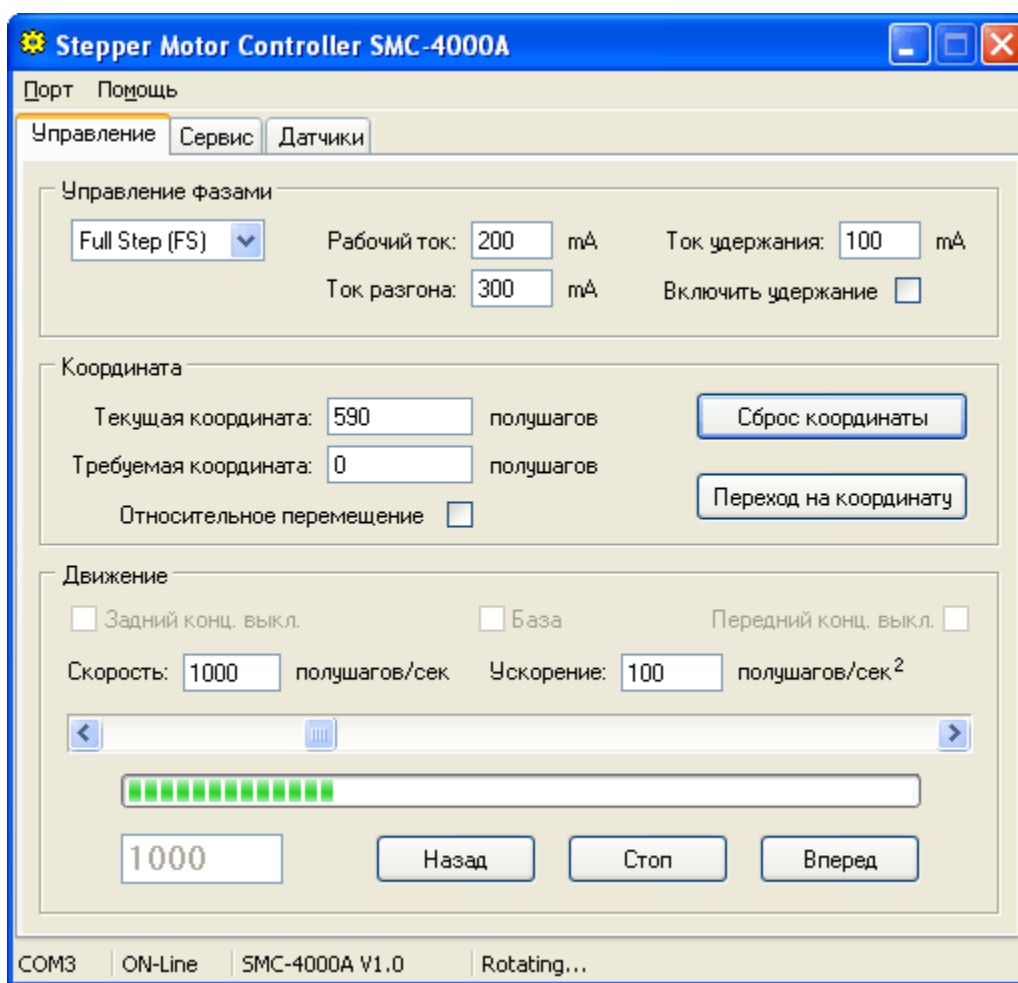


Рис. 2. Закладка «Управление» управляющей программы.

На закладке «Управление» (рис. 2) имеется группа «Управление фазами», внутри которой расположен список режимов дробления шага, а также поля для ввода рабочего тока, тока разгона и тока удержания. Ток удержания включается специальным флажком.

Группа «Координата» содержит поле для отображения текущей координаты, поле для ввода желаемой координаты, кнопку сброса текущей координаты и кнопку перехода на заданную координату. Для выбора режима перехода на относительную координату служит флажок «Относительное перемещение».

Группа «Движение» содержит флажки «Задний конц. выкл.», «База» и «Передний конц. выкл.», которые показывают текущее состояние концевых выключателей и датчика базовой позиции. Для задания рабочей скорости есть поле ввода «Скорость». Скорость можно задать и с помощью скроллера. Максимальная скорость, которую можно задавать, ограничивается на закладке «Сервис». 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».

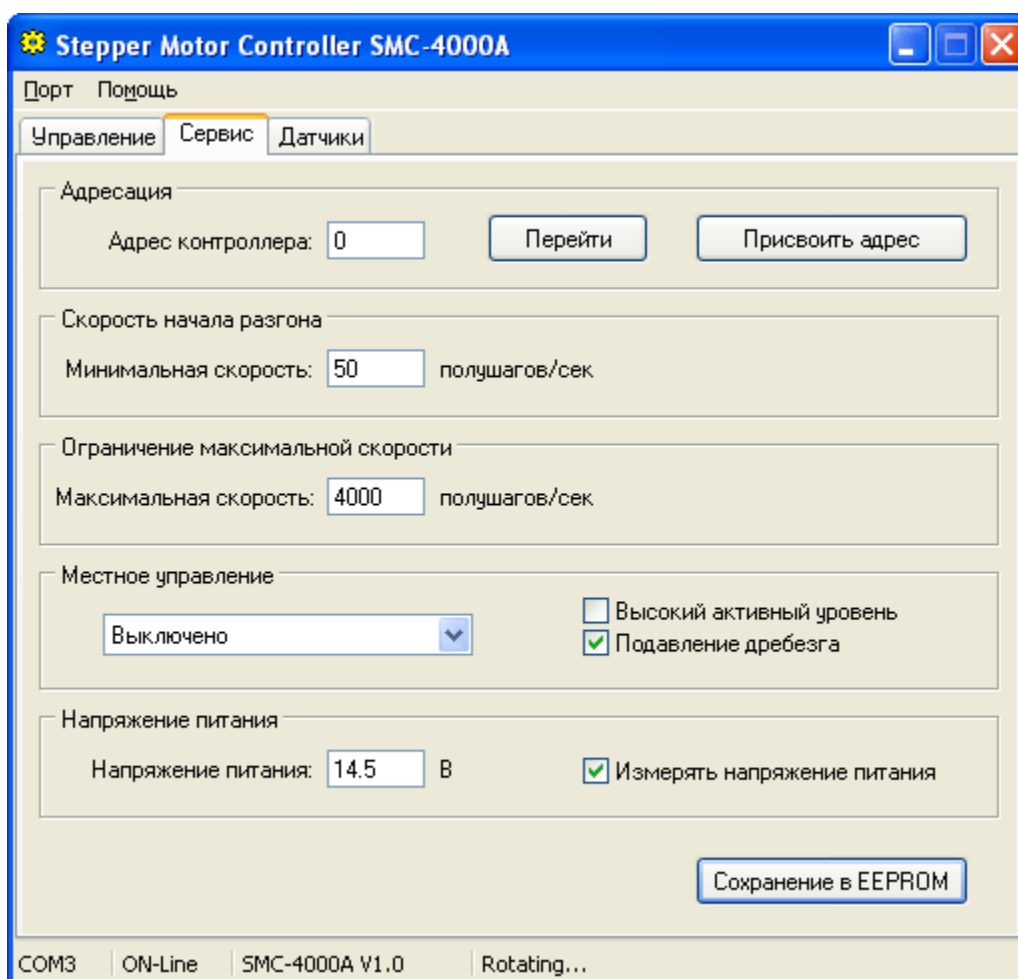


Рис. 3. Закладка «Сервис» управляющей программы.

На закладке «Сервис» (рис. 3) расположена группа «Адресация», которая содержит поле ввода адреса контроллера, кнопку «Перейти» 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. Если адреса уже присвоены и контроллеры объединены в сеть, то чтобы обратиться к нужному контроллеру необходимо ввести его адрес и нажать кнопку «Перейти». После этого все обращения будут осуществляться по заданному адресу.

Группа «Скорость начала разгона» содержит поле для ввода минимальной скорости, с которой начинается разгон двигателя. Это не запрещает запуск двигателя на меньших скоростях, но плавный разгон для этих скоростей производиться не будет. Ограничение минимальной скорости разгона используется для предотвращения повышенного шума и вибрации при пусках и остановках.

Группа «Ограничение максимальной скорости» задает предельное значение скорости, которое может быть задано с помощью поля ввода и скроллера, расположенного на закладке «Управление». Эта возможность служит для лучшего использования шкалы скроллера.

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.

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» (рис. 4) 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 полушагах.

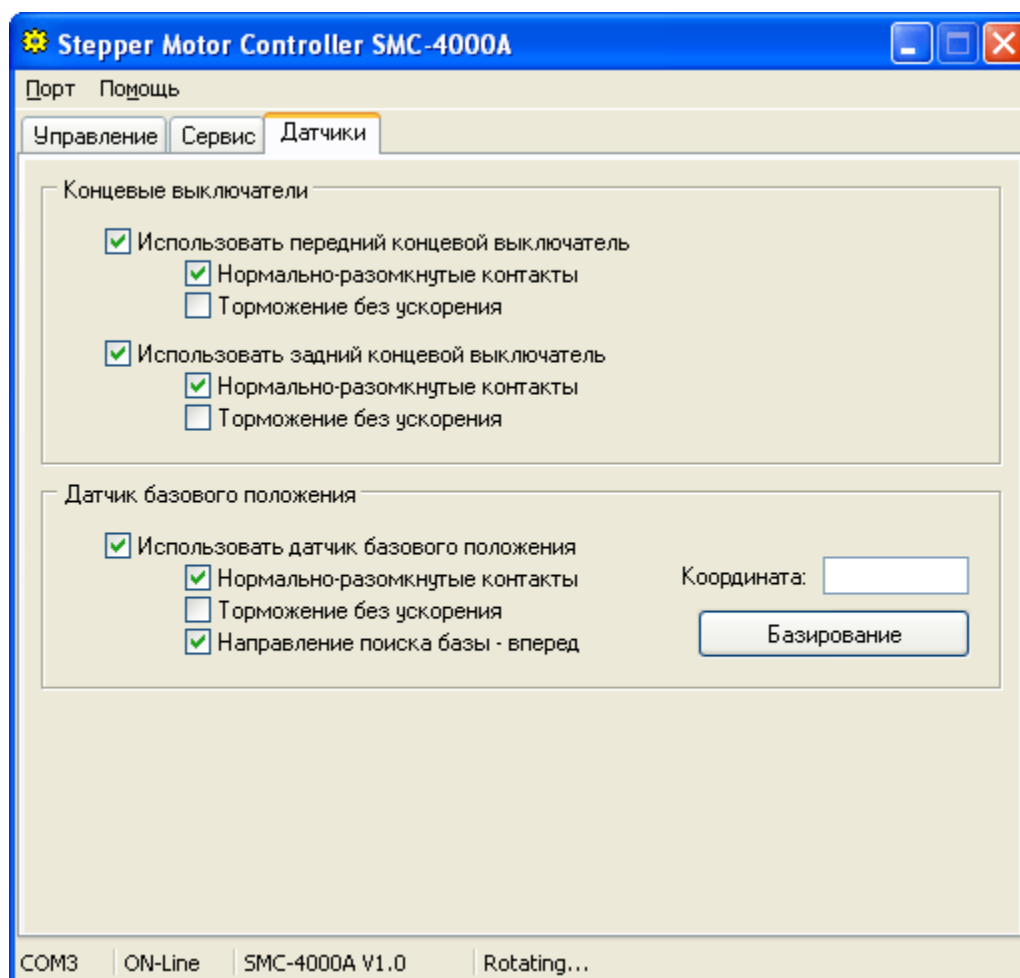


Рис. 4. Закладка «Датчики» управляющей программы.

## ИНФОРМАЦИЯ О ПРОИЗВОДИТЕЛЕ

Частное предприятие «Диджит-ЭЛ»,  
пр. Независимости 44, ком. П7,  
220005, г. Минск, Республика Беларусь

Тел./факс: (017) 284-79-41

e-mail: [info@digit-el.com](mailto:info@digit-el.com)

[www.digit-el.com](http://www.digit-el.com)