Программное обеспечение «ПАПИЛОН-LSSDK»
Руководство программиста

Содержание

Состав программного обеспечения

Программное обеспечение (ПО) состоит из следующих компонентов:

- Динамически подключаемая библиотека (*.dll или *.so);
- заголовочный файл lsclient.h;
- пример использования библиотеки – бинарная реализация клиентского приложения lsclient_test;
- пакеты с драйверами для сканеров

Установка драйвера электронного дактилоскопического сканера

Linux

Для установки драйвера электронного дактилоскопического сканера необходимот предварительно установить на компьютере пакет DKMS, если он не был установлен ранее.

Также на компьютере предварительно должны быть установлены заголовочные файлы ядра.

Драйвер устройства устанавливается командой вида:

```
rpm -i <…>dkms.noarch.rpm
```

В результате этих действий DKMS скомпилирует драйвер и установит его в каталог /lib/modules/`uname -r`/kernel/drivers/misc.

Файл загрузки драйвера и создания специальных файлов в каталоге /dev/ds22_usb будет установлен в каталоге /etc/rc.d/init.d/ и включен в 3,4,5 уровни исполнения (runlevel).

При загрузке компьютеры сервис ds22_usb загрузит драйвер сканера и создаст необходимые файлы в каталоге /dev/usb.

Windows

После подключения сканера к компьютеру ОС запросит драйвера для сканера - необходимо указать каталог, в котором находятся драйвера для соответствующей модели сканера, например Drivers/DS22/. Если сканер поставляется со встроенным LCD-экраном, то после установки драйверов для сканера будет выдан запрос на драйвера для LCD-экрана, необходимо указать каталог Drivers/LCD.

Поддерживаемые сканеры

Библиотекой поддерживаются следующие сканеры, выпускаемые фирмой "Папилон":

- Сканер для получения оттисков отпечатков DS21/DS22
- Сканер для получения оттисков отпечатков DS20/FX2000
- Сканер для получения прокатанных отпечатков и контрольных оттисков DS9.
- Сканер для получения прокатанных отпечатков и контрольных оттисков DS30.
- Сканер для получения прокатанных отпечатков и контрольных оттисков DS30M.
- Сканер для получения прокатанных отпечатков и контрольных оттисков DS31.
- Сканер для получения прокатанных отпечатков, контрольных оттисков и изображений ладоней DS7.
- Сканер для получения прокатанных отпечатков, контрольных оттисков и изображений ладоней DS10.
- Сканер для получения прокатанных отпечатков, контрольных оттисков и изображений ладоней DS14.
- Сканер для получения прокатанных отпечатков, контрольных оттисков и изображений ладоней DS40.
- Сканер для получения прокатанных отпечатков, контрольных оттисков и изображений ладоней DS40M.
- Сканер для получения прокатанных отпечатков, контрольных оттисков и изображений ладоней DS45.
- Сканер для получения прокатанных отпечатков, контрольных оттисков и изображений ладоней DS45M.
- Сканер для получения прокатанных отпечатков, контрольных оттисков DS30N.
- Сканер для получения прокатанных отпечатков, контрольных оттисков DS30NM.
- Сканер для получения прокатанных отпечатков DS22N.
- Сканер для получения прокатанных отпечатков DS21S (Orion) с возможностью определения муляжа пальца.

Порядок установки драйверов описан в разделе "Драйвера".

Порядок работы с библиотекой

Приложение, использующее данную библиотеку, должно:

1. Создать объект LSClient
2. Инициализировать следующие поля:
    - LSClient::Client
    - LSClient::debug_level
    - LSClient::GetCommand
    - LSClient::DrawPreview
    - LSClient::ClearPreview
    - LSClient::TakeImage
    - LSClient::GetWorkDir
    - LSClient::ProcessButton
    - LSClient::InitDone
    - LSClient::StateChanged
    - LSClient::DrawLCD
3. Вызывать функцию LSCInit().

4. Запустить поток прокатки функцией LSCStart().
5. Дождаться окончания инициализации по смене статуса с LSC_STATE::LSC_STAT_INIT на другой, либо после вызова callback-функции LSClient::InitDone().
6. Начать работу с потоком прокатки отдавая команды. через callback-функцию LSClient::GetCommand().
7. Для включения алгоритма слежения за отпечатком и прокатки необходимо подать команду LSC_COMMAND::LSC_CMD_ROLL.
8. После завершения работы с потоком необходимо либо остановить его командой LSC_COMMAND::LSC_CMD_STOP (если планируется дальнейшее использование сканера), либо завершить функцией LSCShutdown().
9. После остановки потока командой LSC_COMMAND::LSC_CMD_STOP можно возобновить его работу, подав команду LSC_COMMAND::LSC_CMD_ROLL.
10. После завершения функцией LSCShutdown() возобновить работу данного потока невозможно, в этом случае необходимо заново инициализировать объект LSClient и сам поток прокатки.

Группы объектов программы

Полный список групп.

- Команды для потока прокатки
- Коды кнопок клавиатуры сканера
- Коды режимов управления светодиодами индикации сканера
- Номера диодов индикации сканера
- Ошибки процесса прокатки, допущенные оператором
- Типы сканеров
- Возвращаемые функциями ошибки
- Состояние потока прокатки
- Зоны прокатки
- Структура LSClient
- Callback-функции клиентского приложения
- Функции библиотеки
- Функции сжатия/разжатия WSQ

Команды для потока прокатки

Перечисления

enum LSC_COMMAND

Элементы перечислений:

- LSC_CMD_NOP - пустая команда
- LSC_CMD_EXIT - завершить поток прокатки
- LSC_CMD_STOP - остановить режим ожидания отпечатка и прокатки
- LSC_CMD_ROLL - запустить режим ожидания отпечатка и прокатки

- LSC_CMD_CLEAR - компенсировать грязь на призме
- LSC_CMD_TOUCH - захватить оттиск прямо сейчас
- LSC_CMD_START - не использовать
- LSC_CMD_STOP_ROLL - не использовать
- LSC_CMD_FINGERS - команды переключения текущего отпечатка. Для пальцев 0-9 -прокатка, для остальных - оттиск
- LSC_CMD_FINGER1 - Правый большой
- LSC_CMD_FINGER2 - Правый указательный
- LSC_CMD_FINGER3 - Правый средний
- LSC_CMD_FINGER4 - Правый безымянный
- LSC_CMD_FINGER5 - Правый мизинец
- LSC_CMD_FINGER6 - Левый большой
- LSC_CMD_FINGER7 - Левый указательный
- LSC_CMD_FINGER8 - Левый средний
- LSC_CMD_FINGER9 - Левый безымянный
- LSC_CMD_FINGER10 - Левый мизинец
- LSC_CMD_FINGER11 - Левые 4 оттиска
- LSC_CMD_FINGER12 - Левый большой оттиск
- LSC_CMD_FINGER13 - Правый большой оттиск
- LSC_CMD_FINGER14 - Правые 4 оттиска
- LSC_CMD_FINGER15 - Левая ладонь
- LSC_CMD_FINGER16 - Правая ладонь
- LSC_CMD_FINGER17 - Левая ладонь, ребро
- LSC_CMD_FINGER18 - Правая ладонь, ребро

Коды кнопок клавиатуры сканера

Перечисления

enum LSC_BUTTONS

Элементы перечислений:

- LSC_BTN_NONE - ничего не нажато
- LSC_BTN_LEFT - нажата стрелка влево
- LSC_BTN_RIGHT - нажата стрелка вправо
- LSC_BTN_OK - нажата кнопка Ок
- LSC_BTN_CANCEL - нажата кнопка Cancel

Коды режимов управления светодиодами индикации сканера

Перечисления

enum LSC_LED_MODE

Режим индикации изменяется функцией LSC_SetLedMode().

Элементы перечислений:

- LSC_LED_AUTO - автоматический режим индикации
- LSC_LED_MANUAL - ручной режим, индикация управляется функцией LSC_SetLed()

Номера диодов индикации сканера

Перечисления

enum LSC_LED

Элементы перечислений:

- LSC_GREEN_LED - зеленый диод
- LSC_RED_LED - красный диод

Типы сканеров

Перечисления

enum LSC_DEVICE

Элементы перечислений:

- UNKNOWN - Неизвестный сканер
- DS7 -Ладонный сканер ДС7
- DS9 - Пальцевый сканер ДС9 IEEE1394 или PCI
- DS21 - Пальцевый сканер ДС22 или ДС21
- DS30 - Пальцевый сканер с контрольными оттисками ДС30
- DS31 - Пальцевый сканер с контрольными оттисками ДС31(2 призмы)
- DS14 - Ладонный сканер ДС14-USB
- DS40 - DS40 palm scanner
- DS16 - Palm scanner DS16 + DS22 for rolling
- DS24 - Сканер отпечатков DS24, 1000дпи
- FDS7 - DS7 IEEE-1394.
- DS45 - Сканер DS45, DS45M.
- DS22N - Сканер DS22N.
- DS30N - Сканер DS30N.
- DS30NM - Сканер DS30NM.
- DS21N - Сканер DS21N.
- CM_1000P - Сканер DS21S(Orion).

Возвращаемые функциями ошибки

Перечисления

enum LSC_ ERROR

Элементы перечислений:

- LSC_CONNECT_ERROR - No errors
- LSC_MEMORY_ERROR - Ошибка соединения со сканером
- LSC_RESOURCES_ERROR - Мало памяти
- LSC_READ_CONFIG_ERROR - Ошибка выделения ресурсов IPC
- LSC_CALIBRATE_ERROR - Ошибка чтения конфигурации сканера
- LSC_ROLL_ERROR - Сканер не откалиброван
- LSC_EXIT_ERROR - Ошибка прокатки
- LSC_SLAP_IMAGE_ERROR - Поток прокатки закрыт Ошибка разделения контрольных оттисков
- LSC_SLAP_FINGER_NOT_FOUND - Ошибка разделения контрольных оттисков, не все отпечатки обнаружены
- LSC_ARG_ERROR - Неверные аргументы функции
- LSC_LICENSE_ERROR - Неверная лицензия

Состояние потока прокатки

Перечисления

enum LSC_ STATE

Элементы перечислений:

- LSC_STAT_INIT - инициализация
- LSC_STAT_WAIT - ожидание команды без слежения за отпечатком и прокатки
- LSC_STAT_WAIT_FINGER - слежение за отпечатком
- LSC_STAT_ROLL - собственно процесс прокатки
- LSC_STAT_GLUE - обработка полученного после прокатки изображения
- LSC_STAT_CLEARING - компенсация грязи на призме
- LSC_STAT_EXITED - поток завершился
- LSC_STAT_ERROR - произошла какая-либо ошибка

Зоны прокатки

Перечисления

enum LSC_ ROLL_PLACE

Определение зоны прокатки.

Элементы перечислений:

- LSC_RPLACE_AUTO - Автоматический выбор зоны прокатки.
- Правая рука прокатывается справа, левая рука прокатывается слева.
- LSC_RPLACE_LEFT - Прокатка выполняется в левой части призмы.
- LSC_RPLACE_RIGHT - Прокатка выполняется в правой части призмы.

Callback-функции клиентского приложения

Переменные

LSC_COMMAND(* LSClient::GetCommand)(LSC_HNDL Cl) [inherited]

Получение потоком прокатки команды от программы верхнего уровня. Функция вызывается внутри потока прокатки для получения команды от клиентского приложения.

Аргументы:

Cl - указатель на объект клиентского приложения.

Возвращает: команду для потока прокатки.

int(* LSClient::DrawPreview)(LSC_HNDL Cl, int finger_num, unsigned char *Buf, int W, int H) [inherited]

Вывод preview на экран. Когда отпечаток обнаружен на призме прибора, подготавливается изображение для показа оператору на экране. Эта функция вызывается из потока прокатки для передачи клиентскому приложению подготовленного preview-изображения.

Аргументы:

Cl	- указатель на объект клиентского приложения.
finger_num	- номер текущего отпечатка, установленного одной из команд LSC_CMD_FINGERS.
Buf	- указатель на буфер с изображением, буфер глубиной 8 bpp (256 градаций серого), первый байт - левый верхний пиксель изображения.
W	- ширина изображения (количество колонок).
H	- высота изображения (количество строк).

Возвращает: 0 - в случае успеха.

int(* LSClient::ClearPreview)(LSC_HNDL Cl) [inherited]

Очистка окна preview. Вызывается, когда отпечаток убран с призмы либо процесс прокатки завершен.

Аргументы:

Cl - указатель на объект клиентского приложения.

Возвращает: 0 - в случае успеха.

int(* LSClient::TakeImage)(LSC_HNDL Cl, int finger_num, unsigned char *Buf, int W, int H, unsigned int roll_errors) `[inherited]`

Передача готового изображения отпечатка из потока прокатки в клиентское приложение. Вызывается когда процесс прокатки завершен.

Аргументы:

Cl	- указатель на объект клиентского приложения.
finger_num	- номер отпечатка
Buf	- указатель на буфер с изображением отпечатка, освобождается в потоке прокатки. Глубина буфера 8 bpp (256 градаций серого), разрешение 500 ppi, первый байт - верхний левый пиксел изображения. Может быть NULL, в этом случае будет установлена переменная roll_errors в соответствующее значение ошибки прокатки.
W	- ширина изображения(количество колонок).
H	- высота изображения(количество строк).
roll_errors	- маска ошибок прокатки. В случае критических ошибок параметр Buf возвращается как NULL.

Возвращает: 0 - в случае успеха.

char*(* LSClient::GetWorkDir)(LSC_HNDL Cl) `[inherited]`

Получить путь к рабочему каталогу. Вызывается из потока прокатки для получения каталога в котором можно сохранять временные файлы.

Аргументы:

Cl - указатель на объект клиентского приложения.

Возвращает: path - путь к каталогу.

int(* LSClient::ProcessButton)(LSC_HNDL Cl, LSC_BUTTONS button) `[inherited]`

Передача кода кнопки, нажатой на сканере. Вызывается из потока прокатки для передачи информации о нажатии какой-либо кнопки на клавиатуре сканера.

Аргументы:

Cl - указатель на объект клиентского приложения.
button - нажатая кнопка.

Возвращает: 0 в случае успеха.

void(* LSClient::InitDone)(LSC_HNDL Cl) [inherited]

Уведомление клиентского приложения о завершении инициализации потока прокатки. Вызывается из потока прокатки после завершения процедуры тестирования и инициализации сканера. После этого можно начинать работу с потоком прокатки - инициализация preview LSCSetPreview(), LSCGetPreviewSlap(), передавать команды в поток через callback LSClient::GetCommand().

Аргументы:

Cl - указатель на объект клиентского приложения.

void(* LSClient::StateChanged)(LSC_HNDL Cl, LSC_STATE state) [inherited]

Уведомление клиентского приложения о изменении состояния потока прокатки. Вызывается из потока прокатки после изменения состояния потока прокатки.

Аргументы:

Cl - указатель на объект клиентского приложения.
state - новое состояние потока прокатки.

void(* LSClient::DrawLCD)(LSC_HNDL Cl) [inherited]

Уведомление клиентского приложения о возможности отправить картинку на USB LCD. Вызывается из потока прокатки во время перерыва передачи данных со сканера по USB-шине. Используется только со сканерами, оборудованными LCD монитором(DS30M и DS15M). Если DrawLCD не инициализирован(равен NULL), то поток прокатки не будет вызывать этот callback.

Аргументы:

Cl - указатель на объект клиентского приложения.

void(* LSClient::RemoveFinger)(LSC_HNDL Cl) [inherited]

Уведомление клиентского приложения о том что накоплено достаточное количество кадров для построения финального изображения. Эта функция вызывается из потока прокатки, когда накоплено достаточное количество кадров для построения финального изображения, в случае прокатки отпечатка оператор должен сам решить, когда прокатка выполнена полностью. Если RemoveFinger не инициализирован (равен NULL), то поток прокатки не будет вызывать этот callback.

Аргументы:

Cl - указатель на объект клиентского приложения.

void(* LSClient::ScanComplete)(LSC_HNDL Cl) [inherited]

Сканирование всех отпечатков закончено. Вызывается из потока сканирования, когда отсканированы все отпечатки, используется только для сканера CorrMatch VMW.

void(* LSClient::AddScanStat)(LSC_HNDL Cl, char *data, int tag, int multiple) [inherited]

Передает в клиентское приложение статистику сканирования. Эти данные могут быть сохранены в отдельные текстовые записи для дальнейшего анализа ошибок прокатки оператора, ошибок работы сканера. Callback вызывается после LSC_CMD_START и LSC_CMD_ROLL команд для сохранения информации о модели, настройке сканера и версиях программного обеспечения.

TakeImage callback для сохранения информации о параметрах сканирования.

Аргументы:

Cl - указатель на объект клиентского приложения.
data - текстовая строка, заканчивающаяся нулем
tag - номер текстового тега записи
multiple - флаг множественного тега:

- 1 - множественный, данные должны быть добавлены к записям с таким же номером тега
- 0 - единичный, данные должны заместить записи с таким же номером тега

Функции для работы с устройством

Функции

```
LSCLIENT_API LSC_ERROR LSCInit   ( LSClient *       cl,
                                   LSC_DEVICE     device,
                                   int            num
                                 )
```

Инициализация библиотеки прокатки. Функция захватывает необходимые для работы ресурсы и создает соединение со сканером.

Аргументы:

cl - указатель на структуру LSClient.
device - тип сканера, с которым будет работать данный поток.
num - номер сканера, начиная с нуля.

Возвращает: код ошибки.

```
LSCLIENT_API LSC_ERROR LSCSetPreview   ( LSClient *       cl,
                                         int *            width,
                                         int *            height
                                       )
```

Установка размеров preview-изображения при прокатке. Функция вычисляет параметры preview изображения и возвращает размеры preview в аргументах width и height. В дальнейшем именно с этими размерами будет вызываться callback-функция LSClient::DrawPreview(). Данную функцию можно вызывать только после окончания процесса инициализации. Окончание инициализации сигнализируется сменой статуса потока прокатки с LSC_STAT_INIT на любой другой и вызовом callback-функции LSClient::InitDone().

Аргументы:

cl - Указатель на структуру LSClient.
width - ширина изображения
height - высота изображения

Возвращает: код ошибки.

```
LSCLIENT_API LSC_ERROR LSCGetPreviewSlap   ( LSClient *       cl,
                                             int *            width,
                                             int *            height
                                           )
```

Установка размеров preview-изображения при снятии оттисков. Функция вычисляет параметры preview изображения при снятии контрольных оттисков и возвращает размеры preview в аргументах width и height. В дальнейшем именно с этими размерами будет вызываться callback-функция LSClient::DrawPreview(). Данную функцию можно вызывать только после окончания процесса инициализации. Окончание инициализации сигнализируется сменой статуса потока прокатки с LSC_STAT_INIT на любой другой и вызовом callback-функции LSClient::InitDone().

Аргументы:

cl - Указатель на структуру LSClient.
width - ширина изображения
height - высота изображения

Возвращает: код ошибки.

```
LSCLIENT_API LSC_ERROR LSCGetSlapSize   ( LSClient *    cl,
                                          int *         width,
                                          int *         height
                                        )
```

Не используется.

Аргументы:

cl - Указатель на структуру LSClient.
width - Указатель на ширину изображения.
height - Указатель на высоту изображения.

Возвращает: код ошибки.

```
LSCLIENT_API LSC_ERROR LSCGetFingerSize   ( LSClient *    cl,
                                            int *         width,
                                            int *         height
                                          )
```

Не используется.

Аргументы:

cl - Указатель на структуру LSClient.
width - Указатель на ширину изображения.
height - Указатель на высоту изображения.

Возвращает: код ошибки.

LSCLIENT_API LSC_ERROR LSCStart (LSClient * cl)

Запуск потока прокатки. Функция порождает дополнительный поток, который выполняет процесс прокатки. Все callback-функции вызываются из этого потока. Поэтому любое обращение к графическому интерфейсу пользователя из них нежелательно. Лучше использовать систему сообщений между callback-функциями и главным потоком клиентского приложения.

Аргументы:

cl - Указатель на структуру LSClient.

Возвращает: код ошибки.

LSCLIENT_ API void LSCShutDown (LSClient * cl)

Завершение потока прокатки и освобождение всех ресурсов. Функция передает в поток прокатки команду на завершение и ожидает окончания этого потока, затем освобождает все занятые ресурсы и закрывает соединение со сканером.

Аргументы:

cl - Указатель на структуру LSClient.

Примеры: mainform.cpp

LSCLIENT_API LSC_STATE LSCGetState (LSClient * cl)

Получение текущего состояния потока прокатки. Функция возвращает статус потока прокатки. При изменении статуса так же вызывается callback-функция LSClient::StateChanged().

Аргументы:

cl - Указатель на структуру LSClient.

Возвращает: Состояние потока прокатки.

LSCLIENT_API LSC_ERROR LSCGetError (LSClient * cl)

Получение кода ошибки потока прокатки. Функция возвращает код ошибки потока прокатки. В случае какой-либо ошибки внутри потока статус потока меняется на LSC_STATE::LSC_STAT_ERROR.

Аргументы:

cl - Указатель на структуру LSClient.

Возвращает: код ошибки.

Примеры: mainform.cpp

```
LSCLIENT_API char* LSCGetVersion      ( LSClient *  cl )
```

Получить версию библиотеки. Функция позволяет получить версию библиотеки в виде указателя на строку.

Аргументы:

cl - Указатель на структуру LSClient.

Возвращает: Строку с версией библиотеки.

```
LSCLIENT_API void LSCSetFingerArea      ( LSClient *      cl,
                                          int             left,
                                          int             right
                                        )
```

Не используется.

Аргументы:

cl - Указатель на структуру LSClient.
left - левая граница изображения.
right - правая граница изображения.

```
LSCLIENT_API LSC_ERROR LSCslapSegm      ( LSClient *        cl,
                                          unsigned char *   image,
                                          int               width,
                                          int               height,
                                          int               num,
                                          int *             left,
                                          int *             top,
                                          int *             fwidth,
                                          int *             fheight
                                        )
```

Сегментация контрольных оттисков. Находит отпечаток с заданным номером в изображении контрольных оттисков и возвращает его координаты.

Аргументы:

cl - Указатель на структуру LSClient
image - изображение контрольных оттисков
width - ширина изображения
height - высота изображения
num - номер отпечатка для выделения.
left - левая граница отпечатка в изображении
top - верхняя граница отпечатка в изображении
fwidth - ширина отпечатка
fheight - высота отпечатка

Возвращает: коды ошибок

```
LSCLIENT_API LSC_ERROR LSCslapSegmAll    ( LSClient *          cl,
                                           unsigned char *     image,
                                           int                 width,
                                           int                 height,
                                           int *               num,
                                           int *               left,
                                           int *               top,
                                           int *               fwidth,
                                           int *               fheight
                                         )
```

Сегментация контрольных оттисков. Разделяет контрольные оттиски на отдельные отпечатки и возвращает координаты и размеры найденных отпечатков.

Аргументы:

cl - указатель на структуру LSClient
image - изображение оттисков
width - ширина изображения
height - высота изображения
num - количество выделенных отпечатков
left - массив (int[4]) для сохранения левых границ отпечатков
top - массив (int[4]) для сохранения верхних границ отпечатков
fwidth - массив (int[4]) для сохранения ширины отпечатков
fheight - массив (int[4]) для сохранения высоты отпечатков

Возвращает: коды ошибок

Примеры: mainform.cpp.

```
LSCLIENT_API int LSC_GetScannersList      ( LSC_SCANNERS_LIST **   list  )
```

Формирует список подключенных сканеров.

Аргументы:

list - Указатель на указатель для списка сканеров.

Возвращает: длину выделенного списка подключенных сканеров

Примеры: mainform.cpp

```
LSCLIENT_API void LSC_SetLedMode    ( LSClient *      cl,
                                      LSC_LED_MODE    mode
                                    )
```

Установить режим индикации светодиодов сканера.

Аргументы:

cl - указатель на структуру клиентского приложения
mode - режим управления индикацией

Примеры: mainform.cpp

```
LSCLIENT_API LSC_ERROR LSC_SetLed   ( LSClient *      cl,
                                      LSC_LED         led,
                                      BOOL            on
                                    )
```

Выключение/выключение светодиодов индикации сканера в ручном режиме управления.

Аргументы:

cl - указатель на структуру клиентского приложения
led - номер светодиода, красный или зеленый.
on - состояние диода, TRUE - включить, FALSE - выключить.

Возвращает: коды ошибок

Примеры: mainform.cpp.

```
LSCLIENT_API LSC_ERROR LSC_SetUsbType   ( LSClient *      cl,
                                          BOOL            slow
                                        )
```

Переключение режима передачи данных сканером по USB-шине. Функция может применяться для уменьшения потока данных от сканера к компьютеру по USB-шине в случае медленной USB-шины.

Аргументы:

cl - указатель на структуру клиентского приложения
slow - TRUE - медленная USB-шина, FALSE - полноценная USB2 HIGH-SPEED 480mb/s шина.

Возвращает: коды ошибок

Примеры: mainform.cpp.

LSCLIENT_API void LSCSetCaptureMode	**(LSClient * LSC_CAPTURE_MODE)**	**cl, mode**

Установить режим захвата оттисков.

Аргументы:

cl - указатель на структуру клиентского приложения
mode - режим захвата оттисков

Примеры: mainform.cpp.

LSCLIENT_API LSC_ERROR LSCSetThreshold	**(LSClient * int)**	**cl, threshold**

Установить чувствительность алгоритма определения наличия отпечатков на призме. Чем больше число, те менее чувствителен алгоритм к отпечаткам и грязи соответственно. Рекомендуемое значение(оно же и по умолчанию) - 500.

Аргументы:

cl - указатель на структуру клиентского приложения
threshold - чувствительность, от LSC_MIN_THRESHOLD до LSC_MAX_THRESHOLD

LSCLIENT_API void LSCSetTouchMode	**(LSClient * BOOL)**	**cl, touch_mode**

Переключение режимов оттисков/прокатки для отпечатков.

Аргументы:

cl - указатель на структуру клиентского приложения
touch_mode - режим

- TRUE - берутся оттиски пальцев
- FALSE - пальца прокатываются

Примеры: mainform.cpp.

```
LSCLIENT_API LSC_ERROR LSC_SendJpeg2Device ( LSClient *          cl,
                                             unsigned char *     buffer,
                                             int                 length
                                           )
```

Отправить JPEG-изображение на монитор сканера. Данная функция работает только со сканером CrossMatch VMW.

Аргументы:

cl - указатель на структуру клиентского приложения
buffer - указатель на jpeg-изображения
length - длина буфера изображения

Возвращает: коды ошибок

Примеры: mainform.cpp.

```
LSCLIENT_API LSC_ERROR LSC_GetSizeLCD ( LSClient *    cl,
                                        int *         width,
                                        int *         height
                                      )
```

Получить размеры экрана сканера. Работает только с интегрированным LCD-экраном сканера DS30NM, с остальными сканерами для вывода на экран необходимо использовать библиотеку liblcd.

Аргументы:

cl - указатель на структуру клиентского приложения
width - возвращается ширина экрана
height - возвращается высота экрана

Возвращает: возвращаемые функциями ошибки.

```
LSCLIENT_API LSC_ERROR LSC_ResetDataLCD ( LSClient *  cl, )
```

Сброс USB-буферов для передачи на LCD экран сканера. Работает только с интегрированным LCD-экраном сканера DS30NM, с остальными сканерами для вывода на экран необходимо использовать библиотеку liblcd.

Аргументы:

cl - указатель на структуру клиентского приложения

Возвращает: возвращаемые функциями ошибки.

```
LSCLIENT_API LSC_ERROR LSC_WriteImageLCD   ( LSClient *    cl,
                                             long          size,
                                             void *        image
                                           )
```

Передать изображение на LCD экран сканера. Работает только с интегрированным LCD-экраном сканера DS30NM, с остальными сканерами для вывода на экран необходимо использовать библиотеку liblcd.

Аргументы:

cl - указатель на структуру клиентского приложения
size - размер буфера с изображением
image - буфер с изображением

Возвращает: возвращаемые функциями ошибки.

```
LSCLIENT_API LSC_ERROR LSC_WriteMaskLCD   ( LSClient *    cl,
                                            long          AMaskSize,
                                            void *        AMaskBuff
                                          )
```

Передать палитру на LCD экран сканера. Работает только с интегрированным LCD-экраном сканера DS30NM, с остальными сканерами для вывода на экран необходимо использовать библиотеку liblcd.

Аргументы:

cl - указатель на структуру клиентского приложения
AMaskSize - размер палитры
AMaskBuff - буфер с палитрой, по 3 байта на цвет, 256 цветов.

Возвращает: возвращаемые функциями ошибки.

```
LSCLIENT_API LSC_ERROR LSC_SetLightLCD   ( LSClient *     cl,
                                           int            light
                                         )
```

Установить яркость подсветки LCD экрана сканера. Работает только с интегрированным LCD-экраном сканера DS30NM, с остальными сканерами для вывода на экран необходимо использовать библиотеку liblcd.

Аргументы:

cl - указатель на структуру клиентского приложения
light - яркость подсветки, от 0 до 31.

Возвращает: возвращаемые функциями ошибки.

```
LSCLIENT_API LSC_ERROR LSC_SetLCDPower   ( LSClient *     cl,
                                           int            on
                                         )
```

Включение/выключение питания экрана сканера. Работает только с интегрированным LCD-экраном сканера DS30NM, с остальными сканерами для вывода на экран необходимо использовать библиотеку liblcd.

Аргументы:

cl - указатель на структуру клиентского приложения
on - 1 - включить питание, 0 - выключить.

Возвращает: возвращаемые функциями ошибки.

```
LSCLIENT_API BOOL LSC_IntegratedLCD      ( LSClient *     cl, )
```

Возвращает флаг наличия в сканере интегрированного LCD экрана.

Аргументы:

cl - указатель на структуру клиентского приложения

Возвращает:

- TRUE - экран интегрирован
- FALSE - экран не интегрирован

```
LSCLIENT_API LSC_ERROR LSC_GetFirmware   ( LSClient *    cl,
                                           char *        firmware
                                         )
```

Возвращается версию EEPROM firmware сканера.

Аргументы:

cl - указатель на структуру клиентского приложения
firmware - возвращается версия firmware

Возвращает: возвращаемые функциями ошибки.

```
LSCLIENT_API LSC_ERROR LSC_GetAfisQuality   ( LSClient *          cl,
                                              unsigned char *     image,
                                              int                 width,
                                              int                 height,
                                              int                 res,
                                              int *               aquality,
                                              int *               vquality,
                                              float *             pressure
                                            )
```

Расчет качества отпечатка по алгоритму Папилон.

Аргументы:

[in]	*cl*	- object of library.
[in]	*image*	- буфер с изображением. Изображение должно иметь глубину 8 bpp.
[in]	*width*	- количество колонок
[in]	*height*	- количество строк
[in]	*res*	- разрешение изображения в пикселах на дюйм, обычно 500.
[in]	*cl*	- указатель на структуру клиентского приложения.
[out]	*aquality*	- указатель для сохранения качества папиллярного узора. Изменяется от 0 (плохое) до 100 (отличное).
[out]	*vquality*	- указатель для сохранения качества изображения изменяется от 0 (худшее) до 100 (отличное). Характеризует динамический диапазон изображения.
[out]	*pressure*	- указатель для сохранения силы прижима пальца к призме. Изменяется 0.0 (слабое давление) до 2.0 (сильное давление). Нормальное значение 1.0.

Возвращает: возвращаемые функциями ошибки.

Примеры: mainform.cpp.

LSCLIENT_API LSC_ERROR LSC_GetNistQuality	**(LSClient ***	**cl,**
	unsigned char *	**image,**
	int	**width,**
	int	**height,**
	int	**res,**
	int *	**aquality,**
	int *	**vquality,**
	float *	**pressure**
	)	

Расчет качества отпечатка по алгоритму NIST.

Аргументы:

[in]	*cl*	- object of library.
[in]	*image*	- буфер с изображением. Изображение должно иметь глубину 8 bpp.
[in]	*width*	- количество колонок
[in]	*height*	- количество строк
[in]	*res*	- разрешение изображения в паикселах на дюйм, обычно 500.
[in]	*cl*	- указатель на структуру клиентского приложения.
[out]	*aquality*	- указатель для сохранения качества папиллярного узора. Изменяется от 5 (плохое) до 1 (отличное).
[out]	*vquality*	- указатель для сохранения качества изображения изменяется от 0(худшее) до 100(отличное). Характеризует динамический диапазон изображения.
[out]	*pressure*	- указатель для сохранения силы прижима пальца к призме. Изменяется 0.0 (слабое давление) до 2.0 (сильное давление). Нормальное значение 1.0.

Возвращает: возвращаемые функциями ошибки.

Примеры: mainform.cpp.

LSCLIENT_API BOOL LSC_ SupportRollPlace	**(LSClient ***	**cl,)**

Проверить поддерживает ли сканер разные зоны прокатки.

Аргументы:

cl - указатель на структуру клиентского приложения.

Возвращает: true, если сканер поддерживает выбор различных зон прокатки.

```
LSCLIENT_API LSC_ERROR LSC_ SetRollPlace    ( LSClient *          cl,
                                              LSC_ROLL_PLACE      place
                                            )
```

Выбор зоны прокатки. Функцию можно использовать только для сканеров, которые поддерживают выбор зоны прокатки, в данный момент только для сканера ДС45 (М).

Аргументы:

cl - указатель на структуру клиентского приложения.
place - зона прокатки

Возвращает: возвращаемые функциями ошибки.

```
LSCLIENT_API BOOL LSC_EnableFakeDetection    ( LSClient *    cl,
                                               bool          _enable
                                             )
```

Включение/выключение режима определения муляжа отпечатка. При включении этой функции время захвата отпечатка увеличивается примерно в 2 раза. Данный режим работает только на некоторых сканерах.

Аргументы:

cl - указатель на структуру клиентского приложения.
_enable - true - включить определение муляжа, false - выключить.

Возвращает: false, если данная функция не поддерживается сканером.

```
LSCLIENT_API LSC_ERROR LSC_ CheckEncode    ( LSClient *          cl,
                                             unsigned char *     _image,
                                             int                 _width,
                                             int                 _height,
                                             BOOL *              _good
                                           )
```

Функция проверяет возможность извлечения мелких особенностей из изображения отпечатка. Эта функция анализирует качество изображения на предмет пригодности для кодирования и возвращает результат в параметре _good. Если изображение достаточного качества для последующего кодирования и сравнения, то выходной параметр _good будет содержать TRUE. Функция достаточно ресурсоемкая и занимает времени порядка 1 секунды на процессоре Intel Core i7 3ГГц для одного отпечатка. Использование данной функции защищено коммерческой лицензией, за получением лицензии обратитесь к разработчику.

Аргументы:

	cl	- указатель на структуру клиентского приложения.
	_image	- указатель на изображение отпечатка, 8bpp, 500ppi
	_width	- ширина изображения в пикселах, количество колонок
	_height	- высота изображения в пикселах, количество строк
[out]	**_good**	- на выходе содержит TRUE если изображение пригодно для обработки

Возвращает: возвращаемые функциями ошибки.

Примеры: mainform.cpp.

LSCLIENT_API LSC_ERROR LSC_ CheckQuality	**(LSClient ***	**cl,**
	unsigned char *	**_image,**
	int	**_width,**
	int	**_height,**
	int *	**_ quality**
	)	

Вычисление качества отпечатка применительно к алгоритму поиска компании Папилон. Эта функция рассчитывает качество отпечатка 0 (плохое) ... 3 (отличное). Функция может быть использована для оценки необходимого количества отпечатков для выполнения автоматического поиска по базе. Наилучшее качество поиска будет достигнуто при минимум двух отпечатках с качеством не ниже 2. Возможно подавать на поиски один отпечаток с качеством 3. В остальных случаях необходимо сканировать дополнительные отпечатки. Функция защищена коммерческой лицензией.

Аргументы:

	cl	- указатель на структуру клиентского приложения.
	_image	- указатель на изображение отпечатка, 8bpp, 500ppi
	_width	- ширина изображения в пикселах, количество колонок
	_height	- высота изображения в пикселах, количество строк
[out]	*_quality*	- выходное значение качества отпечатка.

Возвращает: возвращаемые функциями ошибки.

Примеры: mainform.cpp.

LSCLIENT_API LSC_ERROR LSC_SetLicense	**(LSClient ***	**cl,**
	const char *	**_path**
	)	

Установить файл лицензии LSSDK. Лицензия может быть привязана к серийному номеру сканера или к идентификатору оборудования на котором работает SDK. Лицензия проверяется в функциях

- LSC_CheckQuality()
- LSC_CheckEncode()
- LSC_CompressWsqPpln()
- LSC_DecompressWsqPpln()

Аргументы:

cl - указатель на структуру клиентского приложения.
_path -путь до файла лицензии, обычно называется lssdk.lic

Возвращает: возвращаемые функциями ошибки.

Примеры: mainform.cpp.

Функции сжатия/разжатия WSQ

Эти функции реализуют сжатие и разжатие изображений методом WSQ.

Функции

LSCLIENT_API LSC_ERROR LSC_CompressWsqPpln	**(LSClient ***	**cl,**
	unsigned char *	**image,**
	int	**width,**
	int	**height,**
	int	**bps,**
	int	**spp,**
	double	**compression,**
	unsigned char **	**wsq_rec,**
	int *	**wsq_len**
	)	

Сжатие изображения с использованием WSQ.

Аргументы:

[in]	*cl*	- объект библиотеки.
[in]	*image*	- указатель на буфер с изображением
[in]	*width*	- количество колонок изображения
[in]	*height*	- количество строк изображения
[in]	*bps*	- количество бит в одной компоненте, на данный момент поддерживатеся только 8.
[in]	*spp*	- количество цветовых компонент в пикселе, поддерживается только 1 для серых и 3 для RGB.
[in]	*compression*	- коэффициент сжатия, меняется от MIN_COPMPRESSION до MAX_COPMPRESSION.
[out]	*wsq_rec*	- указатель для сохранения указателя на выделенную память со сжатым изображением, память выделяется внутри функции

и должна быть освобождена функцией LSC_FreeWsqPpln() после использования.

`[out]` *wsq_len* - указатель для сохранения длинны получившегося сжатого изображения.

Возвращает: возвращаемые функциями ошибки.

Примеры: mainform.cpp.

```
LSCLIENT_API LSC_ERROR LSC_ DecompressWsqPpln    ( LSClient *          cl,
                                                   unsigned char *     wsq_rec,
                                                   int                 wsq_len,
                                                   unsigned char *     image,
                                                   int *               width,
                                                   int *               height,
                                                   int                 bps,
                                                   int                 spp
                                                 )
```

Разжатие изображения WSQ.

Аргументы:

`[in]` *cl* - объект библиотеки.

`[in]` *wsq_rec* - указатель на сжатое изображение.

`[in]` *wsq_len* - длина сжатого изображения в байтах.

`[out]` *image* - указатель для сохранения указателя на выделенный буфер с разжатым изображением, память выделяется внутри функции и должна быть освобождена функцией LSC_FreeWsqPpln() после использования.

`[out]` *width* - указатель для сохранения количества колонок в изображении.

`[out]` *height* - указатель для сохранения количества строк в изображении.

`[in]` *bps* - количество бит в одной цветовой компоненте сжатого изображения, поддерживается только 8.

`[in]` *spp* - количество цветовых компонент в пикселе, 1 для серых и 3 для RGB изображений.

Возвращает: возвращаемые функциями ошибки.

Примеры: mainform.cpp.

```
LSCLIENT_API LSC_ERROR LSC_ FreeWsqPpln    ( LSClient *          cl,
                                             unsigned char *     rec,
                                           )
```

Освобождение памяти захваченной в функциях LSC_CompressWsqPpln() и LSC_DecompressWsqPpln().

Аргументы:

`[in]` *cl* - объект библиотеки.
`[in]` *rec* - указатель на память, которая должна быть освобождена.

Примеры: mainform.cpp.

Классы

Классы с их кратким описанием.

- **LSC_SCANNERS_LIST** - Структура для описания списка подключенных сканеров
- **LSClient** - Структура, передаваемая во все функции