

Система контроля и управления
доступом «Сфинкс».

Руководство по программному
использованию контрольного считывателя
«Sphinx Reader EN»

Оглавление

1. Введение.....	3
2. Состав пакета разработчика.....	4
3. Разработка на C/C+.....	5
3.1 Пример проекта.....	5
3.2 Разработка собственных проектов.....	5
3.3 Интерфейс библиотеки «spnreader».....	6
3.3.1 Функция SpnxReaderOpen.....	7
3.3.2 Функция SpnxReaderClose.....	7
3.3.3 Функция SpnxReaderReceiveW26.....	7
3.3.4 Функция SpnxReaderReceiveW26T.....	8
4. Разработка на Java.....	10
4.1 Пример проекта.....	10
4.2 Разработка собственных проектов.....	10
4.3 Интерфейс библиотеки «spnreader».....	11
4.3.1 Метод open.....	12
4.3.2 Метод close.....	12
4.3.3 Метод receiveW26.....	12
4.3.4 Метод receiveW26T.....	13
5. Разработка на других языках.....	14
6. Получение технической поддержки.....	15

1. Введение

Данный документ содержит сведения, необходимые для программной работы с контрольным считывателем «Sphinx Reader EH» (далее - «устройство»), производимым ООО «ПромАвтоматика» и входящим в программно-аппаратный комплекс СКУД «Сфинкс».

Устройство представляет собой считыватель настольного исполнения, позволяющий считывать бесконтактные карты доступа стандартов «EM-Marine» и «HID». Устройство подключается к компьютеру по USB интерфейсу. Программная работа с устройством позволяет получать коды считываемых им карт доступа.

ООО «ПромАвтоматика» поставляет комплект разработчика, позволяющий осуществлять работу с устройством из следующих сред:

- Из C/C++ приложений на операционной системе Windows XP/2000/2003/Vista.
- Из Java приложения на операционной системе Windows XP/2000/2003/Vista.
- Из Win32-приложения, разработанного на других языках, при запуске на операционной системе Windows XP/2000/2003/Vista.

Использование данного пакета разработчика является предпочтительным способом написания приложений, работающих с устройствами «Sphinx Reader EH», т.к. в рамках интерфейса пакета прозрачным для разработчика образом реализуется поддержка различных исполнений устройств.

Более подробно о контрольном считывателе можно узнать из документации, которая доступна на диске, поставляемом с устройством, а также может быть получена с сайта производителя на странице <http://www.spx.ru/docs.php>.

2. Состав пакета разработчика

Установленный пакет разработчика содержит следующие каталоги:

Каталог	Содержимое
sdk	<p>Непосредственно файлы пакета разработчика:</p> <ul style="list-style-type: none">● spnxreader.dll● spnxreader.h● spnxreader.jar● spnxreader.lib● FTD2XX.dll <p>Метод использования этих файлов описан в последующих разделах данного документа.</p>
sdk_doc	Данный документ.
examples	<p>Примеры использования пакета разработчика:</p> <ul style="list-style-type: none">● cpp_example● java_example <p>Эти примеры описаны в последующих разделах данного документа.</p>

3. Разработка на C/C++

3.1 Пример проекта

Каталог «examples/cpp_example» содержит пример проекта, написанного с использованием данного пакета разработчика.

Проект представляет собой тестовое консольное win32-приложение, которое последовательно выводит на стандартный вывод коды карт, читаемые подключенным контрольным считывателем.

Для запуска прекомпилированной версии приложения перейдите в каталог «examples/cpp_example/Release» и запустите исполняемый файл «cpp_example.exe».

В случае, если контрольный считыватель недоступен (не подключен, не установлен требуемый драйвер устройства или устройство уже задействовано другим процессом), то приложение завершит свою работу с сообщением об ошибке.

Пример работы приложения:

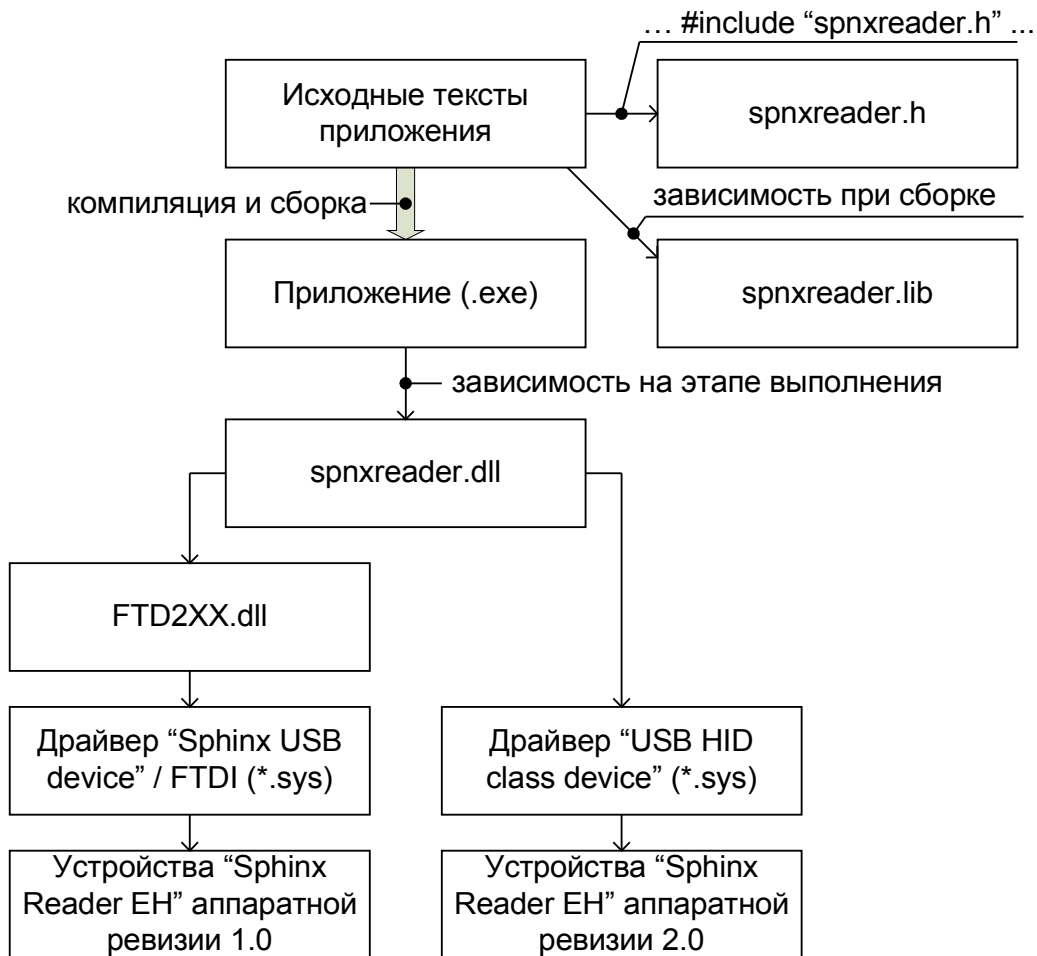
```
> cpp_example.exe
156,21837
102,38108
075,06712
075,06711
075,06710
```

3.2 Разработка собственных проектов

Для использования пакета разработчика требуется предпринять следующие шаги:

- Указать в параметрах сборки проекта на необходимость использовать каталог sdk для поиска заголовочных и библиотечных файлов.
- Включить файл «spxreader.h» в модули, которые будут получать данные от контрольного считывателя (#include "spxreader.h"). Реализовать работу с функциями пакета (см. нижеследующий раздел)
- Добавить «spxreader.lib» в список библиотек, используемых при сборке проекта.
- Обеспечить доступность файлов «spxreader.dll» и «FTD2XX.dll» на момент запуска скомпилированного приложения.

На нижеследующей картинке схематично показан процесс сборки и запуска проекта, использующего данный пакет разработчика.



Элементы рисунка ниже «sphinxreader.dll» приведены для ознакомления. Для работы с устройством понимание внутренних аспектов функционирования библиотеки «sphinxreader» не требуется.

Смотрите также прилагаемый пример проекта, находящийся в каталоге «examples/cpp_example».

3.3 Интерфейс библиотеки «sphinxreader»

Заголовочный файл «sphinxreader.h» содержит объявление следующего типа данных:

Тип данных	Использование
SPNX_HANDLE	Указатель (handle) на устройство, с которым ведется работа.

Также библиотека «sphinxreader» экспортирует следующие функции:

Функция	Использование
SphinxReaderOpen	Пытается «открыть» подключенный к системе контрольный считыватель.
SphinxReaderClose	«Закрывает» ранее открытый функцией SphinxReaderOpen контрольный считыватель.

SpnxReaderReceiveW26 и SpnxReaderReceiveW26T	Пытается получить следующий Wiegand-26 код от ранее открытого контрольного считывателя.
--	---

Эти функции подробно рассмотрены ниже.

3.3.1 Функция SpnxReaderOpen

Прототип:

```
SPNXREADER_API BOOL SpnxReaderOpen (SPNX_HANDLE *pxHandle);
```

Описание:

Пытается «открыть» подключенный к системе контрольный считыватель.

Параметры:

Параметр	Использование
pxHandle	Указатель на переменную типа SPNX_HANDLE, куда будет записан handle устройства в случае его успешного открытия.

Возвращаемое значение:

Возвращает TRUE, если открыть устройство удалось, и его handle был записан в *pxHandle.

Возвращает FALSE, если открыть устройство не удалось.

3.3.2 Функция SpnxReaderClose

Прототип:

```
SPNXREADER_API void SpnxReaderClose (SPNX_HANDLE *pxHandle);
```

Описание:

«Закрывает» ранее открытый функцией SpnxReaderOpen контрольный считыватель.

Параметры:

Параметр	Использование
pxHandle	Указатель на переменную типа SPNX_HANDLE, содержащую handle, полученный ранее с помощью функции SpnxReaderOpen. После закрытия устройства переменная будет модифицирована в знак того, что она более не указывает на открытое устройство.

Возвращаемое значение:

Нет.

3.3.3 Функция SpnxReaderReceiveW26

Прототип:

```
SPNXREADER_API BOOL SpnxReaderReceiveW26 (SPNX_HANDLE xHandle, LPVOID pBuffer, DWORD nBufferSize);
```

Описание:

Пытается получить следующий Wiegand-26 код от ранее открытого контрольного считывателя.

Функция блокирует вызвавший ее поток до момента получения кода или наступления ошибки. В случае если функция возвращает FALSE, это может означать одно из следующих событий:

- Переданный буфер слишком мал (см. ниже)
- Ошибка информационного обмена с устройством. Это, вероятно, означает, что USB устройство было отключено или его работа была приостановлена. Устройство в таком случае следует закрыть функцией SpnxReaderClose, после чего можно повторно попытаться его открыть через некоторое время.

Параметры:

Параметр	Использование
pxHandle	Указатель на переменную типа SPNX_HANDLE, содержащую handle, полученный ранее с помощью функции SpnxReaderOpen.
pBuffer	Указатель на буфер, куда будет записан полученный код.
nBufferSize	Длина буфера. Должна составлять минимум 3 байта.

Возвращаемое значение:

Возвращает TRUE в случае успеха. В таком случае в pBuffer был помещен полученный код.

Возвращает FALSE, если получить код не удалось.

3.3.4 Функция SpnxReaderReceiveW26T

Прототип:

```
SPNXREADER_API DWORD SpnxReaderReceiveW26T(SPNX_HANDLE xHandle, LPVOID pBuffer, DWORD nBufferSize,
DWORD nTimeout);
```

Описание:

Пытается получить следующий Wiegand-26 код от ранее открытого контрольного считывателя.

Функция блокирует вызвавший ее поток до наступления одного из следующих событий:

- Получен код.
- Наступила ошибка работы с устройством. Это, означает одно из следующих вариантов:
 - USB устройство было отключено или его работа была приостановлена. Устройство в таком случае следует закрыть функцией SpnxReaderClose, после чего можно повторно попытаться его открыть через некоторое время.
 - Переданный буфер слишком мал (см. ниже)
- Истек таймаут ожидания кода, но код так и не был получен.

Параметры:

Параметр	Использование
pxHandle	Указатель на переменную типа SPNX_HANDLE, содержащую handle, полученный ранее с помощью функции SpnxReaderOpen.

pBuffer	Указатель на буфер, куда будет записан полученный код.
nBufferSize	Длина буфера. Должна составлять минимум 3 байта.
nTimeout	Таймаут ожидания кода в миллисекундах.

Возвращаемое значение:

0, если истек таймаут ожидания, но код так и не был получен.

-1, если наступила ошибка (см. выше)

Кол-во полученных байт кода, если код получен.

4. Разработка на Java

4.1 Пример проекта

Каталог «examples/java_example» содержит пример проекта, написанного с использованием данного пакета разработчика.

Проект представляет собой тестовое приложение, которое последовательно выводит на стандартный вывод коды карт, читаемые подключенным контрольным считывателем.

Для запуска прекомпилированной версии приложения перейдите в каталог «examples/java_example» и запустите исполняемый файл «java_example.bat». Для запуска прекомпилированной версии потребуется JRE SE 1.6. Если JRE каталог «bin» не находится в составе путей поиска исполняемых файлов (переменная окружения «PATH»), то в файле «java_example.bat» требуется указать полный путь к файлу «bin/java.exe» из состава JRE.

Если контрольный считыватель недоступен (не подключен, не установлен требуемый драйвер устройства или устройство уже задействовано другим процессом), то приложение завершит свою работу с сообщением об ошибке.

Пример работы приложения:

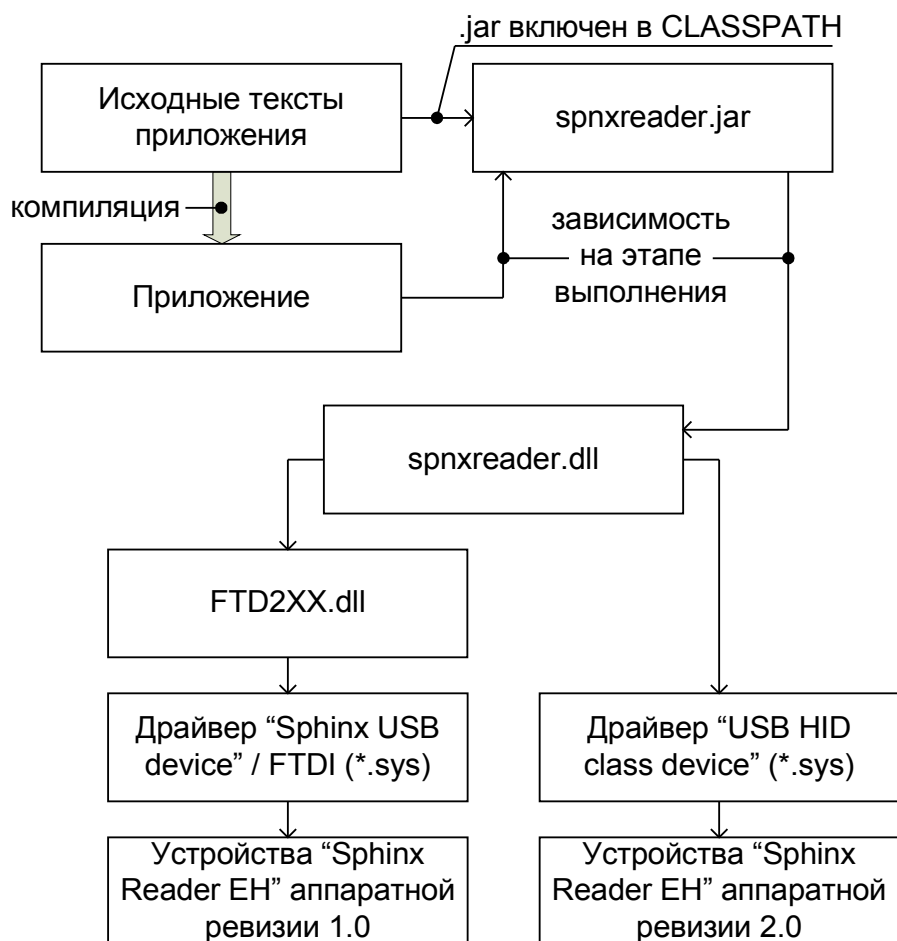
```
> java_example.bat
075,06712
075,06711
075,06710
156,21837
102,38108
```

4.2 Разработка собственных проектов

Для использования пакета разработчика требуется предпринять следующие шаги:

- Включить файл «spnхreader.jar» в список CLASSPATH вашего проекта.
- Реализовать работу с функциями пакета (см. нижеследующий раздел), используя методы класса «SpnхReader» из пакета «spnхsdk».
- Обеспечить доступность файлов «spnхreader.dll» и «FTD2XX.dll» на момент запуска приложения.

На нижеследующей картинке схематично показан процесс сборки и запуска проекта, использующего данный пакет разработчика



Элементы рисунка ниже «sphinxreader.dll» приведены для ознакомления. Для работы с устройством понимание внутренних аспектов функционирования библиотеки «sphinxreader» не требуется.

Смотрите также прилагаемый пример проекта, находящийся в каталоге «examples/java_example».

4.3 Интерфейс библиотеки «sphinxreader»

«sphinxreader.jar» содержит пакет «sphinxsdk», а в нем класс «SpnxReader».

Методы класса «SpnxReader»:

Метод	Использование
open	Пытается «открыть» подключенный к системе контрольный считыватель. Создает и возвращает экземпляр класса для работы с открытым устройством.
close	«Закрывает» ранее открытое устройство.
receiveW26 и receiveW26T	Пытается получить следующий Wiegand-26 код от ранее открытого устройства.

Эти методы подробно рассмотрены ниже.

4.3.1 Метод open

Объявление:

```
public static SpnxReader open()
```

Описание:

Пытается «открыть» подключенный к системе контрольный считыватель.

Возвращаемое значение:

Возвращает экземпляр класса, если открыть устройство удалось.

Возвращает null, если открыть устройство не удалось.

4.3.2 Метод close

Объявление:

```
public void close()
```

Описание:

«Закрывает» контрольный считыватель.

Параметры:

Нет

Возвращаемое значение:

Нет.

4.3.3 Метод receiveW26

Объявление:

```
public boolean receiveW26(byte buffer[])
```

Описание:

Пытается получить следующий Wiegand-26 код от контрольного считывателя.

Функция блокирует вызвавший ее поток до момента получения кода или наступления ошибки. В случае если функция возвращает false, это может означать одно из следующих событий:

- Переданный буфер слишком мал (см. ниже)
- Ошибка информационного обмена с устройством. Это, вероятно, означает, что USB устройство было отключено или его работа была приостановлена. Устройство в таком случае следует закрыть функцией close, после чего можно повторно попытаться открыть его через некоторое время.

Параметры:

Параметр	Использование
buffer	Буфер, куда будет записан полученный код. Длина буфера должна составлять минимум 3 байта.

Возвращаемое значение:

Возвращает true в случае успеха. В таком случае в buffer был помещен полученный код.

Возвращает false, если получить код не удалось.

4.3.4 Метод receiveW26T

Объявление:

```
public int receiveW26T(byte buffer[],int timeoutMs)
```

Описание:

Пытается получить следующий Wiegand-26 код от контрольного считывателя.

Функция блокирует вызвавший ее поток до наступления одного из следующих событий:

- Получен код.
- Наступила ошибка работы с устройством. Это, означает одно из следующих вариантов:
 - USB устройство было отключено или его работа была приостановлена. Устройство в таком случае следует закрыть функцией SpnxReaderClose, после чего можно повторно попытаться его открыть через некоторое время.
 - Переданный буфер слишком мал (см. ниже)
- Истек таймаут ожидания кода, но код так и не был получен.

Параметры:

Параметр	Использование
buffer	Буфер, куда будет записан полученный код. Длина буфера должна составлять минимум 3 байта.
timeoutMs	Таймаут ожидания получения кода.

Возвращаемое значение:

0, если истек таймаут ожидания, но код так и не был получен.

-1, если наступила ошибка (см. выше)

Кол-во полученных байт кода, если код получен.

5. Разработка на других языках

Вы также можете разрабатывать проект под платформу win32 на других языках программирования с использованием любых сред разработки (например, «Delphi»).

Для этого вам потребуется задействовать DLL-библиотеку «sphinxreader.dll» и вызывать ее функции из своей программы. Описание функций можно найти в разделе данного документа о использовании библиотеки из C/C++.

6. Получение технической поддержки

Для получения консультаций по использованию пакета разработчика обращайтесь в компанию «ПромАвтоматика».

Телефон/факс: +7 (831) 464-97-16

Телефон: +7 (831) 415-50-67

Электронная почта: info@spx.ru

Актуальная документация на все оборудование и программное обеспечение, а также драйверы устройств могут быть получены на сайте: www.spx.ru

ООО «ПромАвтоматика»

603057, Нижний Новгород, пр. Гагарина, д. 50 корп. 14 офис 411

Телефон/факс: +7 (831) 464-97-16

Техническая поддержка: +7 (831) 415-50-67

Система контроля и управления доступом «Сфинкс»

Веб: <http://www.spnx.ru>

Электронная почта: info@spnx.ru