Считыватель PR-G07-N

Руководство программиста

Версия 1.0 июнь 2016 г.

Оглавление

История документа

Версия	Дата	Изменения
1.0	27.06.2016	Первая редакция документа

Введение

Считыватель PR-G07-N разработан для замены ранее выпускавшихся считывателей типа PR-G07. По базовому функционалу считыватель унаследовал практически все функции старой версии, но при этом отличительные особенности считывателя — добавление новых функций, а именно:

- Наличие двух интерфейсов связи с хостом: RS-485 и Ethernet. Оба интерфейса активны одновременно, но пользоваться в конкретный момент времени можно только одним из них. Исключение составляет встроенный WEB-сервер, которым, в принципе, можно пользоваться одновременно с интерфейсом RS-485.
- Возможность питания как от отдельного источника питания 12В постоянного тока, так и по стандарту PoE (питание по сети Ethernet).
- Дополнение функций считывателя функционалом контроллера доступа с возможностью управления исполнительными устройствами с помощью встроенных реле.
- Наличие встроенного WEB- сервера для программирования параметров считывателя (программирование через основной интерфейс сохраняется).

В связи с переходом на новую концепцию построения встроенного ПО считывателя и существенное расширение функционала протокол обмена со считывателем кардинально отличается от протокола старой версии, однако в дальнейшем планируется написание динамической библиотеки, позволяющей реализовать старый функционал без изменения формата и параметров функций DLL.

Данный документ предоставляет информацию, необходимую для непосредственной работы со считывателем дальней идентификации PR-G07-N по интерфейсам RS-485 и Ethernet. При этом в качестве ведущего может выступать любое устройство, в том числе контроллеры пользователя.

Не допускается использование кодов команд иных, чем описанные в данном документе.

Что нового в этом документе

Данный документ представляет собой первую редакцию. Если вы нашли в документе опечатки или неточности, пожалуйста, сообщите о них по электронной почте: sleo@relvest.ru.

Совместимость

Данный документ полностью соответствует функционалу считывателей PR-G07-N версии 2.1 или старше. В последующих версиях считывателей описанные в данном документе функции даже при замене на новые будут оставлены для совместимости с ранее написанными пользовательскими приложениями.

Принцип работы устройства

Считыватель предназначен для слежения за активными метками (тагами), попадающими в зону считывания. Считыватель может одновременно отслеживать до 64-х тагов, находящихся в зоне чтения каждого радиоканала. Дальность чтения в зависимости от условий эксплуатации и применяемой антенны составляет от 10 до 50 метров.

Для обеспечения требуемых характеристик при использовании в системах доступа (например, на автомобильных проходных) считыватель выполнен двухканальным. В общем случае каналы считывателя могут работать независимо, хотя для работы в составе систем доступа имеет мощный настраиваемый механизм межканального взаимодействия, позволяющий наряду с опросом датчиков (по два на канал) реализовывать сложные механизмы прохода или проезда.

Ниже описаны основные понятия, позволяющие лучше понять принципы работы считывателя.

Время памяти тага

Обмен информацией между считывателем и тагом производится примерно два раза в секунду, но при этом возможны пропуски обменов за счет эфирных помех, коллизий (наложений обменов двух и более тагов в одно и то же время) и других факторов.

Для того, чтобы сделать слежение за тагами устойчивым, считыватель в каждом канале имеет встроенную память на 64 тага (стек текущих тагов). В этой памяти фиксируется каждый вновь появившийся таг, и для него заводится таймер, по обнулении которого таг считается вышедшим из зоны чтения (потерянным), и, соответственно, удаляется из памяти. Если таг с зафиксированным в памяти номером обнаружен считывателем до обнуления таймера, то таймер вновь перевзводится на полное время. Это время далее именуется **временем памяти тага в радиоканале**.

Данное время определяет помехозащищенность системы (чем больше время, тем меньше возникает ложных пропаданий тага с последующим его появлением). В то же время, чрезмерно большое время памяти повышает инерционность (в части фиксации выхода тага из зоны чтения). Данный параметр является программируемым для адаптации к конкретным условиям применения системы. На практике можно рекомендовать устанавливать это время не менее 10...20 периодов обмена тага со считывателем, то есть при периоде 0,5 секунды это время желательно устанавливать в пределах от 5 до 10 секунд.

В системе команд считывателя имеется функция для установки времени памяти тага в радиоканале.

Стек тагов

Для каждого канала имеется свой выделенный стек на 64 тага. Каждая запись содержит код (ID) тага, счетчик времени (время памяти тага), а также статусную информацию (в частности, признак разряженной батарейки).

Каждый прочитанный радиоканалом таг помещается в стек (если таг уже имеется в стеке, то перевзводится его время памяти). При достижении счетчиком времени нуля формируется событие пропадания тага из поля чтения.

Стек просматривается каждые 100 мсек.

Память транзакций

Для использования в режиме мониторинга, а также для реализации функции аудита (использование в системе доступа) считыватель имеет встроенную энергонезависимую память событий (транзакций). Буфер транзакций организован в виде кольцевого буфера. Все факты появления и исчезновения тагов из поля считывателя, срабатывание датчиков автоматики и так далее заносится в эту память вместе с текущими датой и временем. Если кольцевой буфер переполняется, то первыми затираются наиболее старые события.

Размер кольцевого буфера составляет 1000 событий. При получении транзакций от считывателя первыми передаются наиболее старые транзакции. После передачи транзакции ведущему (по его запросу) транзакция из памяти считывателя удаляется.

Буфер транзакций является общим для обоих каналов считывателя.

Для манипулирования с буфером транзакций имеется специальный набор функций в системе команд считывателя.

Конструктивное исполнение

Считыватель выполнен в пластиковом герметичном корпусе с классом защиты IP-67 для уличного применения. Для подключения внешних антенн к обоим каналам имеются два антенных разъема, а для ввода кабелей управления и Ethernet имеется три гермоввода.

Считыватель имеет светодиодную индикацию (3 сверхъярких светодиода), что позволяет диагностировать режимы работы и текущее состояние считывателя непосредственно «в поле» даже при солнечной засветке.

Физический интерфейс

Общие положения

В данном разделе рассмотрен формат пакетов, которыми обменивается считыватель с ведущим (например, с ПК) по интерфейсам RS-485 и Ethernet. При этом логический формат пакетов для обоих интерфейсов является одинаковым, хотя физический формат несколько отличается в силу отличия среды передачи — для RS-485 добавляются коды начала и конца пакета, что позволяет проще осуществлять прием и дешифрацию пакета.

Кроме того, описанная ниже процедура перекодирования байтов с кодами 0xFA и выше используется и в протоколе Ethernet для упрощения написания драйвера.

Считыватель всегда является ведомым, компьютер (ПК или другое внешнее по отношению к считывателю устройство) – ведущим.

Ведущий формирует запросы к считывателю, а считыватель обязан на каждый запрос отправить ответ. Ниже рассмотрены форматы пакетов запроса мастера и пакеты ответов ведомого.

Параметры интерфейса

Для работы со считывателем по интерфейсу RS-485 необходимо установить следующие параметры обмена:

- Скорость 9600 бод
- Длина данных 8 бит
- Один стоповый бит
- Без контроля четности

При использовании интерфейса Ethernet применяется протокол UDP, для обмена открываются порты 8872 (на приеме в считывателе) и 8873 (на приеме у ведущего).

Адресация

На одну линию RS-485 можно подключить до 30 считывателей (ограничение связано с физической нагрузочной способностью микросхем драйверов линии). При этом каждый считыватель должен иметь уникальный адрес в пределах от 1 до 254.

Нулевой адрес зарезервирован, а адрес 255 (0xFF) является общим адресом (broadcast). При передаче команды по общему адресу все считыватели принимают такую команду, но ни один из них не отвечает на команду.

Общий адрес может использоваться, например, для одновременной синхронизации встроенных часов реального времени, а также для смены адреса считывателя на новый известный, если текущий адрес считывателя по каким-то причинам не известен.

Во время операции по смене адреса считывателя с использованием общего адреса на шине RS-485 должен быть подключенным только один считыватель!

Смена текущего адреса считывателя производится программно, при этом считыватель запоминает свой адрес во встроенной энергонезависимой памяти.

Физический уровень

Обмен между ведущим и считывателем ведется пакетами, при этом со стороны ведущего формируется пакет запроса, а считыватель на него формирует пакет ответа. На физическом уровне пакет имеет нижеследующую структуру:

Таблица 1

Номер байта	Ведущий	Ведомый	Комментарий
0	0xFB	0xFC	Символ начала пакета
1	0x??	0x??	Номер пакета
2	0x??	0x??	Адрес ведомого (RS-485)
3	0x??	0x??	Тип пакета
4	0x??	0x??	Имя компонента
5	0x??	0x??	Номер компонента
6	0x??	0x??	Имя подкомпонента
7	0x??	0x??	Номер подкомпонента
6	0x??	0x??	Код команды
7	0x??	0x??	Код результата
8	0x??	0x??	Длина данных
			Опциональные данные (дина = M)
8 + M	0x??	0x??	Контрольная сумма
9 + M	0xFD	0xFE	Символ конца пакета

- Для интерфейса RS-485 каждый пакет начинается символом начала и завершается символом конца. Данные значения зарезервированы только для этих целей. Если внутри пакета между символами начала и конца встречается байт со значением большим или равным, чем 0xFA, то такой байт заменяется на два байта [0xFA, X], где X есть заменяемый байт минус 0xF0. Замена байт осуществляется для данных всего пакета, включая поле контрольной суммы.

Пример функций кодирования и декодирования пакетов приведен в Приложении 1.

- Каждый пакет содержит поле с контрольной суммой пакета. Контрольная сумма CRC-8 вычисляется для всех полей, кроме символов начала и конца пакета, до замены спецсимволов на двухбайтные коды. Код для вычисления контрольной суммы приведен в Приложении 2. Начальное значение двухбайтовой контрольной суммы равно 0x5A. Вместе с байтом CRC результат вычисления должен быть равным нулю.
- Все запросы ведущего нумеруются (поле номера пакета), причем после успешного обмена с ведущим мастер инкреминирует номер пакета. Ведомый при ответе всегда ставит тот номер пакета, на который он отвечает.

Для интерфейса Ethernet физический формат пакета такой же, как описано выше, за исключением отсутствия символов начала и конца пакета, которые нужны в RS-485 для пакетной синхронизации при обмене. При использовании Ethernet поле адреса в пакете сохраняется, но не используется, так как адресация осуществляется по IP адресу и номеру порта.

Выделенные желтым цветом поля образуют пакет формата приложения, и дальнейшее рассмотрение команд будем производить, полагая, что нижний уровень драйвера декодирует байтстаффинг (замену однобайтовых символов двухбайтовыми и наоборот), проверяет CRC и отбрасывает ненужные прикладному уровню данные.

Прикладной уровень

Ниже рассмотрим основные компоненты пакета прикладного уровня.

Поле «тип пакета»

Типы пакетов и их назначение приведено в следующей таблице:

Таблица 2

Мнемоника	Значение	Комментарий
PACK_TYPE_EVENT	0x80	Пакет в формате внутреннего эвента
PACK_TYPE_CONFIG	0x81	Пакет с данными конфигурации
PACK_TYPE_COMMAND	0x82	Пакет содержит команду встроенному контроллеру
PACK_TYPE_GET_DATA	0x83	Пакет содержит статус контроллера
PACK_TYPE_TRANSACT	0x84	Пакет является транзакцией от считывателя

Адресация компонентов

Для понимания работы прикладного уровня надо представлять внутреннюю программную модель считывателя, основанную на объектной модели программирования. Используемые в качестве программных объектов классы образуют иерархическую структуру, и каждый объект имеет в ней свое имя и номер (например, «Радиоканал», «номер 1». Именно в связи с этим в пакете присутствуют поля имени компонента и подкомпонента, которым адресуются команды.

Обмен информацией между компонентами происходит преимущественно через общую очередь событий (эвентов), также имеющих в себе имена источников и/или получателей, поэтому часть команд считывателя имеет формат внутреннего эвента для непосредственного помещения в эту очередь.

Использование внутренних адресов компонентов будет понятнее при рассмотрении конкретных команд.

Коды команд и результата операции

Это следующие два поля пакета. Первое из них содержит код команды от ведущего считывателю, а второе — результат выполнения операции считывателем. При успешном выполнении операции код результата равен нулю.

Поле длины и данных

Данные поля содержат соответственно длину данных в пакете и опционально сами данные. При отсутствии данных поле длины равно нулю. Формат данных определяется назначением команды.

Ответы считывателя

При ответе на команду считыватель не меняет никакие поля пакета, кроме поля результата и данных, если они присутствуют в ответе на команду,поэтому из ответа сразу понятно, на какую команды отвечает считыватель.

Если в поле результата стоит значение, отличное от нуля, то никаких корректных данных считыватель не возвращает.

Команды считывателя

Коды ошибок

При некорректных командах считыватель формирует в ответном пакете код ошибки (поле результата). Список основных ошибок приведен в таблице ниже.

Таблица 3:

Мнемоника	Значение	Примечание
ERR_NO_ERROR	0x00	Команда выполнена без ошибок
ERR_INVALID_PARAM	0x10	Неправильное количество или значение параметров команды
ERR_INVALID_COMMAND	0x11	Неизвестная команда

Другие коды ошибок формирует драйвер ведущего (некорректная контрольная сумма, таймаут обмена).

Синхронизация часов реального времени

Назначение

Команда предназначена для синхронизации встроенных энергонезависимых (с питанием от резервной батарейки) часов реального времени.

Формат пакета

Таблица 4:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_COMMAND	0x82	
Имя компонента	NAME_CONTROLLER	0xC0	
Номер компонента	1		
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_SYNC_CLOCK	0xC2	
Результат	0		Ставится нулевым
Длина данных	8		
Данные	Время и дата по 4 байта		Внутренний формат

Время и дата представлены 4-х байтовыми значениями (uint32_t) во внутреннем формате считывателя:

Таблица 5:

Байт	MSB (3)	2	1	LSB (0)
Назначение	День недели	Часы	Минуты	Секунды

Таблица 6:

Байт	MSB (3)	2	1	LSB (0)
Назначение	Год		Месяц	День

Ответ считывателя

В ответном пакете считыватель возвращает пакет без данных с кодом результата, равным нулю (при корректных данных от ведущего), либо код ошибки, если данные в команде некоректны.

Получение текущих даты и времени

Назначение

Команда предназначена для получения из встроенных энергонезависимых часов считывателя информации о текущих дате и времени.

Формат пакета

Таблица 7:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_CONTROLLER	0xC0	
Номер компонента	1		
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_DATE_TIME	0x66	
Результат	0		Ставится нулевым
Длина данных	0		
Данные			Данных нет

Ответ считывателя

В ответ считыватель передает значение текущего времени и даты в формате, аналогичном формату ведущего:

Таблица 8:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	

Поле	Мнемоника	Значение	Примечание
Имя компонента	NAME_CONTROLLER	0xC0	
Номер компонента	1		
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_DATE_TIME	0x66	
Результат	0		
Длина данных	8		
Данные	Время и дата по 4 байта		Внутренний формат

Получение числа тагов в стеке

Назначение

Команда предназначена для получения текущего количества тагов в стеках радиоканалов считывателя.

Формат пакета

Таблица 9:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_TAG_STACK	0x1A	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_NUM_TAGS	0x61	
Результат	0		Ставится нулевым
Длина данных	0		
Данные			Данных нет

Ответ считывателя

Считыватель отвечает пакетом следующего формата:

Таблица 10:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_TAG_STACK	0x1A	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	0		

Поле	Мнемоника	Значение	Примечание
Номер подкомпонента	0		
Команда	CMD_GET_NUM_TAGS	0x61	
Результат	0		Ставится нулевым
Длина данных	2		
Данные	Количество тагов		Младший байт первым

Команда получения первого тага из стека

Назначение

Команда предназначена для получения первого тага из стека заданного радиоканала. Одновременно внутренний указатель позиционируется на начало стека.

Команда обязательно применяется перед последующими командами получения следующего тага из стека.

Формат пакета

Таблица 11:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_TAG_STACK	0x1A	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_FIRST_TAG	0x62	
Результат	0		Ставится нулевым
Длина данных	0		
Данные			Данных нет

Ответ считывателя

Считыватель отвечает пакетом следующего формата:

Таблица 12:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_TAG_STACK	0x1A	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_FIRST_TAG	0x62	

Поле	Мнемоника	Значение	Примечание
Результат	0		Ставится нулевым
Длина данных	4		
Данные	ID первого в стеке ага		Младший байт первым

Если тагов в стеке нет, то в качестве ID передается нулевое значение.

Команда получения следующего тага из стека

Назначение

Команда предназначена для получения кода (ID) следующего тага из стека заданного радиоканала. Перед первым применением обязательно использование команды .

Формат пакета

Таблица 13:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_TAG_STACK	0x1A	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_NEXT_TAG	0x63	
Результат	0		Ставится нулевым
Длина данных	0		
Данные			Данных нет

Ответ считывателя

Таблица 14:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_TAG_STACK	0x1A	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_NEXT_TAG	0x63	
Результат	0		Ставится нулевым
Длина данных	4		
Данные	ID следующего тага в стеке		Младший байт первым

Команда получения затухания радиоканала

Назначение

Команда предназначена для получения текущего затухания заданного радиоканала.

Формат пакета

Таблица 15:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_RFCHANNEL	0x19	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_DUMPING	0x65	
Результат	0		Ставится нулевым
Длина данных	0		
Данные			Данных нет

Ответ считывателя

Таблица 16:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_RFCHANNEL	0x19	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_DUMPING	0x65	
Результат	0		Ставится нулевым
Длина данных	2		
Данные	Затухание от 0 до 31 дБ		Младший байт первым

Команда установки затухания радиоканала

Назначение

Команда предназначена для установки затухания в заданном радиоканале.
Установленное значение сохраняется в энергонезависимой памяти считывателя.

Формат пакета

Таблица 17:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_CONFIG	0x81	
Имя компонента	NAME_RFCHANNEL	0x19	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	NAME_PARAM_DUMPING	0x15	
Номер подкомпонента	1		
Команда	EV_SET_CONFIG	0x76	
Результат	0		Ставится нулевым
Длина данных	2		
Данные	Затухание от 0 до 31 дБ		Младший байт первым

Ответ считывателя

Считыватель отвечает пакетом без данных. Если задан неправильный номер радиоканала или значение затухания выходит за допустимый диапазон, то возвращается код ошибки.

Команда получения времени памяти тага

Назначение

Команда предназначена для получения времени памяти тага из заданного радиоканала.

Формат пакета

Таблица 18:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_TAG_STACK	0x1A	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_TAGMEMTIME	0x64	
Результат	0		Ставится нулевым
Длина данных	0		
Данные			Данных нет

Ответ считывателя

Таблица 19:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_TAG_STACK	0x1A	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_TAGMEMTIME	0x64	
Результат	0		Ставится нулевым
Длина данных	2		
Данные	Время памяти тага		Младший байт первым

Время памяти тага в считывателе устанавливается в квантах по 100 мсек.

Команда установки времени памяти тага

Назначение

Команда предназначена для установки времени памяти тага в стеке заданного радиоканала. Значение запоминается в энергонезависимой памяти считывателя.

Формат пакета

Таблица 20:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_CONFIG	0x81	
Имя компонента	NAME_TAG_STACK	0x1A	
Номер компонента	1 или 2		Номер стека радиоканала
Имя подкомпонента	NAME_PARAM_TIME	0x15	
Номер подкомпонента	1		
Команда	EV_SET_CONFIG	0x76	
Результат	0		Ставится нулевым
Длина данных	4		
Данные	От 1 до 65000		Младший байт первым

Ответ считывателя

Считыватель отвечает пакетом без данных. Если задан неправильный номер стека, то возвращается код ошибки.

Команда получения транзакции

Назначение

Команда предназначена для для получения транзакции из кольцевого энергонезависимого буфера считывателя. Первыми передаются наиболее старые транзакции из буфера. Транзакции могут быть с данными или без данных

Формат пакета

Таблица 21:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_TRANSACT	0x84	
Имя компонента	NAME_CONTROLLER	0xC0	
Номер компонента	1		
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_TRANSACTION	0x41	
Результат	0		Ставится нулевым
Длина данных	0		
Данные	Нет		

Ответ считывателя

Если буфер транзакций пуст, то считыватель отвечает пустым (с нулевой длиной данных) пакетом. Если транзакция в буфере имеется, то длина данных в ответе отлична от нуля.

Транзакции могут быть двух типов: с опциональными данными или без данных (только общая часть). В общем виде формат ответа выглядит как показано ниже.

Таблица 22:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_TRANSACT	0x84	
Имя компонента	NAME_CONTROLLER	0xC0	
Номер компонента	1		
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_TRANSACTION	0x41	
Результат	0		Ставится нулевым
Длина данных			
Данные	Короткая или длинная		

Данные транзакции содержат следующие поля:

Таблица 23:

Поле	Длина	Примечание
Адрес	1	Адрес считывателя
Имя источника	1	Имя и номер компонента в считывателе
Номер источника	1	
Номер детали	1	Имя и номер подкомпонента
Имя детали	1	
Код транзакции	2	Младший байт первым
Год	2	
Месяц	1	
День	1	
Часы	1	
Минуты	1	
Секунды	1	
Длина дополнительных данных	1	0, 4 или 8 байтов
Код тага	4	
Дополнительные данные	4	

Возможные коды транзакций приведены в следующей таблице.

Таблица 24:

Мнемоника	Код	Примечание
TRN_NEW_TAG_FOUND	0x3001	Обнаружен новый таг в радиоканале
TRN_TAG_LOSS	0x3002	Таг потерян из поля считывателя
TRN_DC_IS_ON	0x3003	Датчик ворот включился
TRN_DC_IS_OFF	0x3004	Датчик ворот выключился
TRN_SENS_IS_ON	0x3005	Датчик присутствия сработал
TRN_SENS_IS_OFF	0x3006	Датчик присутствия восстановился
TRN_SEND_TAG_1	0x3007	Код тага канала 1 отправлен в контроллер
TRN_SEND_TAG_2	0x3008	Код тага канала 2 отправлен в контроллер
TRN_LOCKED	0x3009	Канал считывателя заблокирован
TRN_UNLOCKED	0x300A	Канал считывателя разблокирован
TRN_SET_ON	0x300B	*Транзакция старого протокола*
TRN_SET_OFF	0x300C	*Транзакция старого протокола*
TRN_TAG_IN_OPPOSIT	0x300D	Таг обнаружен в противоположном канале
TRN_STACK_CLEARED	0x300E	Стек тагов очищен
TRN_TAG_REMOVED	0x300F	Таг удален из стека
TRN_WAIT_CAR	0x3010	Состояние ожидания автомобиля на датчике
TRN_WAIT_ACCESS	0x3011	Состояние предоставления доступа от контроллера
TRN_WAIT_DEF_GATE	0x3012	*Транзакция старого протокола*

Мнемоника	Код	Примечание
TRN_WAIT_GATE_CLOSE	0x3013	*Транзакция старого протокола*
TRN_ACCESS_FINISH	0x3014	Цикл обработки доступа закончен
TRN_CAR_NOT_FOUND	0x3015	В заданное время автомобиль не обнаружен
TRN_GATE_NOT_OPEN	0x3016	Ворота не были открыты контроллером
TRN_GATE_NOT_CLOSE	0x3017	Не закрыты ворота (шлагбаум)
TRN_WAIT_DEF_ACCESS	0x3018	Ожидание цикла доступа (без датчиков)
TRN_STACK_OVERFLOW	0x3019	Нет места в стеке для занесения тага

Команда установки режима работы

Назначение

Команда предназначена для переключения режима работы считывателя.

Формат пакета

Таблица 25:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_CONFIG	0x81	
Имя компонента	NAME_CONTROLLER	0xC0	
Номер компонента	1		
Имя подкомпонента	NAME_PARAM_MODE	0x13	
Номер подкомпонента	1		
Команда	EV_SET_CONFIG	0x76	
Результат	0		Ставится нулевым
Длина данных	2		
Данные	Номер режима		Младший байт первым

Номер режима в данных задается в соответствии со следующей таблицей:

Таблица 26:

Название режима	Значение
Режим мониторинга	0x00
Режим однократного чтения	0x01
Режим регистрации на проезд	0x02
Режим без ворот	0x03
Расширенный режим	0x04
Режим контроллера (не используется)	0x05

Ответ считывателя

Считыватель отвечает пакетом без данных. Если задан неправильный номер режима или другие параметры команды, то возвращается код ошибки.

Команда получения текущего режима

Назначение

Команда предназначена для получения значения текущего режима работы считывателя.

Формат пакета

Таблица 27:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_CONTROLLER	0xC0	
Номер компонента	1		
Имя подкомпонента	NAME_PARAM_MODE	0x13	
Номер подкомпонента	1		
Команда	CMD_GET_DEVICE_MODE	0x6E	
Результат	0		Ставится нулевым
Длина данных	0		
Данные	Данных нет		

Ответ считывателя

Таблица 28:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_GET_DATA	0x83	
Имя компонента	NAME_CONTROLLER	0xC0	
Номер компонента	1		
Имя подкомпонента	NAME_PARAM_MODE	0x13	
Номер подкомпонента	1		
Команда	CMD_GET_DEVICE_MODE	0x6E	
Результат	0		Ставится нулевым
Длина данных	2		
Данные	Номер режима		Младший байт первым

Команда очистки буфера транзакций

Назначение

Команда предназначена для очистки внутреннего буфера транзакций (например, когда за время автономной работы буфер заполнился, и нет необходимости «выкачивать» все устаревшие транзакции).

Формат пакета

Таблица 29:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_COMMAND	0x82	
Имя компонента	NAME_CONTROLLER	0xC0	
Номер компонента	1		
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_CLEAR_TRANSACT	0xA2	
Результат	0		Ставится нулевым
Длина данных	0		
Данные	Данных нет		

Ответ считывателя

Считыватель отвечает пакетом без данных. Если заданы неправильные параметры команды, то возвращается код ошибки.

Команда получения числа транзакций

Назначение

Команда предназначена для получения числа транзакций, находящихся в данный момент в кольцевом буфере считывателя.

Формат пакета

Таблица 30:

Поле	Мнемоника	Значение	Примечание
Тип пакета	PACK_TYPE_COMMAND	0x82	
Имя компонента	NAME_CONTROLLER	0xC0	
Номер компонента	1		
Имя подкомпонента	0		
Номер подкомпонента	0		
Команда	CMD_GET_TRAN_COUNT	0x72	

Поле	Мнемоника	Значение	Примечание
Результат	0		Ставится нулевым
Длина данных	0		
Данные	Данных нет		

Ответ считывателя

Считыватель отвечает пакетом без данных. Если заданы неправильные параметры команды, то возвращается код ошибки.

Приложение 1

Кодирование и декодирование пакетов

Кодирование пакета

```
void tBaseHost::EncodePacket(BYTE *src, BYTE *dest, int *len)
{
int sptr = 0;
int dptr = 0;

  while (sptr < *len) {
    if (src[sptr] >= 0xFA) {
      dest[dptr++] = 0xFA;
      dest[dptr++] = src[sptr] & 0x0F;
      sptr++;
    }
    else {
      dest[dptr++] = src[sptr++];
    }
  }
  *len = dptr++;
}
```

Декодирование пакета

```
void tBaseHost::DecodePacket(BYTE *data, int *len)
{
int sptr = 0;
int dptr = 0;

  while (sptr < *len) {
    if (data[sptr] == 0xFA) {
      sptr++;
      data[dptr] = data[sptr] + 0xF0;
      sptr++; dptr++;
    }
    else {
      data[dptr] = data[sptr];
      sptr++; dptr++;
    }
  }
  *len = dptr; //++;
}
```

Приложение 2

Вычисление контрольной суммы пакета

```
const static uint8_t CRCTBL[256] = {
    0,  94, 188, 226,  97,  63, 221, 131, 194, 156, 126,  32, 163, 253,  31,  65,
  157, 195,  33, 127, 252, 162,  64,  30,  95,   1, 227, 189,  62,  96, 130, 220,
   35, 125, 159, 193,  66,  28, 254, 160, 225, 191,  93,   3, 128, 222,  60,  98,
  190, 224,   2,  92, 223, 129,  99,  61, 124,  34, 192, 158,  29,  67, 161, 255,
   70,  24, 250, 164,  39, 121, 155, 197, 132, 218,  56, 102, 229, 187,  89,   7,
  219, 133, 103,  57, 186, 228,   6,  88,  25,  71, 165, 251, 120,  38, 196, 154,
  101,  59, 217, 135,   4,  90, 184, 230, 167, 249,  27,  69, 198, 152, 122,  36,
  248, 166,  68,  26, 153, 199,  37, 123,  58, 100, 134, 216,  91,   5, 231, 185,
  140, 210,  48, 110, 237, 179,  81,  15,  78,  16, 242, 172,  47, 113, 147, 205,
   17,  79, 173, 243, 112,  46, 204, 146, 211, 141, 111,  49, 178, 236,  14,  80,
  175, 241,  19,  77, 206, 144, 114,  44, 109,  51, 209, 143,  12,  82, 176, 238,
   50, 108, 142, 208,  83,  13, 239, 177, 240, 174,  76,  18, 145, 207,  45, 115,
  202, 148, 118,  40, 171, 245,  23,  73,   8,  86, 180, 234, 105,  55, 213, 139,
   87,   9, 235, 181,  54, 104, 138, 212, 149, 203,  41, 119, 244, 170,  72,  22,
  233, 183,  85,  11, 136, 214,  52, 106,  43, 117, 151, 201,  74,  20, 246, 168,
  116,  42, 200, 150,  21,  75, 169, 247, 182, 232,  10,  84, 215, 137, 107,  53
  };
```

Начальное значение CRC = 0x5A
Для вычисления CRC можно использовать такую функцию:

```
uint8_t RecCRC_OK(uint8_t *data, uint8_t len) {
int i;
BYTE crc;

  crc = INITIAL_CRC;
  for (i = 0; i < len; i++) {
    crc = CRCTBL[crc ^ data[i]];
  }
  if (crc == 0) {
    return(1);
  }
  else {
    return(0);
  }
}
```

Для заметок