Считыватель документов «Регула»:
модели 70x3, 70x4, 70x7, 70x8, 4820, 8307, 83x3, 83x4

Программный пакет разработчика
Версия 5.4

Руководство программиста

РГВИ.01.01.00 М3

Регула 2020

Редакция от 17.12.2020

ОГЛАВЛЕНИЕ

СПИСОК СОКРАЩЕНИЙ

SDK – **Software Development Kit** – программный пакет разработчика

СИС – справочно-информационная система, содержащая сведения об основных признаках подлинности паспортов, идентификационных и проездных документов, документов на право управления и распоряжения автотранспортными средствами на пяти уровнях системы защиты: полиграфия, ультрафиолетовая 365 нм и 254 нм, инфракрасная и специальные материалы

OCR – **Optical Character Recognition** – оптическое распознавание символов

MRZ (МСЗ) – **Machine Readable Zone** – машиносчитываемая зона (документа) - МСЗ

RFID – **Radio Frequency Identification** – радиочастотная идентификация

RFID-микросхема – бесконтактная идентификационная микросхема

MCL – **Main Control Library** – главная управляющая библиотека SDK

ВД – владелец документа

ВУ – водительское удостоверение

ИК – инфракрасный (диапазон спектра) свет

ОС – операционная система

ПК – персональный компьютер

ТС – транспортное средство

УФ – ультрафиолетовый (диапазон спектра) свет

ВВЕДЕНИЕ

Данное «*Руководство программиста*» описывает порядок использования SDK при разработке пользовательских приложений для работы с приборами «Регула» моделей 70x3, 70x4, 70x7, 70x8, 4820, 83x3, 83x4, 8307 (расшифровку обозначений символов «x» см. в «*Руководстве по эксплуатации*»).

МИНИМАЛЬНЫЕ СИСТЕМНЫЕ ТРЕБОВАНИЯ

Процессор	Intel Core 2 1.8 GHz
RAM	1 Gb
VideoRAM	32 Mb
ОС	Windows Vista (ServicePack 2)
SystemBus	USB 2.0 HighSpeed
HDD свободное место	1 Gb

Дополнительное ПО	СИС SDU, FDS, Passport или Autodocs

1. СТРУКТУРА SDK

\ProgramData\Regula\Document Reader SDK\:

RegulaReader.ini	– файл конфигурации SDK;

\Windows\System32\:

borlndmm.dll cc32240mt.dll gmp.dll IJL15.dll ipp*7.0.dll libeay32.dll libiomp5md.dll libssl32.dll opencv_*343.dll rtl240.bpl ssleay32.dll tbb.dll tbbmalloc.dll tbbmalloc_proxy.dll vcl240.bpl vclimg240.bpl vclx240.bpl xmlrtl240.bpl	– системные библиотеки;

\Program Files\Regula\Drivers – драйвера устройств;

\Program Files\Regula\Document Reader SDK\:

PasspR40.dll devices.dat	– главная управляющая библиотека и ее файл данных;
RegulaReaderFX.dll Reader3M.dll	– библиотеки управления устройствами;
DRUster.dll, .ini	– служебная библиотека (проведение оценки геометрических искажений);
MRZCMP.dll mrzproc.dll, .ini TestOcrB.bin OCRBEXT.dat	– служебные библиотеки и сопутствующие файлы для проведения OCR MC3;
Imaging.dll	– служебная библиотека (чтение/запись файлов изображений);
bsdk_c50.dll bsdk50.dll ftrScanAPI.dll	– служебные библиотеки (поддержка сканера Biolink);
IntelIPPDlls.ini ippImProc.dll, .ini	– служебные библиотеки и сопутствующие файлы данных для проведения распознавания типа документа и чтения текстовых (OCR) и графических

	данных из полей заполнения документов, а также автоматической проверки подлинности документа;
READERDEMO.exe, Translator.xml	– демонстрационная программа SDK и файл локализации;
Dbgview.exe	– программа Debug View для анализа журнала;

\Program Files\Regula\Document Reader SDK\Data – директория, содержащая данные, необходимые для обработки документа (классификация типа, OCR и т. д.);

\Program Files\Regula\Document Reader SDK\Languages – директория, содержащая языковые файлы;

\Program Files\Regula\:

RegulaQS.url	– ссылка на Regula Quick Support;

\Windows\twain_32\:

RegulaReaderDS.ds	– файл интерфейса TWAIN для работы со считывателем;

\Program Files\Regula\Samples – директория, содержащая проекты тестовых приложений, иллюстрирующих использование программных средств SDK;

\Program Files\Regula\Samples\Include\:

PasspR.h	– заголовочный файл SDK (C++);
PasspR.pas	– заголовочный файл SDK (Delphi);
READERDEMO.tlb READERDEMO_TLB.pas	– библиотека типов SDK (Delphi);

\Program Files\Regula\Document Reader SDK\Doc:

testobject #1.pdf	– изображения тест-объекта № 1;
testobject #2.pdf	– изображения тест-объекта № 2;
Documents List.pdf	– содержимое текущей базы данных документов SDK;
Documents List Autodocs.pdf	– содержимое текущей базы данных документов SDK (только автодокументы);
Documents List Passports.pdf	– содержимое текущей базы данных документов SDK (только проездные документы);
Documents List Other.pdf	– содержимое текущей базы данных документов SDK (остальные документы);
SDK Release Notes (en).pdf	– замечания к выпуску SDK на английском языке;
SDK Release Notes (ru).pdf	– замечания к выпуску SDK на русском языке;

Programmers Guide (en).pdf	– данное Руководство на английском языке;
Programmers Guide (ru).pdf	– данное Руководство на русском языке;
COM interface documentation.pdf	– документация на COM-объект;
Test Application(en).pdf Test Application(ru).pdf	– описание демонстрационной программы.

2. ВОЗМОЖНОСТИ SDK

SDK позволяет:

- получать изображения от считывателей документов моделей 70x3, 70x4, 70x7, 70x8, 4820, 83x3, 83x4, 8307 с использованием различных схем освещения;
- локализовать страницу документа на изображении;
- производить чтение текстовых данных, проверять подлинность и оценивать качество печати МСЗ;
- осуществлять поиск и чтение штрихкодов;
- производить чтение данных из текстовых и графических полей заполнения документа;
- автоматически проверять подлинность документа путем анализа изображений, полученных с использованием различных схем освещения;
- проводить сравнительный анализ текстовых данных, полученных при распознавании МСЗ, полей заполнения документа, чтении штрихкодов и данных, считанных из памяти RFID-микросхемы проездного документа (при совместном использовании со средствами «*SDK для считывателей бесконтактных идентификационных микросхем*»);
- вызывать образцы изображений и описание документа из СИС для проведения визуального сравнительного анализа с изображениями обрабатываемого документа.

3. УСТАНОВКА И ИСПОЛЬЗОВАНИЕ СРЕДСТВ SDK

Для установки программного пакета необходимо запустить **Regula Document Reader SDK.exe** из директории **\SDK** инсталляционного носителя. При этом происходит установка всех необходимых для работы программных компонентов, включая драйвера считывателя документов. Драйвера будут установлены автоматически. Пользователь должен подтвердить установку, если система спросит его об этом.

ВНИМАНИЕ! Установка должна производиться пользователем ОС, обладающим правами администратора.

Подключить устройство в свободный слот USB. При этом ОС сообщит о нахождении нового устройства и произведет активизацию драйвера. В случае запроса системы о местоположении отдельных компонент драйвера необходимо указать путь к директории **\Program Files\Regula\Drivers** или соответствующей директории установочного носителя. При успешной активизации драйвера индикаторная лампочка состояния прибора загорится оранжевым светом (см. «*Руководство по эксплуатации*»).

Для использования программных средств SDK в пользовательском проекте необходимо:

1) подключить заголовочный файл **PasspR.h** с описаниями экспортируемых из **PasspR40.dll** функций, используемых структур данных и констант или заменить их соответствующими объявлениями, если приложение разрабатывается не на C++;
2) динамически подключить управляющую библиотеку **PasspR40.dll**, получить указатели на экспортируемые функции библиотеки с помощью функции Windows API **GetProcAddress()**.

Путь к библиотеке **PasspR40.dll** регистрируется в строковом поле **Path** ключа **HKEY_LOCAL_MACHINE\SOFTWARE\Regula\Document Reader SDK** в реестре при инсталляции SDK. Версия SDK указывается в строковом поле **Version** того же ключа.

В процессе функционирования главной управляющей библиотеки некоторые необходимые для ее работы данные (калибровочные сведения, файлы конфигурации, файлы протоколирования при работе в отладочном режиме) сохраняются в директории `\Users\[User Name]\AppData\Local\Regula\Document Reader SDK`.

4. РАБОТА СО СЧИТЫВАТЕЛЯМИ ДОКУМЕНТОВ

4.1. Общие сведения

Различные модели приборов обладают различной функциональностью и могут быть классифицированы по следующим признакам:

1. **Формату обрабатываемого документа** (ISO/IEC 7810):
   - для документов формата `ID1` (модели 70x3);
   - документов формата `ID1`–`ID3` (модели 70x4, 70x7, 70x8,4820, 83x3 и 83x4);
   - только МСЗ для документов формата `ID1` (модель 8307).

2. **Типу осветителей**:
   - общий белый свет;
   - белый свет с раздельным управлением боковыми и верхним/нижним осветителями;
   - общий ИК-свет;
   - ИК-свет с раздельным управлением боковыми и верхним/нижним осветителями;
   - общий УФ-свет;
   - коаксиальный белый свет с раздельным управлением левым и правым осветителями;
   - белый свет с матричным управлением, позволяющим контролировать состояние каждого точечного источника света, входящего в состав осветителей;
   - ИК-свет с матричным управлением, позволяющим контролировать состояние каждого точечного источника света, входящего в состав осветителей.

Программные средства SDK поддерживают работу со всеми вышеперечисленными типами считывателей документов в полном объеме.

В качестве дополнительного устройства считыватели документов могут комплектоваться **считывателем RFID-микросхем**, работа с которым должна производиться в соответствии с рекомендациями «*SDK для считывателей бесконтактных идентификационных микросхем*».

4.2. Организация работы с главной управляющей библиотекой

Главная управляющая библиотека SDK **PasspR40.dll** экспортирует ряд функций, с помощью которых осуществляется работа со считывателями документов.

Их работа организована с учетом возможного использования многопоточной схемы обработки данных. Вызов любых функций библиотеки (за небольшим исключением) может производиться из разных потоков пользовательского приложения. Это, например, дает возможность организовать сканирование и обработку данных в фоновом режиме, оставляя основной интерфейс программы незаблокированным.

Внимание! Выполнение функций MCL **_Initialize()**, **_Free()**, а так же выполнение команд **RPRM_Command_Device_Connect** и **RPRM_Command_Device_Disconnect** должно выполняться в главном потоке программы.

Библиотека **PasspR40.dll** разработана для проведения динамического подключения с помощью Windows API функции **LoadLibrary()**. Указатели на экспортируемые функции можно получить с использованием Windows API функции **GetProcAddress()**.

После загрузки библиотеки в память необходимо произвести вызов функции инициализации **_Initialize()**. Она объявляет необходимые для дальнейшей работы системные ресурсы, производит поиск и строит список всех подключенных к ПК в данный момент времени считывателей документов.

Основной функцией библиотеки, посредством которой пользовательское приложение может инициировать все необходимые действия для работы со считывателями документов, является функция **_ExecuteCommand()**. В качестве параметров она принимает командный триплет: *код команды*, *входной параметр команды* и *указатель на контейнер-приемник* для возвращаемых результатов.

Выполнение управляющей библиотекой всех команд проводится по синхронной схеме. Это означает, что на момент возврата из функции **_ExecuteCommand()** в вызывающую ее функцию пользовательского приложения запрошенное действие полностью завершено, все возможные результаты выполнения команды получены и являются действительными.

Для получения подробной информации о текущих действиях, происходящих в процессе выполнения команд, используется механизм callback-функции. С помощью экспортируемой функции **_SetCallbackFunc()** можно инициализировать указатель на функцию пользовательского приложения, имеющую тип **NotifyFunc**, которая

будет вызываться на различных этапах выполнения той или иной команды с представлением кода события и дополнительных сведений в контексте данного события. При нахождении в callback-функции повторный вызов **`_ExecuteCommand()`** приведет к блокировке выполняющегося потока. Допускается повторный вызов **`_ExecuteCommand()`** только с командой управления индикаторами **`RPRM_Command_Device_LED`**.

В ряде особых случаев callback-функция может вызываться асинхронно, т. е. вне контекста текущего рабочего потока приложения, например, при срабатывании датчика наличия документа.

Запрашиваемые результаты сканирования и обработки данных поступают в пользовательское приложение через другую callback-функцию, имеющую тип **`ResultReceivingFunc`**, которая также устанавливается вызовом **`_SetCallbackFunc()`**.

Все результаты, сформированные в процессе последнего цикла сканирования документа и последующей обработки полученных изображений, остаются доступными также для чтения посредством вызовов функций **`_CheckResult()`** и **`_CheckResultFromList()`** до момента проведения следующего цикла сканирования. Это дает возможность приложениям, разрабатываемым в средах программирования с существующими объективными ограничениями в работе со сложными типами данных, получать доступ к результатам с использованием упрощенного механизма передачи данных (через буфер обмена, файл, в текстовом виде, формате XML или JSON и т. п.).

По окончании работы с главной управляющей библиотекой необходимо вызвать функцию **`_Free()`** и выгрузить библиотеку из памяти с помощью Windows API функции **`FreeLibrary()`**.

Успех выполнения той или иной функции или команды определяется кодом возврата при ее выполнении. Набор возможных значений определен в перечислении **`eRPRM_ErrorCodes`**.

4.3. Подключение/отключение считывателя документов

При подключении считывателя документов к свободному USB 2.0 разъему ПК ОС определит его наличие, произведет активацию драйвера и первичную инициализацию. При этом в *Диспетчере устройств* считыватель документов будет зарегистрирован в разделе «`Regula Document Readers`» либо «`Forensic Devices`».

Главная управляющая библиотека SDK поддерживает работу с любым количеством считывателей, одновременно подключенных к ПК, но в один момент времени доступным для обработки может быть лишь один из них.

Для определения общего количества считывателей, подключенных к ПК в текущий момент времени, служит команда **`RPRM_Command_Device_Count`**.

Каждому конкретному устройству в качестве идентификатора ставится в соответствие его порядковый номер (индекс) в общем списке. Для получения индекса устройства, с которым установлена связь в текущий момент, служит команда **`RPRM_Command_Device_ActiveIndex`**.

Для получения информации о каком-либо устройстве из общего списка служит команда **`RPRM_Command_Device_Features`**. Данная команда заполняет структуру **`TRegulaDeviceProperties`** информацией о считывателе документов с указанным индексом, что дает пользовательскому приложению возможность определить конкретное устройство, которое необходимо подключить для дальнейшей работы.

Поскольку считыватели документов, являясь USB-устройствами, могут отключаться и подключаться к ПК в любой момент времени, список подключенных устройств теряет свою актуальность. Для обновления списка подключенных к ПК считывателей документов служит команда **`RPRM_Command_Device_RefreshList`**. В отличие от процедуры поиска подключенных устройств при выполнении функции **`_Initialize()`** эта команда лишь проверяет наличие устройства из ранее сформированного списка, и, если произошло отключение устройства, удаляет его из списка. Для обнаружения ***новых*** устройств необходимо провести процедуру переинициализации `MCL` – произвести последовательный вызов **`_Free()`**/**`_Initialize()`**.

Для подключения конкретного считывателя из общего списка служит команда **`RPRM_Command_Device_Connect`**.

Все команды на чтение данных и все получаемые сообщения о появлении документа в считывателе будут предназначаться (соответствовать) только текущему активному устройству.

В случае, если калибровочная информация считывателя отсутствует на ПК, она скачивается из считывателя. В ходе этого процесса вызывается callback-функция **NotifyFunc** с сообщением **RPRM_Notification_DownloadingCalibrationInfo** и с параметром, содержащим текущее значение процесса в процентах.

Для завершения работы с текущим считывателем документов служит команда **RPRM_Command_Device_Disconnect**.

Для переключения с одного считывателя документов на другой пользовательское приложение должно сначала отключить текущее устройство командой **RPRM_Command_Device_Disconnect** и лишь затем для другого считывателя выполнять **RPRM_Command_Device_Connect**.

При необходимости (в случае отсутствия ее вызова со стороны пользовательского приложения) команда **RPRM_Command_Device_Disconnect** выполняется автоматически при вызове **_Free()** или во время выгрузки библиотеки из памяти.

Если программно подключенное устройство было физически отключено от USB-порта ПК, будет вызвана callback-функция **NotifyFunc** с кодом сообщения **RPRM_Notification_DeviceDisconnected**. В этом случае необходимо завершить текущую сессию работы со считывателем, выполнив команду **RPRM_Command_Device_Disconnect**.

Код возврата **RPRM_Error_NoError** (или **RPRM_Error_AlreadyDone**) из функции **_ExecuteCommand()** при выполнении команды **RPRM_Command_Device_Connect** означает, что устройство успешно подключено и полностью готово к работе. Любой другой код возврата говорит о возникновении ошибок и невозможности дальнейшей работы с устройством.

4.4. Определение функциональных возможностей считывателя документов

Одним из этапов подключения считывателя документов является определение набора функций получения изображений и обработки данных, доступных при работе с этим устройством.

Битовую комбинацию флагов, определяющую функциональные возможности по обработке данных для активного считывателя, можно получить с помощью команды **`RPRM_Command_Options_GetSDKCapabilities`**.

Набор функций получения изображений определяется доступными схемами освещения считывателя. Для получения полного их списка необходимо воспользоваться командами опроса *списка схем освещения для сканирования*. При подключении считывателя происходит заполнение этого *списка* значениями по умолчанию, которыми и являются идентификаторы всех доступных схем освещения.

Для принудительного заполнения *списка* значениями по умолчанию служит команда **`RPRM_Command_Device_Light_ScanList_Default`**. Для получения количества элементов *списка* – **`RPRM_Command_Device_Light_ScanList_Count`**. Для получения значения элемента *списка* – **`RPRM_Command_Device_Light_ScanList_Item`**.

Таким образом, сразу после подключения считывателя документов пользовательское приложение должно опросить *список схем освещения для сканирования*, чтобы получить полный набор доступных схем освещения. В дальнейшем для проведения сканирования можно указывать только эти значения. Любые другие идентификаторы схем освещения будут игнорироваться.

4.5. Процедура калибровки

Важнейшим этапом подготовки считывателя документов к полноценной работе является процедура **калибровки**. Она нужна для того, чтобы определить необходимые параметры работы цифровой камеры (коэффициент усиления, баланс белого цвета) для каждой схемы освещения и подготовить данные, которые впоследствии будут использованы для компенсации *геометрических искажений*, *неравномерности освещения* и *цветового баланса* в получаемых изображениях.

Все приборы откалиброваны и требуют перекалибровки только в случаях нарушения качества получаемых изображений, каждый раз после использования команды **`RPRM_Command_Device_SetFrequencyDivider`** и изменения делителя частоты, при переходе с питания устройства по USB-кабелю на питание от внешнего источника и наоборот.

Процедура калибровки инициируется командой **`RPRM_Command_Device_Calibration`**.

На первом этапе калибровки будут определены необходимые параметры цифровой камеры для каждой схемы освещения – устройство последовательно включит соответствующие осветители. Вначале будет проведен полный цикл сканирования чистого (белого) тест-объекта, изображения которого будут использоваться для *компенсации неравномерности освещения* и *цветового баланса*. Вслед за этим – полный цикл сканирования тест-объекта в виде шахматной доски, данные обработки изображений которого необходимы для *компенсации геометрических искажений*.

Набор схем освещения для процедуры калибровки определяется *списком схем освещения для сканирования* по умолчанию.

Начало и окончание процедуры калибровки устройства отмечается вызовами callback-функции **`NotifyFunc`** с кодом сообщения **`RPRM_Notification_Calibration`** с параметром «`false`» и «`true`» соответственно.

Прогресс проведения калибровки отмечается вызовами callback-функции **`NotifyFunc`** с кодом сообщения **`RPRM_Notification_CalibrationProgress`** и параметром, содержащим значение степени выполнения калибровки в процентах.

Для представления пользователю диалога с просьбой вставить тот или иной калибровочный тест-объект используется вызов callback-функции **`NotifyFunc`** с кодом сообщения **`RPRM_Notification_CalibrationStep`** и параметром, содержащим фазу калибровки:

- `0` – необходим калибровочный тест-объект № 1 (белый);
- `1` – необходим калибровочный тест-объект № 2 (шахматная доска).

Полученные калибровочные данные сохраняются в памяти прибора и на диске в директориях `\Users\[User Name]\AppData\Local\Regula\Document Reader SDK \Calibration`.

Загрузка ранее полученных калибровочных данных входит в процедуру подключения считывателя документов к пользовательскому приложению.

Команда **`RPRM_Command_Device_IsCalibrated`** позволяет узнать, была ли проведена процедура калибровки для текущего активного считывателя и являются ли данные калибровки корректными.

Без проведения калибровки возможно только сканирование необрезанных изображений. Никакая их обработка выполняться не будет. Об этом будет сигнализировать код возврата из функций главной управляющей библиотеки **`RPRM_Error_DeviceNotCalibrated`**.

4.6. Сканирование и обработка данных

4.6.1. Представление и хранение данных

Результаты сканирования и обработки полученных изображений представляются единообразно с использованием структуры-контейнера **TResultContainer** (для отдельного результата) и списка **TResultContainerList** (для хранения и передачи нескольких результатов за одну операцию).

Для хранения различных типов результатов работы MCL существует набор соответствующих структур данных. Каждая из них может быть помещена в **TResultContainer** с целью передачи в пользовательское приложение.

Тип данных, содержащихся в **TResultContainer**, определяется полем result_type этой структуры. Оно может содержать одно из значений перечисления **eRPRM_ResultType** и однозначно определяет тип указателя, хранящегося в поле buffer. Пользовательское приложение должно лишь привести этот указатель к необходимому типу, чтобы получить доступ к содержимому структуры данных результата.

Данные результатов, полученных в одном цикле сканирования и обработки, сохраняются и после возврата из функции **_ExecuteCommand()** до момента ее следующего вызова для выполнения команд **RPRM_Command_Process** или **RPRM_Command_ProcessImagesList**.

4.6.2. Сканирование документа и получение результатов обработки изображений

Цикл сканирования и обработки получаемых изображений должен быть инициирован пользовательским приложением путем выполнения команды **RPRM_Command_Process**. При этом необходимо убедиться, чтобы в этот момент времени документ находился в считывателе.

О наличии документа в считывателе сигнализирует приход в callback-функцию **NotifyFunc** сообщения с кодом **RPRM_Notification_DocumentReady**. Значение параметра сообщения («true» или «false») показывает, что документ был соответственно помещен в считыватель или удален из него.

Перед тем как непосредственно запустить цикл сканирования и обработки изображений следует предварительно:

- определить набор данных, который необходимо получить в качестве результата;

- определить набор схем освещения, для которых необходимо получить отсканированные изображения документа (если выбран соответствующий тип запрашиваемого результата – **RPRM_GetImage_Modes_GetImages**);
- в случае необходимости получения изображения для схемы УФ-освещения установить для нее значение экспозиции сканирования;
- если в набор запрашиваемых данных входит результат проверки качества заполнения МСЗ (**RPRM_GetImage_Modes_OCR_TestMRZQuality**) – установить параметры проведения проверки.

Набор запрашиваемых результатов формируется логической комбинацией (операцией «OR») значений из перечисления **eRPRM_GetImage_Modes** и передается в качестве параметра команды **RPRM_Command_Process**. Только выбранные таким образом результаты будут переданы пользовательскому приложению в процессе выполнения команды.

Набор необходимых схем освещения формируется заполнением *списка схем освещения для сканирования*.

Команда **RPRM_Command_Device_Light_ScanList_Clear** очищает текущий *список*.

Команда **RPRM_Command_Device_Light_ScanList_AddTo** заносит в *список* новый элемент. Если указанный идентификатор схемы освещения не входит в состав *списка схем освещения по умолчанию*, он игнорируется, что индицируется кодом возврата **RPRM_Error_LightIsNotAllowed**.

Команда **RPRM_Command_Device_Get_ParamLowLight** возвращает текущую величину экспозиции сканирования для схемы УФ-освещения.

Команда **RPRM_Command_Device_Set_ParamLowLight** устанавливает величину экспозиции сканирования для схемы УФ-освещения.

В дополнение к схеме освещения могут быть установлены опциональные параметры “Умный УФ” и “Компенсация бликов”. Их установка осуществляется вызовом команд **RPRM_Command_Options_Set_SmartUV** и **RPRM_Command_Options_Set_GlareCompensation** соответственно.

В случае если пользовательское приложение установило callback-функцию **ResultReceivingFunc** для получения результатов, она будет вызываться по мере формирования результатов сканирования и последующей обработки изображений. В качестве параметра эта функция получает указатель на структуру **TResultContainer**, содержащую данные результата того или иного типа.

В процессе выполнения цикла сканирования и обработки данных возможен также вызов другой callback-функции – **NotifyFunc** – для передачи информационных сообщений.

Начало и окончание сканирования изображений документа отмечается приходом сообщения **RPRM_Notification_Scanning** с параметром «false» при начале сканирования и «false» при его окончании.

Начало и окончание обработки изображений документа отмечается приходом сообщения **RPRM_Notification_Processing** с параметром «false» при начале обработки и «false» при ее окончании.

Получение сообщения **RPRM_Notification_DocumentCanBeRemoved** означает, что документ может быть убран из считывателя. До этого момента наличие неподвижного документа в считывателе обязательно.

Получение сообщения **RPRM_Notification_Error** с параметром, содержащим один из кодов **eRPRM_ErrorCodes**, говорит о возникновении некритической ошибки, чаще всего – о невозможности формирования какого-либо из запрошенных результатов (например, когда был запрошен результат чтения МСЗ, а сама МСЗ не была обнаружена на документе).

Код возврата из **_ExecuteCommand()** говорит об успешности завершения цикла сканирования и обработки данных. При его значении, отличном от **RPRM_Error_NoError**, операция завершилась с ошибкой или была выполнена не в полном объеме. В любом случае все уже полученные пользовательским приложением результаты остаются действительными до момента запуска следующего цикла сканирования.

4.6.3. Получение результатов прямым опросом

Кроме использования механизма получения результатов сканирования и обработки данных через callback-функцию **ResultReceivingFunc** существует возможность доступа к ним уже после возврата из **_ExecuteCommand()**.

Для реализации дополнительного способа получения результатов чтения данных используются следующие экспортируемые функции главной управляющей библиотеки: **_ResultTypeAvailable()**, **_CheckResult()** и **_CheckResultFromList()**.

Этот способ получения результатов следует использовать в случаях:

- необходимости представления результатов в формате XML;
- необходимости записи полученных изображений или XML-представления результата в файл;
- если пользовательское приложение не использует механизм получения результатов через callback-функцию;

- при разработке пользовательского приложения в среде программирования с ограниченными возможностями использования сложных структурированных типов данных. В этом случае предлагаемый способ обеспечивает доступ к результатам с использованием простых стандартных типов данных (Windows DIB, текст) и способов обмена ими (через буфер обмена ОС или файл);
- для получения дополнительных результатов обработки изображений (например, **RPRM_ResultType_ChosenDocumentTypeCandidate**).

Общее количество (соответственно, их доступность) результатов определенного типа, сформированных за последний цикл сканирования и обработки данных, возвращается функцией **_ResultTypeAvailable()**.

В качестве входных параметров **_CheckResult()** принимает тип запрашиваемых данных (одно из значений **eRPRM_ResultType** в параметре функции **type**), индекс объекта в общем списке доступных результатов запрашиваемого типа (параметр **idx**), формат и механизм передачи данных (OR-комбинация значений **eRPRM_ OutputFormat** в параметре функции **output**) и параметры передачи данных (параметр **param** в контексте значения **output**).

Если возвращаемое **_CheckResult()** значение «*меньше* 0», то оно является одним из кодов ошибки **eRPRM_ResultStatus**. Если возвращаемое значение «*больше* 0», то оно фактически является указателем на структуру **TResultContainer**, содержащую запрашиваемые данные. Его можно использовать непосредственно, приведя к типу **TResultContainer***, либо для доступа к полям структур, являющихся списками однотипных данных, через функцию **_CheckResultFromList()**. Во втором случае вызов функции **_CheckResult()** является промежуточным шагом для получения конкретных данных.

Кроме получения указателя на соответствующую структуру данных результата, дополнительную форму представления данных можно формировать комбинацией значений из перечисления **eRPRM_OutputFormat** в параметре **output**:

- для записи *изображений* (тип результата **RPRM_ResultType_RawImage**) в файл необходимо указать комбинацию **ofrFormat_FileBuffer** и **ofrTransport_File**. Формат записи будет соответствовать текущему установленному формату записи графических файлов. Указатель на символьную строку с именем файла в кодировке UTF8 должен передаваться в параметре **param**;
- для формирования XML-представления структуры данных результата необходимо указать **ofrFormat_XML**. В этом случае поле **XML_buffer** возвращаемой структуры **TResultContainer** будет содержать указатель на начало текстового буфера XML-образа, а поле **XML_length** – длину этого буфера. Добавление **ofrTransport_File** указывает на необходимость записи сформированного XML-представления результата в файл, имя которого в кодировке UTF8 передается в **param**. В случае если **ofrFormat_XML** находится в комбинации

с **ofrFormat_FileBuffer**, в XML-форму будет преобразовано содержимое буфера, включающего образ графического файла – результата записи изображения;
- для передачи изображения или текста XML-представления результата через буфер обмена ОС необходимо включить **ofrTransport_Clipboard** в комбинацию **output**.

Для доступа к отдельным полям структур, являющихся списками однотипных текстовых (**TDocVisualExtendedInfo**) или графических (**TDocGraphicsInfo**) данных, служит функция **_CheckResultFromList()**.

В качестве входных данных **_CheckResultFromList()** принимает в параметре **container** дескриптор списка результатов, полученный при вызове **_Check Result()**, в параметре **output** – идентификатор механизма передачи данных (одно из значений **eRPRM_OutputFormat**) и в **param** – специфические (в контексте значения **output**) параметры передачи данных. Возвращаемое функцией значение содержит тип передаваемого поля (одно из значений **eVisualFieldType** или **eGraphicFieldType**). В случае возникновения ошибки или при достижении конца списка полей возвращается одно из значений **eRPRM_ResultStatus**.

В отличие от **_CheckResult()** параметр **output** функции **_CheckResultFromList()** должен содержать не комбинацию значений из **eRPRM_OutputFormat**, а конкретное значение из этого перечисления, определяющее способ передачи содержимого полей списка результатов:
- **ofrTransport_Clipboard** – через буфер обмена Windows (для текстовых полей структуры **TDocVisualExtendedInfo** и графических изображений из структуры **TDocGraphicsInfo**);
- **ofrTransport_File** – через файл (для графических изображений из структуры **TDocGraphicsInfo**).

В первом случае параметр функции **param** игнорируется. Во втором случае он должен содержать указатель на символьную строку с именем файла. Формат записи будет соответствовать текущему установленному формату записи графических файлов.

Команда **RPRM_Command_Options_GraphicFormat_Count** возвращает общее количество доступных графических форматов записи изображений.

Команда **RPRM_Command_Options_GraphicFormat_Name** возвращает символьную строку с расширением файла для указанного индекса формата записи изображений.

Выбор формата записи изображений по индексу производится командой **RPRM_Command_Options_GraphicFormat_Select**.

Команда **RPRM_Command_Options_GraphicFormat_ActiveIndex** возвращает индекс текущего формата записи изображений в общем списке.

Таким образом, для получения содержимого всех текстовых или графических полей, содержащихся в структурах-списках, необходимы следующие действия:

- вызов **_CheckResult()** с указанием соответствующего типа запрашиваемого результата;
- вызов **_CheckResultFromList()** с использованием полученного дескриптора результата до момента достижения конца списка (код возврата – `RPRM_ResultStatus_EndOfList`).

После возврата из **_CheckResultFromList()** запрашиваемые данные находятся либо в буфере обмена, либо записаны в файл с указанным именем.

4.6.4. Обработка списка заранее полученных изображений

Для выполнения цикла обработки заранее полученных изображений документа служит команда **RPRM_Command_ProcessImagesList**. Она является полным аналогом команды **RPRM_Command_Process** с той только разницей, что при ее выполнении не происходит реального сканирования документа, а в качестве входных данных используется список изображений, передаваемый в параметре команды.

По умолчанию можно обработать только изображения, полученные ранее на этом же считывателе. Возможность **RPRM_Capabilities_ProcessImages**, позволяющая обработать изображения, полученные из другого источника, должна быть добавлена в считыватель по индивидуальному запросу.

Обязательным условием успешного выполнения данной команды является тот факт, что все передаваемые изображения должны быть полными (необработанными). Это достигается путем их получения в цикле сканирования без указания каких-либо дополнительных функций обработки, кроме **RPRM_GetImage_Modes_GetImages**. В этом случае получаемые в контейнере с типом **RPRM_ResultType_RawImage** изображения будут единственным результатом выполнения цикла сканирования.

В качестве параметра команды **RPRM_Command_ProcessImagesList** выступает список **TResultContainerList** структур-контейнеров **TResultContainer**, содержащих изображения документа. Изображения должны быть представлены в формате **TRawImageContainer**, на что должно указывать значение поля **result_type** каждой структуры **TResultContainer** списка (оно должно быть равно **RPRM_ResultType_RawImage**).

Существуют две экспортируемые функции главной управляющей библиотеки SDK, отвечающие за правильное выделение памяти для хранения изображений в виде **TRawImageContainer** и освобождение этой памяти. Это функции **_AllocRawImageContainer()** и **_FreeRawImageContainer()** соответственно. При создании копии получаемых изображений для последующего использования в команде **RPRM_Command_ProcessImagesList** рекомендуется, чтобы пользовательское

приложение оперировало этими сервисными функциями для гарантированно правильного распределения памяти.

4.7. Функции обработки данных и порядок их применения при проведении циклов сканирования и обработки

Как уже отмечалось, определение набора функций получения изображений и обработки данных, доступных при работе со считывателем документов, производится на этапе подключения устройства.

Битовую комбинацию значений из перечисления **eRPRM_Capabilities**, определяющую функциональные возможности по обработке данных для активного считывателя, можно получить с помощью команды **RPRM_Command_Options_GetSDKCapabilities**.

Набор действий, которые необходимо выполнить в процессе проведения *цикла сканирования и обработки данных*, и, соответственно, набор данных-результатов этих действий, требуемый пользовательским приложением, определяется параметром команды **RPRM_Command_Process** (или **RPRM_Command_ProcessImagesList**).

Далее приводится описание каждого из этапов получения и обработки данных в хронологической последовательности проведения цикла сканирования.

4.7.1. Получение отсканированных изображений

Для получения изображений документа в качестве результатов выполнения *цикла сканирования и обработки данных* необходимо указать **RPRM_GetImage_Modes_GetImages** в наборе функций цикла сканирования (в параметре команд **RPRM_Command_Process** или **RPRM_Command_ProcessImagesList**).

Набор схем освещения, для которых необходимо получить изображения, определяется содержимым *списка схем освещения для сканирования*. Каждое из соответствующих изображений будет представлено в виде структуры данных **TRawImage Container** и передано через callback-функцию **ResultReceivingFunc** либо может быть получено через вызов **_CheckResult()**.

Помимо указанных в *списке* схем освещения для выполнения заявленных функций обработки могут понадобиться и изображения документа, снятые для схем освещения, не включенных в *список*. В этом случае они будут получены, но не будут включены в набор сформированных результатов для передачи пользовательскому приложению. Начало и окончание *основного* сканирования изображений документа отмечается сообщением **RPRM_Notification_Scanning** (с параметром «`false`» при начале сканирования и «`true`» по его окончании).

Список необходимых для сканирования схем освещения может быть также уточнен после проведения процедуры *определения типа документа*. В этом случае необходимые изображения будут получены *дополнительным* сканированием, после чего

пользовательское приложение получит сообщение **RPRM_Notification_DocumentCanBeRemoved**. Получение именно этого сообщения означает, что документ может быть убран из считывателя. До данного момента наличие *неподвижного* документа в считывателе обязательно.

Изображения документа, получаемые при дальнейшей обработке (вырезанные по границе документа, с проведенной компенсацией неравномерности освещения и коррекцией цветового баланса), будут переданы в пользовательское приложение при завершении выполнения команды **RPRM_Command_Process** (или **RPRM_Command_ProcessImagesList**). Для этих изображений тип результата будет установлен в **RPRM_ResultType_RawImage**.

4.7.2. Локализация документа на изображении

Для нахождения документа на отсканированных изображениях, а также формирования результирующих изображений документа, вырезанных по найденным границам, с проведенной компенсацией неравномерности освещения и коррекцией цветового баланса, необходимо указать **RPRM_GetImage_Modes_LocateDocument** в наборе функций цикла сканирования (в параметре команд **RPRM_Command_Process** или **RPRM_Command_ProcessImagesList**).

После успешного определения границ документа производится дополнительный анализ формата документа (по ISO/IEC 7810). В соответствии с его результатами возможен дополнительный поворот изображения.

4.7.3. Чтение и контроль качества заполнения машиносчитываемой зоны

Для получения результатов чтения МСЗ документа в качестве результата выполнения *цикла сканирования и обработки данных* необходимо указать **RPRM_GetImage_Modes_OCR_MRZ** в наборе функций цикла сканирования (в параметре команд **RPRM_Command_Process** или **RPRM_Command_ProcessImagesList**). Для получения результата контроля качества заполнения МСЗ – **RPRM_ResultType_MRZ_TestQuality**.

Результаты передаются в пользовательское приложение в виде структур **TDocVisualExtendedInfo** и **TDocMRZTestQuality** соответственно.

Для структуры `TDocVisualExtendedInfo` тип результата устанавливается равным `RPRM_ResultType_MRZ_OCR_Extended`, для `TDocMRZTestQuality` им является `RPRM_ResultType_MRZ_TestQuality`.

После чтения МСЗ в зависимости от ее месторасположения может быть выполнен дополнительный поворот изображений.

Параметры проверки качества заполнения МСЗ устанавливаются командой **`RPRM_Command_Options_Set_MRZTestQualityParams`**.

Команда **`RPRM_Command_Options_Get_MRZTestQualityParams`** возвращает текущие значения параметров проверки качества заполнения МСЗ.

Правильность текстового заполнения МСЗ определяется в соответствии с требованиями *ICAO 9303* для машиносчитываемых документов.

Качество печати текста в МСЗ проверяется в соответствии с ISO 1831:1980 «Характеристики печатного изображения для оптического распознавания символов».

По умолчанию выполняются проверки на предмет правильности заполнения МСЗ и контрольных сумм.

4.7.4. Определение типа документа

Для определения типа документа и получения этой информации в качестве результата выполнения цикла сканирования и обработки данных необходимо указать **`RPRM_GetImage_Modes_DocumentType`** в наборе функций цикла сканирования (в параметре команд **`RPRM_Command_Process`** или **`RPRM_Command_ProcessImagesList`**).

Определение типа документа основано на информации, хранящейся в базе данных документов, которая входит в состав SDK и копируется на диск в процессе его установки. По умолчанию база данных размещается в **`\<директория_установки_SDK>\Data\`**.

Данные о типе документа содержат разнообразную информацию, включая набор необходимых схем освещения для проведения последующих операций чтения полей заполнения документа и проведения проверки подлинности документа. Именно на основании этих данных происходит получение изображений *дополнительным* сканированием.

Существует *три режима процедуры определения типа документа*:

1. ***Стандартный режим***. В этом режиме работы тип документа определяется автоматически (путем анализа полученных изображений документа) и исключительно на основе данных из базы данных документов.

Информация об обнаруженных кандидатах на искомый тип документа передается в параметре **result** callback-функции **ResultReceivingFunc** в виде списка **TCandidatesListContainer**.

Пользовательское приложение должно выбрать один из предложенных кандидатов и поместить его идентификатор (значение поля **ID** структуры **TOneCandidate**) в возвращаемый параметр callback-функции **PostActionParameter**. При этом другой возвращаемый параметр **PostAction** должен содержать значение **RPRM_PostCalbackAction_ProcessCandidate**.

Если параметр **PostAction** будет содержать значение **RPRM_PostCalbackAction_Continue** по возвращении обратно в главную управляющую библиотеку, для дальнейших операций будет выбран ***первый*** элемент из списка предложенных кандидатов. То же самое произойдет и в случае, если callback-функция получения результатов **ResultReceivingFunc** не была установлена.

Если параметр **PostAction** будет содержать значение **RPRM_PostCalbackAction_Cancel** по возвращении обратно в главную управляющую библиотеку, дальнейшие операции, связанные с необходимостью точного определения типа документа (например, чтение полей заполнения документа), проводиться не будут.

2. ***Пользовательский режим***. В этом режиме работы тип документа определяется пользовательским приложением самостоятельно. Это возможно, например, если оно обладает собственным алгоритмом, на вход которого можно передать изображения (**RPRM_ResultType_RawImage**), полученные ранее в процессе выполнения текущего цикла сканирования и обработки.

В этом режиме вызов **ResultReceivingFunc** производится с пустым параметром **result**, и пользовательское приложение должно заполнить **PostAction** и **PostActionParameter** самостоятельно определенными значениями, имеющими такую же смысловую нагрузку, как и в *стандартном* режиме.

Тонким моментом здесь является тот факт, что идентификатор типа документа, помещаемый в **PostActionParameter**, должен являться действительным идентификатором документа, информация о котором содержится в базе данных документов.

Таким образом, должен существовать некий механизм, позволяющий пользовательскому приложению установить соответствие между типом документа, определенным по собственному алгоритму, и конкретным типом документа, описанным в существующей базе данных.

Одним из шагов, позволяющих установить такую связь, служит возможность получения информации обо всех типах документов, содержащихся в базе данных.

Она реализуется вызовом команды **RPRM_Command_Get_DocumentsInfoList**. Получаемый список **TListDocsInfo** содержит массив элементов **TOCRDocInfo**, каждый из которых соответствует одному зарегистрированному типу документа.

Помимо текстового названия документа, его формата и другой дополнительной информации, структура **TOCRDocInfo** содержит два идентификатора: один – числовой **DocID**, другой – символьный **DocTxtID**. Числовой идентификатор служит для указания типа документа при возврате из **ResultReceivingFunc** (в параметре **PostActionParameter**) и генерируется автоматически при внесении новой записи о документе в базу данных. Символьный же идентификатор может быть определен вручную и содержать какую-либо дополнительную информацию (например, внутренний идентификатор), позволяющую связать этот тип документа с данными, используемыми пользовательским приложением при самостоятельном определении типа документа.

Таким образом, определив по необработанным изображениям тип документа и имея его внутренний идентификатор, пользовательское приложение может осуществить его поиск среди всех документов базы данных, получить соответствующий цифровой идентификатор и далее использовать его для выбора типа документа при возврате из callback-функции.

Тип результата **RPRM_ResultType_DocumentsInfoList**, соответствующий списку **TListDocsInfo**, доступен также и через вызов **_CheckResult()**.

Активизация режима пользовательского определения типа документа производится командой **RPRM_Command_Options_Set_CustomDocTypeMode**.

Команда **RPRM_Command_Options_Get_CustomDocTypeMode** возвращает текущее состояние режима пользовательского определения типа документа (режим активен или нет).

3. ***Ручной режим***. В этом режиме работы тип документа, который необходимо выбрать для проведения дальнейших операций чтения данных, определяется заданием конкретного значения символьного идентификатора (поле **DocTxtID** структуры **TOCRDocInfo**).

Установка символьного идентификатора типа документа производится командой **RPRM_Command_Options_Set_CurrentDocumentType**, чтение его текущего значения – командой **RPRM_Command_Options_Get_CurrentDocumentType**.

Этот режим предназначен для использования при невозможности установки callback-функции для получения результатов **ResultReceivingFunc** и в ситуации, когда выбор первого элемента из списка найденных кандидатов (как при работе в *стандартном* режиме) не является оптимальным.

В этом случае, если текущий установленный идентификатор типа документа не найден в базе данных, процесс выполнения команды **RPRM_Command_Process** (или **RPRM_Command_ProcessImagesList**) будет прерван, и пользовательское приложение получит информационное сообщение **RPRM_Notification_Error** с кодом **RPRM_Error_CantRecognizeDocumentType**.

Таким образом, для определения типа документа в ручном режиме необходимо:

- проанализировать сформированный список кандидатов, запросив его через вызов **_CheckResult()**;
- выбрать нужный тип документа в списке кандидатов;
- в общем списке документов базы данных найти соответствующую запись, определить символьный идентификатор типа документа;
- командой **RPRM_Command_Options_Set_CurrentDocumentType** установить идентификатор типа документа;
- повторить цикл сканирования и обработки данных.

Ручной режим определения типа документа имеет ***наивысший*** приоритет среди всех режимов. Это означает, что, если текущий символьный идентификатор типа документа не пуст, поиск типа документа в базе данных будет всегда проводиться только по нему. Для перехода к *стандартному* или *пользовательскому* режиму необходимо установить командой **RPRM_Command_Options_Set_CurrentDocumentType** пустой текстовый идентификатор.

При выполнении команды **RPRM_Command_ProcessImagesList** существует возможность того, что предоставленный для обработки набор изображений не содержит изображения для всех необходимых схем освещения. И если в случае выполнения команды **RPRM_Command_Process** они могут быть досканированы, то при работе с фиксированным списком это невозможно.

В этом случае выполнение команды будет прервано, и **_ExecuteCommand()** вернет код ошибки **RPRM_Error_IncompleteImagesList**.

Для продолжения работы необходимо получить информацию о типе документа, который был определен на основании анализа текущего списка изображений, с помощью вызова **_CheckResult()** с запросом типа результата **RPRM_ResultType_ChosenDocumentTypeCandidate**. Возвращаемая структура данных **TOneCandidate** будет содержать информацию о необходимых схемах освещения в поле **Necessary Lights**. После этого нужно уточнить содержимое *списка схем освещения для сканирования*, провести цикл сканирования для получения соответствующих необработанных изображений и выполнить команду **RPRM_Command_ProcessImagesList** уже с обновленным списком изображений.

4.7.5. Чтение полей заполнения документа

Для получения результатов чтения текстовых и графических полей заполнения документа в процессе выполнения цикла сканирования и обработки данных необходимо указать **RPRM_GetImage_Modes_OCR_Visual** в наборе функций цикла сканирования (в параметре команд **RPRM_Command_Process** или **RPRM_Command_ProcessImagesList**).

Результаты передаются в пользовательское приложение в виде структур **TDocVisualExtendedInfo** и **TDocGraphicsInfo** соответственно.

Для структуры **TDocVisualExtendedInfo** тип результата устанавливается **RPRM_ResultType_Visual_OCR_Extended**, для **TDocGraphicsInfo** – **RPRM_ResultType_Graphics**.

Набор и характеристики считываемых полей заполнения документа определяются на основе информации для выбранного типа документа, хранящейся в базе данных документов. В случае, когда тип документа не определен, процедура чтения данных из текстовых полей заполнения документа не проводится.

Если геометрический формат документа, определенный на этапе локализации документа на изображении, соответствует формату документа `ID3`, но тип документа не определен, происходит выделение зоны фотографии по координатам, определенным в документе Doc 9303 ICAO. В этом случае результат типа **RPRM_ResultType_Graphics** будет содержать одно графическое поле.

4.7.6. Чтение штрихкодов

Для получения результатов чтения штрихкодов в процессе выполнения цикла сканирования и обработки данных необходимо указать **RPRM_GetImage_Modes_OCR_BarCodes** в наборе функций цикла сканирования (в параметре команд **RPRM_Command_Process** или **RPRM_Command_ProcessImagesList**).

Результаты передаются в пользовательское приложение в виде структур четырех различных типов:

- **TDocBarCodeInfo** (тип результата – **RPRM_ResultType_BarCodes**), содержащий полный набор данных штрихкода в двоичном виде;
- **TDocVisualExtendedInfo** (тип результата – **RPRM_ResultType_BarCodes_TextData**), содержащий логически разделенные по типам текстовые данные;
- **TDocGraphicsInfo** (тип результата – **RPRM_ResultType_BarCodes_ImageData**), содержащий логически разделенные графические данные;
- Массив байт (тип результата – **RPRM_ResultType_FingerprintTemplateISO**), содержащий шаблон отпечатка пальца в формате ISO в двоичном виде.

Если на предыдущем этапе обработки был определен тип документа, то будут читаться лишь штрихкоды, для которых заданы соответствующие области изображения документа в качестве полей заполнения и информация о которых содержится в базе данных.

Если тип документа определен не был, то поиск областей штрихкодов на изображении и считывание данных из них будут произведены автоматически.

Для дополнительной настройки формата считываемых данных модулей штрихкода (массив `mData` структуры **`TIP_DECODE_MODULE`**) служит файл настройки **`IPDecode.ini`**, находящийся в директории установки SDK. В его полях `ErrorByteCode` и `ErrorTextCode` можно указать ASCII-коды символов, которые будут помещаться в результирующий массив данных (для байтовых и текстовых модулей соответственно) вместо ошибочно считанных символов.

4.7.7. Проверка подлинности документа

Для получения результатов проверки подлинности документа в процессе выполнения цикла сканирования и обработки данных необходимо указать **`RPRM_GetImage_Modes_Authenticity`** в наборе функций цикла сканирования (в параметре команд **`RPRM_Command_Process`** или **`RPRM_Command_ProcessImagesList`**).

Набор необходимых проверок подлинности устанавливается командой **`RPRM_Command_Options_Set_AuthenticityCheckMode`**.

Результаты передаются в пользовательское приложение в виде структуры **`TAuthenticityCheckList`** с типом результата **`RPRM_ResultType_Authenticity`**.

Проверка подлинности документа проводится на основании обработки и анализа полученных изображений документа для различных схем освещения.

4.7.8. Сравнительный лексический анализ данных

Лексический анализ данных позволяет сопоставить результаты чтения текстовых данных МСЗ, зоны заполнения документа, штрихкодов и данных из памяти RFID-микросхемы (при совместной работе с «*SDK для считывателей бесконтактных идентификационных микросхем*») для дополнительной оценки подлинности документа.

Команда **`RPRM_Command_OCRLexicalAnalyze`** служит для проведения лексического анализа данных.

В качестве сравниваемых данных используются текстовые результаты чтения МСЗ, зоны заполнения документа и штрихкодов, полученные в последнем цикле сканирования и обработки. Данные из памяти RFID-микросхемы могут передаваться в качестве параметра команды в виде указателя на структуру **`TDocVisualExtendedInfo`**

(см. «*Считыватель бесконтактных идентификационных микросхем. Программный пакет разработчика. Руководство программиста*»).

Результаты передаются в пользовательское приложение в виде структуры **`TListVerifiedFields`** с типом результата **`RPRM_ResultType_OCRLexicalAnalyze`**.

Даты приводятся к формату текущих настроек локализации системы.

4.8. Дополнительные настройки и опции

В качестве дополнительных настроек и опций при работе с главной управляющей библиотекой выступают параметры, описанные в пп. 4.8.1–4.8.6.

4.8.1. Управление индикаторными светодиодами считывателя документов

Для управления индикаторными светодиодами считывателя документов служит команда **`RPRM_Command_Device_LED`**. С ее помощью пользовательское приложение может предложить собственный алгоритм поведения индикаторных светодиодов при выполнении различных команд. В этом случае новые параметры поведения индикаторов необходимо устанавливать при получении нотификационных сообщений и при возврате из функций `MCL` для того, чтобы перекрыть установленные к этому моменту значения по умолчанию.

По умолчанию световая индикация работает следующим образом (задействован только первый индикатор – «*состояние прибора*»):

- в процессе подключения устройства, калибровки и сканирования индикатор мигает оранжевым светом;
- при успешном подключении и в момент окончания сканирования индикатор загорается зеленым светом;
- при отключении устройства индикатор загорается оранжевым светом.

4.8.2. Контроль степени зарядки аккумуляторной батареи при работе с «Регула» 83x3, 83x4

Для контроля степени зарядки аккумуляторной батареи служит команда `RPRM_Command_Options_Get_BatteryStatus`.

4.8.3. Управление рабочим видеорежимом цифровой камеры считывателя

Для переключения рабочего видеорежима цифровой камеры считывателя, например с 3-мегапиксельного режима в 1-мегапиксельный, служит команда **`RPRM_Command_Device_Set_WorkingVideoMode`**.

Доступность различных видеорежимов работы камеры считывателя указывается в поле `VideoModes` структуры `TRegulaDeviceProperties`, возвращаемой по команде **`RPRM_Command_Device_Features`**.

4.8.4. Установка необходимого размера изображений, получаемых через функции _CheckResult() и _CheckResultFromList()

Для установки необходимого размера изображений, получаемых через функции CheckResult() и CheckPesultFromList() служит команда **`RPRM_Command_Options_Set_CheckResultHeight`**. С ее помощью задается желаемая **высота** получаемых изображений.

Эта команда игнорируется при вызове **`_CheckResult()`** в следующих случаях:

- получение `RPRM_ResultType_RawImage` без параметра `output` установленного в `ofrFormat_XML` и/или `ofrFormat_FileBuffer`;
- получение `RPRM_ResultType_EOSImage`.

4.8.5. Установка делителя частоты видеочипа

Изменение делителя частоты видеочипа может понадобиться только при работе со считывателями с видеочипом Micron. Для остальных считывателях эта команда игнорируется.

Для установки делителя частоты видеочипа служит команда **`RPRM_Command_Device_SetFrequencyDivider`**. С ее помощью задается желаемый делитель от `0` до `5`. Нулевое значение означает не использовать делитель частоты работы видеочипа.

ВНИМАНИЕ! После изменения значения необходимо выполнить перекалибровку прибора.

Повторный вызов этой команды с тем же значением делителя частоты не требует калибровки.

Необходимость изменения этого параметра очень редка и может возникнуть только в случае получения «битых» изображений или частого выхода из процесса сканирования с ошибкой `RPRM_Error_ScanAborted`. Рекомендуется повышать этот параметр на одно значение до тех пор, пока получение изображений не стабилизируется.

При работе в операционной системе Windows 7 стабильная работа возможна при делителе частоты, равном `1` и выше.

4.8.6. Режим отладки

Для режима отладки служит параметр команда **`RPRM_Command_Options_BuildExtLog`**.

Протокол работы SDK с подробной информацией о производимых действиях создается в отладочной консоли. Для просмотра отладочной консоли в состав SDK входит программа **DebugView (Dbgview.exe)**.
Дополнительная информация может быть добавлена в журнал путем установки значения 1 для ключа в реестре **HKEY_CURRENT_USER\SOFTWARE\Regula\Document Reader SDK\LogToDebugView**.

4.9. Работа с СИС

Для работы пользовательского приложения с СИС необходимо, чтобы она была предварительно установлена на используемый ПК.

При загрузке главной управляющей библиотеки происходит поиск динамической библиотеки взаимодействия с ИСС – **FDS_Dll.dll**, которая должна присутствовать в директории установки ИСС. По умолчанию поиск проводится по пути, зарегистрированному в процессе установки ИСС в ключе реестра **HKEY_LOCAL_MACHINE\SOFTWARE\Regula\FDS** в строковом элементе **Path**.

Для подключения ИСС к пользовательскому приложению необходимо выполнить следующие действия:

- в рабочем окне пользовательского приложения предусмотреть наличие дополнительного дочернего окна для отображения в нем информации из системы ИСС;
- подключить систему ИСС вызовом функции **_FDSUser_Connect()**, передав ей в качестве параметра дескриптор (**HWND**) выделенного под ИСС дочернего окна;
- установить рабочие размеры отображаемой в дочернем окне панели управления ИСС вызовом функции **_FDSUser_UpdateWindow()** и передачей ей в качестве параметров размеров панели.

Для отображения в окне ИСС информации об отсканированном документе необходимо вызвать функцию **_FDSUser_SelectionDocument()**, передав ей в качестве параметров трехбуквенный код выдачи документа государства, двухбуквенный код типа документа и тип освещения **eFDS_Light**.

Коды государства и типа документа могут быть получены, например, из результатов чтения МСЗ или зоны заполнения документа. При вызове **_FDSUser_SelectionDocument()** для них допустимо указывать нулевые значения. В этом случае будет открыто изображение, соответствующее переданному в третьем параметре функции типу освещения для текущего выбранного в панели ИСС документа.

Набор информационных панелей ИСС, отображаемых на экране, можно контролировать с помощью функции **_FDSUser_UpdatePanel**.

Отключение системы ИСС от пользовательского приложения осуществляется вызовом процедуры **_FDSUser_Disconnect().**

5. ПРОГРАММНЫЕ СРЕДСТВА SDK

5.1. Экспортируемые функции

Все экспортируемые функции объявлены со спецификатором `extern "C"`.

5.1.1. _Initialize()

Тип – `typedef long (*_InitializeFunc)(void *lpParams, HWND hParent)`
Символьное имя – **`_Initialize`**
Назначение: инициализация главной управляющей библиотеки
Параметры:
`lpParams` – зарезервирован
`hParent` – дескриптор окна пользовательского приложения, которое будет являться родительским для всех диалоговых окон, выводимых на экран при работе с главной управляющей библиотекой, или `NULL`

При вызове этой функции происходит инициализация необходимых для дальнейшей работы системных ресурсов, производится поиск и построение списка всех подключенных к ПК в данный момент времени считывателей документов.

Код возврата – одно из значений **`eRPRM_ErrorCodes`**.

Любое из возвращаемых значений, отличных от `RPRM_Error_NoError` (`RPRM_Error_AlreadyDone`), свидетельствует, что дальнейшая работа невозможна из-за возникновения критических ошибок. В этом случае необходимо вызвать функцию деинициализации **`_Free()`**, выгрузить `DLL` из памяти и завершить работу приложения или, предприняв необходимые меры по восстановлению работоспособности, повторить вызов функции инициализации.

5.1.2. _Free()

Тип – `typedef long (*_FreeFunc)()`
Символьное имя – **`_Free`**
Назначение: деинициализация главной управляющей библиотеки

При вызове этой функции происходит отключение активного считывателя документов и освобождение всех ресурсов главной управляющей библиотеки. Для возобновления работы необходимо вновь вызвать **`_Initialize()`**.

Код возврата – одно из значений **eRPRM_ErrorCodes**.

5.1.3. _SetCallbackFunc()

Тип – `typedef void (*_SetCallbackFuncFunc)(ResultReceivingFunc f1, NotifyFunc f2)`
Символьное имя – **_SetCallbackFunc**
Назначение: установка callback-функций для получения результатов и информационных сообщений

В качестве параметров данная функция принимает указатели на функции соответствующих типов, объявленных в пользовательском приложении, которые и будут вызываться по необходимости.

Существует два типа callback-функций пользовательского приложения, вызываемых из главной управляющей библиотеки SDK при возникновении определенных событий (**NotifyFunc**) и по готовности запрошенных результатов (**ResultReceivingFunc**).

Вызов функции **ResultReceivingFunc** всегда является синхронным (т. е. выполняется в контексте главного потока пользовательского приложения), в то время как вызов **NotifyFunc** может выполняться асинхронно, что требует (по необходимости) соответствующих модификаций пользовательского кода для обеспечения поддержки многопоточного режима.

5.1.4. _ExecuteCommand()

Тип – `typedef long (*_ExecuteCommandFunc)(long command, void *params, void *result)`
Символьное имя – **_ExecuteCommand**
Назначение: запрос на выполнение команды
Параметры:
`command` – код команды (одна из констант **eRPRM_Commands**)
`params` – входные параметры команды
`result` – указатель на контейнер для приема результатов

Код возврата – одно из значений **eRPRM_ErrorCodes**.

Типы данных входного параметра и приемника результатов определяются исходя из контекста каждой конкретной команды.

5.1.5. _ResultTypeAvailable()

Тип – `typedef uint32_t (*_ResultTypeAvailableFunc)(long type)`
Символьное имя – **`_ResultTypeAvailable`**
Назначение:
проверка наличия данных для результата заданного типа
Параметр
`type` – тип запрашиваемого результата (одно из значений **`eRPRM_ResultType`**)

Возвращаемое значение – общее количество экземпляров данных для результата указанного типа, сформированных в процессе проведения предшествующего цикла сканирования и обработки данных.

5.1.6. _CheckResult()

Тип – `typedef HANDLE (*_CheckResultFunc)(long type, long idx, long output, void *param)`
Символьное имя – **`_CheckResult`**
Назначение:
получение структуры-контейнера с результатами обработки данных заданного типа;
дополнительное преобразование данных к формату XML;
запись изображений и XML-данных в файл
Параметры:
`type` – тип запрашиваемого результата (один из типов представления данных **`eRPRM_ResultType`**)
`idx` – индекс объекта в общем списке доступных результатов запрашиваемого типа
`output` – дополнительное представление результата (`OR`-комбинация значений **`eRPRM_OutputFormat`**)
`param` – параметры в контексте значения дополнительного представления результата

Если возвращаемое значение `> 0`, то оно является дескриптором структуры, содержащей данные запрашиваемого типа (**`TResultContainer *`**, приведенный к типу `HANDLE`); если `< 0`, то одно из значений **`eRPRM_ResultStatus`**.

Для формирования образа графического файла изображения в параметре `type` необходимо указать **`ofrFormat_FileBuffer`**. Формат кодирования изображения будет соответствовать текущему установленному формату записи графических файлов. Результирующий буфер будет передан в пользовательское приложение в структуре-контейнере, содержащей данные типа **`RPRM_ResultType_FileImage`**, через вызов callback-функции **`ResultReceivingFunc`**. При этом поле `buffer` структуры

TResultContainer будет указывать на начало буфера с двоичным образом файла, а поле buf_length содержать длину этого буфера.

Для формирования дополнительного XML- или JSON-представления возвращаемой структуры данных (в том числе и для **RPRM_ResultType_FileImage**) в параметре type необходимо указать **ofrFormat_XML** или **ofrFormat_JSON** соответственно.

Для физической записи изображения или XML/JSON-представления результата на диск в параметре type необходимо указать **ofrTransport_File**. В этом случае параметр param должен содержать указатель на символьную строку в кодировке UTF8 с полным именем файла.

Если в параметре type дополнительно присутствует значение **ofrTransport_Clipboard**, изображение или XML/JSON-представление результата будет помещено в буфер обмена ОС: изображение – как объект CF_DIB, текст – как объект CF_TEXT (см. документацию по программированию в среде Windows). Эти данные будут доступны в пользовательском приложении после возврата из **_CheckResult()** через функцию Windows API **GetClipboardData()**.

5.1.7. _CheckResultFromList()

Тип – `typedef long (*_CheckResultFromListFunc)(HANDLE container, long output, void* param)`

Символьное имя – **_CheckResultFromList**

Назначение: доступ к отдельным полям структуры результата, являющейся списком

Параметры:

container – дескриптор результата, полученного через функцию **_CheckResult()**

output – дополнительное представление результата (одно из значений **eRPRM_OutputFormat**)

param – параметры для дополнительного представления результата

Если возвращаемое значение ≥ 0, то оно является цифровым кодом передаваемого поля (одно из значений **eVisualFieldType** или **eGraphicFieldType**); если < 0, то одно из значений **eRPRM_ResultStatus**.

При type=**ofrTransport_Clipboard** содержимое текущего поля (текстового или графического) будет помещено в буфер обмена Windows в формате CF_TEXT или CF_DIB и будет доступно после возвращения из **_CheckResultFromList()** через функцию WindowsAPI **GetClipboardData()**.

type=**ofrTransport_File** используется только для приема графических изображений из структуры-списка **TDocGraphicsInfo**). Изображение из текущего

поля будет записано в файл. В `param` необходимо передать указатель на текстовую строку (в кодировке UTF8), содержащую полное имя файла. Формат записи будет соответствовать текущему установленному формату записи графических файлов.

`type=`**`ofrFormat_FileBuffer`** используется только для приема графических изображений из структуры-списка **`TDocGraphicsInfo`**). Будет создан образ файла для изображения из текущего поля. В этом случае `param` должен содержать указатель **`TResultContainer *`** для сохранения результата. Указатель на массив данных результирующего образа файла будет записан в поле `buffer`, а его длина – в поле `buf_length`.

5.1.8. _AllocRawImageContainer()

Тип – `typedef long (*_AllocRawImageContainerFunc) (TRawImageContainer **cont, long w, long h, long bits_cnt, long resolution)`

Символьное имя – **`_AllocRawImageContainer`**

Назначение: выделение памяти под структуру данных **`TRawImageContainer`** для хранения изображения с заданными характеристиками

Параметры:

`cont` – адрес указателя в пользовательском приложении

`w` – ширина изображения в точках

`h` – высота изображения в точках

`bits_cnt` – число бит на цвет

`resolution` – разрешение изображения (число точек на метр)

Код возврата – одно из значений **`eRPRM_ErrorCodes`**.

Память, выделяемую этой функцией, необходимо освобождать через вызов функции **`_FreeRawImageContainer()`**.

5.1.9. _FreeRawImageContainer()

Тип – `typedef long (*_FreeRawImageContainerFunc) (TRawImageContainer *cont)`

Символьное имя – **`_FreeRawImageContainer`**

Назначение: освобождение области памяти, выделенной функцией `_AllocRawImageContainer()`

Параметр

`cont` – указатель на структуру

Код возврата – одно из значений **`eRPRM_ErrorCodes`**.

5.1.10. _FDSUser_Connect()

Тип – `typedef long (*_FDSUser_ConnectFunc)(HWND hWnd)`
Символьное имя – **`_FDSUser_Connect`**
Назначение: подключение СИС к пользовательскому приложению
Параметр
`hWnd` – дескриптор окна пользовательского приложения, в пределах которого будут отображаться информационные панели СИС

Код возврата – одно из значений **`eRPRM_ErrorCodes`**.

5.1.11. _FDSUser_Disconnect()

Тип – `typedef long (*_FDSUser_DisconnectFunc)()`
Символьное имя – **`_FDSUser_Disconnect`**
Назначение: отключение системы СИС от пользовательского приложения

Код возврата – одно из значений **`eRPRM_ErrorCodes`**.

5.1.12. _FDSUser_UpdateWindow()

Тип – `typedef long (*_FDSUser_UpdateWindowFunc)(long w, long h, long show)`
Символьное имя – **`_FDSUser_UpdateWindow`**
Назначение: установка новых общих размеров информационных панелей СИС и их видимости
Параметры:
`w` – новая ширина для информационных панелей СИС
`h` – новая суммарная высота для отображаемых информационных панелей СИС
`show` – общая видимость информационных панелей СИС («`true`» или «`false`»)

Код возврата – одно из значений **`eRPRM_ErrorCodes`**.

5.1.13. _FDSUser_UpdatePanel()

Тип – `typedef long(*_FDSUser_UpdatePanelFunc)(long panel, long position, long height)`
Символьное имя – **`_FDSUser_UpdatePanel`**
Назначение: индивидуальное управление расположением, размерами и видимостью информационных панелей СИС
Параметры:
`panel` – идентификатор панели (одно из значений **`eFDS_Panel`**)

`position` – расположение и видимость панели (одно из значений **`eFDS_Panel_Position`**)

`height` – новая высота панели

Код возврата – одно из значений **`eRPRM_ErrorCodes`**.

5.1.14. _FDSUser_SelectDocument()

Тип – `typedef long (*_FDSUser_SelectDocumentFunc)(char *CountryCode, char *DocCode, long Light)`

Символьное имя – **`_FDSUser_SelectDocument`**

Назначение: отображение в окне СИС информации о документе

`CountryCode` – трехбуквенный код государства выдачи документа (символы с третьего по пятый в первой строке МСЗ)

`DocCode` – двухбуквенный код типа документа (первые два символа в первой строке МСЗ)

`Light` – код типа освещения/уровня защиты, который необходимо активизировать в панели СИС (одно из значений **`eFDS_Light`**)

Код возврата – одно из значений **`eRPRM_ErrorCodes`**.

5.1.15. _LibraryVersion()

Тип – `typedef uint32_t (*_LibraryVersionFunc)()`

Символьное имя – **`_LibraryVersion`**

Назначение: возвращает версию главной управляющей библиотеки SDK:
`HIWORD()` – старший номер версии,
`LOWORD()` – младший номер версии (для версии 5.2 – 5 и 2 соответственно).

5.2. CALLBACK-ФУНКЦИИ

Это функции пользовательского приложения, которые вызываются в процессе функционирования главной управляющей библиотекой для сообщения о статусе выполнения команд или об изменении состояния устройства, а также для передачи результатов проведения цикла сканирования и обработки данных.

Типы callback-функций объявлены в `PasspR.h`:

```
typedef void (_stdcall *ResultReceivingFunc)(TResultContainer *result,
  uint32_t *PostAction, uint32_t *PostActionParameter);
typedef void (_stdcall *NotifyFunc)( intptr_t code, intptr_t value );
```

В пользовательском приложении они должны быть объявлены как:

```
void_stdcall MyResultReceivingFunc(TResultContainer *result,
  uint32_t *PostAction, uint32_t *PostActionParameter);
void_stdcall MyNotifyFunc(intptr_t code, intptr_t value);
```

и установлены следующим образом:

```
SetCallbackFunc(MyResultReceivingFunc, MyNotifyFunc);.
```

5.2.1. NotifyFunc()

Тип – `typedef void (_stdcall*NotifyFunc)( intptr_t code, intptr_t value)`

Назначение: получение сообщения о статусе выполнения команд или об изменении внутреннего состояния библиотеки или устройства

Параметры:

`code` – код сообщения (`eRPRM_NotificationCodes`)

`value` – значение (в контексте кода)

5.2.2. ResultReceivingFunc()

Тип – `typedef void (_stdcall*ResultReceivingFunc)(TResultContainer *result, uint32_t *PostAction, uint32_t *PostActionParameter)`

Назначение: получение результатов сканирования или обработки данных

`result` – указатель на структуру, содержащую данные результата сканирования или обработки данных

`PostAction` – одно из значений `eRPRM_PostCalbackAction`, заполняемое пользовательским приложением при возврате из callback-функции

`PostActionParameter` – параметр для `PostAction`

5.3. Структуры данных

5.3.1. TResultContainerList

Структура **`TResultContainerList`** используется для хранения и передачи списка изображений для обработки командой **`RPRM_Command_ProcessImagesList`**.

```
struct TResultContainerList
{
  uint32_t          Count;
  TResultContainer  *List;
};
```

Объявление: `PasspR.h`
Поля:

`Count` – количество элементов массива `List`
`List` – массив контейнеров для данных различного типа представления

Элементами массива `List` являются структуры типа **`TResultContainer`**, которые должны содержать изображения в формате **`RPRM_ResultType_RawImage`**.

5.3.2. TResultContainer

Структура **`TResultContainer`** используется для хранения и передачи в пользовательское приложение результатов, формируемых в ходе выполнения цикла сканирования и обработки данных.

```
struct TResultContainer
{
  uint32_t result_type;
  uint32_t light;
  uint32_t buf_length;
  void *buffer;
  uint32_t XML_length;
  BYTE *XML_buffer;
  uint32_t list_idx;
  uint32_t page_idx;
};
```

Объявление: `PasspR.h`
Поля:

`result_type` – тип результата, хранящегося в этом контейнере (один из идентификаторов **`eRPRM_ResultType`**)
`light` – код схемы освещения для данного результата (используется только для изображений)

buf_length	– размер структуры данных, на которую указывает buffer
buffer	– указатель на структуру с результатами чтения данных. Конкретный тип данных определяется значением поля result_type
XML_length	– размер массива XML_buffer, байт
XML_buffer	– текстовый массив, содержащий представление структуры с результатами чтения данных в формате XML
list_idx	– для внутреннего использования
page_idx	– индекс страницы (при работе с многостраничным документом)

Поля XML_length и XML_buffer инициализируются только при вызове функции **_CheckResult()** с указанием типа запрашиваемого результата **ofrFormat_XML**.

5.3.3. TRawImageContainer

Структура **TRawImageContainer** используется для хранения и передачи в пользовательское приложение графических изображений в формате Windows DIB без компрессии. Соответствующий тип результата – **RPRM_ResultType_RawImage** или **RPRM_ResultType_RawUncroppedImage**.

```
struct TRawImageContainer
{
  BITMAPINFO *bmi;
  BYTE       *bits;
};
```

Объявление: PasspR.h

Поля:

bmi	– заголовок Windows DIB с палитрой 256 цветов (если формат изображения предусматривает присутствие палитры)
bits	– массив точек изображения (с выравниванием на границу uint32_t)

Объем памяти, выделенной под bmi:

```
sizeof(BITMAPINFOHEADER) + sizeof(RGBQUAD)*256.
```

Объем памяти, выделенной под bits:

```
bmi.bmiHeader.biSizeImage.
```

5.3.4. TDocVisualExtendedInfo

Структура **TDocVisualExtendedInfo** служит для хранения *текстовых* результатов чтения МСЗ, полей заполнения документа и штрихкодов. Типы результата:**RPRM_ResultType_MRZ_OCR_Extended**,

RPRM_ResultType_Visual_OCR_Extended
и **RPRM_ResultType_BarCodes_TextData** соответственно.

```
struct TDocVisualExtendedInfo
{
  int                      nFields;
  TDocVisualExtendedField *pArrayFields;
};
```

Объявление: PasspR.h
Поля:

nFields – количество элементов массива pArrayFields
pArrayFields – массив структур, содержащих логически разделенные текстовые данные

5.3.5. TDocVisualExtendedField

Структура **TDocVisualExtendedField** является базовой структурой-контейнером для списка **TDocVisualExtendedInfo** и хранит информацию об одном текстовом поле данных.

```
struct TDocVisualExtendedField
{
  union
  {
    int FieldType;
    struct
    {
      WORD wFieldType;
      WORD wLCID;
    };
  };
  union
  {
    RECT FieldRect;
    struct
    {
      long RFID_OriginDG;
      long RFID_OriginDGTag;
      long RFID_OriginTagEntry;
      long RFID_OriginEntryView;
    };
  };
  char              FieldName[256];
  int               StringsCount;
  TStringResultSDK *StringsResult;
  intBuf_Length;
  char              *Buf_Text;
  char              *FieldMask;
  int               Validity;
  int               InComparison;
  uint32_t             Reserved2;
  uint32_t             Reserved3;
};
```

Объявление: PasspR.h
Поля:

FieldType – логический тип текстового поля (одно из значений **eVisualFieldType**)
wFieldType – логический тип текстового поля (одно из значений **eVisualFieldType**)
wLCID – идентификатор языка, служит для различия полей одного типа (например, 31 страница белорусского паспорта – белорусское и русское поле одного типа)
FieldRect – координаты области текстового поля на изображении документа (для результатов чтения полей заполнения документа **RPRM_ResultType_Visual_OCR_Extended**)
RFID_OriginDG – информационная группа данных – источник текстового поля (**eRFID_eRFDataGroupTypeTag**)
RFID_OriginDGTag – не используется (всегда содержит 0)
RFID_OriginTagEntry – индекс записи-источника текстового поля в информационной группе данных
RFID_OriginEntryView – не используется (всегда содержит 0)
FieldName – символьное имя текстового поля
StringsCount – количество элементов массива StringsResult
StringsResult – массив результатов распознавания отдельных строк многострочного текстового поля
Buf_Length – длина текстовой строки в Buf_Text
Buf_Text – строка с текстовыми данными поля в формате UTF8. Результаты чтения разных строк многострочного поля разделяются символом «^»
FieldMask – строка маски формата текстовых данных поля (для внутреннего использования)
Validity – для внутреннего использования SDK
InComparison – для внутреннего использования SDK
Reserved2 – не используется
Reserved3 – не используется

5.3.6. TStringResultSDK

Структура **TStringResultSDK** описывает результат распознавания отдельной строки многострочного текстового поля документа.

```
struct TStringResultSDK
{
  uint32_t          SymbolsCount;
  uint32_t          Reserved;
  TSymbolResult *StringResult;
};
```

Объявление: PasspR.h
Поля:
SymbolsCount – количество элементов массива StringResult
Reserved – для внутреннего использования
StringResult – массив результатов распознавания отдельных символов строки

5.3.7. TSymbolResult

Структура **TSymbolResult** описывает результат распознавания отдельного символа в строке поля документа.

```
struct TSymbolResult
{
   RECT             SymbolRect;
   uint32_t            CandidatesCount;
   TSymbolCandidate ListOfCandidates[4];
   uint32_t            Reserved;
};
```

Объявление: PasspR.h

Поля:

SymbolRect – границы области, занимаемой символом на изображении

CandidatesCount – количество значимых элементов массива ListOfCandidates

ListOfCandidates – массив символов-кандидатов. Отсортирован по убыванию вероятности распознавания (первый элемент имеет наивысшую вероятность)

Reserved – не используется

5.3.8. TSymbolCandidate

Структура **TSymbolCandidate** описывает результат распознавания отдельного символа в строке поля документа.

```
struct TSymbolCandidate
{
   uint32_t SymbolCode;
   uint32_t SymbolProbability;
   uint32_t Reserved;
};
```

Объявление: PasspR.h

Поля:

SymbolCode – ASCII-код символа

SymbolProbability – вероятность распознавания символа (0–100, %)

Reserved – не используется

5.3.9. TDocGraphicsInfo

Структура **TDocGraphicsInfo** служит для хранения *графических* результатов чтения полей заполнения документа и штрихкодов. Типы результата: **RPRM_ResultType_Graphics** и **RPRM_ResultType_BarCodes_ImageData** соответственно.

```
struct TDocGraphicsInfo
```

```
{
   int               nFields;
   TDocGraphicField *pArrayFields;
};
```

Объявление: PasspR.h

Поля:

nFields – количество элементов массива pArrayFields

pArrayFields – массив изображений

5.3.10. TDocGraphicField

Структура **TDocGraphicField** является базовой структурой-контейнером для списка **TDocGraphicsInfo** и содержит информацию об одном графическом поле.

```
struct TDocGraphicField
{
   int FieldType;
   union
   {
      RECT FieldRect;
      struct
      {
         long RFID_OriginDG;
         long RFID_OriginDGTag;
         long RFID_OriginTagEntry;
         long RFID_OriginEntryView;
      };
   };
   char                FieldName[256];
   TRawImageContainer image;
};
```

Объявление: PasspR.h

Поля:

FieldType – логический тип графического поля (одно из значений **eGraphicFieldType**)

FieldRect – координаты области поля на общем изображении (для результатов чтения полей заполнения документа **RPRM_ResultType_Graphics**)

RFID_OriginDG – информационная группа данных, которая является источником изображения (одно из значений **eRFID_eRFDataGroupTypeTag**)

OriginDGTag – индекс записи-источника изображения с биометрической информацией в информационной группе данных

RFID_OriginTagEntry – индекс образца в записи с биометрическими данными

RFID_OriginEntryView – индекс варианта образца биометрических данных

FieldName – символьное имя графического поля

image – изображение

5.3.11. TDocMRZTestQuality

Структура **TDocMRZTestQuality** служит для хранения информации о результатах проведения контроля качества заполнения МСЗ документа. Тип результата – **RPRM_ResultType_MRZ_TestQuality**.

```
struct TDocMRZTestQuality
{
   long            DOC_FORMAT;
   long            MRZ_FORMAT;
   long            TEXTUAL_FILLING;
   long            CHECK_SUMS;
   long            CONTRAST_PRINT;
   long            STAIN_MRZ;
   long            PRINT_POSITION;
   long            SYMBOLS_PARAM;
   long            StrCount;
   TStrEstimation  Strings[3];
};
```

Объявление: `PasspR.h`

Поля:

`DOC_FORMAT` – геометрический формат документа по ISO/IEC 7810 (одно из значений **`CDocFormat`**)

`MRZ_FORMAT` – формат МСЗ по Doc 9303 ICAO (одно из значений **`CDocFormat`**)

`TEXTUAL_FILLING` – оценка правильности текстового заполнения МСЗ

`CHECK_SUMS` – оценка правильности всех контрольных сумм МСЗ

`CONTRAST_PRINT` – оценка контраста печати МСЗ

`STAIN_MRZ` – оценка наличия пятен в МСЗ

`PRINT_POSITION` – оценка компоновки печати: расположение строк, расстояние между строками, выравнивание символов в строке

`SYMBOLS_PARAM` – оценка параметров символов: изменение контраста в символе, неровность края штриха, наличие пустот в символе, знаковый интервал в строке, выравнивание смежных символов, размер символов

`StrCount` – количество строк в МСЗ

`Strings` – массив результатов проверок для каждой строки

Значения оценок для этой и всех подчиненных структур данных (**`TStrEstimation`** и **`TSymbolEstimation`**) устанавливаются в соответствии с **`eCheckResult`**.

5.3.12. TStrEstimation

Структура **`TStrEstimation`** служит для хранения информации о результатах проверок для отдельной строки МСЗ.

```
struct TStrEstimation
{
   long              SymbolsCount;
   float             StringAngle;
   RECT              StringBorders;
   long              STRING_POSITION;
   TSingleRect       ErrorPOSITION;
   long              STRINGS_DISTANCE;
   float             SizeError_DISTANCE;
   long              STRINGS_INTERVAL;
   float             SizeError_INTERVAL;
   long              ALIGNMENT_SYMBOLS_IN_STRING;
   float             SizeError_ALIGNMENT;
   long              SYMBOLS_PARAM;
   long              STRING_FILLING;
```

```
   long               CHECK_SUMS;
   long               FieldCount;
   TTestTextField     Fields[12];
   TSymbolEstimation SymbolsEstimations[44];
};
```

Объявление: PasspR.h

Поля:

SymbolsCount	–	количество обработанных символов в строке
StringAngle	–	угол наклона строки в радианах («+» – по часовой стрелке, «–» – против часовой стрелки)
StringBorders	–	границы строки
STRING_POSITION	–	оценка позиции строки относительно краев документа
ErrorPOSITION	–	величина отклонения положения строки от допуска, мм
STRINGS_DISTANCE	–	оценка расстояния между строками
SizeError_DISTANCE	–	величина выхода за допуск расстояния между строками, мм
STRINGS_INTERVAL	–	оценка интервала между строками
SizeError_INTERVAL	–	величина выхода за допуск интервала между строками, мм
ALIGNMENT_SYMBOLS_IN_STRING	–	оценка выравнивания символов в строке
SizeError_ALIGNMENT	–	величина выхода за допуск выравнивания символов в строке, мм
SYMBOLS_PARAM	–	оценка параметров символов в строке
STRING_FILLING	–	оценка правильности текстового заполнения полей строки
CHECK_SUMS	–	оценка правильности всех контрольных сумм полей строки
FieldCount	–	количество проверенных текстовых полей в строке
Fields[12]	–	массив результатов проверок текстовых полей строки
SymbolsEstimations	–	подробная информация об оценке параметров символов в строке

5.3.13. TSingleRect

Структура **TSingleRect** служит для хранения информации о величине ошибки позиционирования строки МСЗ. Все значения – в мм.

```
struct TSingleRect
{
   float Left, Top, Right, Bottom;
};
```

Объявление: PasspR.h

Поля:

Left	–	величина отклонения положения строки МСЗ от допуска влево
Top	–	величина отклонения положения строки МСЗ от допуска вправо
Right	–	величина отклонения положения строки МСЗ от допуска вверх

`Bottom` – величина отклонения положения строки МСЗ от допуска вниз

5.3.14. TTestTextField

Структура **`TTestTextField`** служит для хранения информации о результатах проверок правильности заполнения отдельных текстовых полей МСЗ.

```
struct TTestTextField
{
  uint32_t TEST_RESULT;
  uint32_t FieldType;
  WORD  FieldPos;
  WORD  FieldLength;
  WORD  ValidCheckSum;
  WORD  reserved;
};
```

Объявление: `PasspR.h`

Поля:

`TEST_RESULT` – результат проверки поля (комбинация значений **`eTestTextField`**)

`FieldType` – логический тип поля (одно из значений **`eVisualFieldType`**)

`FieldPos` – индекс (начиная с `0`) первого символа поля в строке МСЗ

`FieldLength` – количество символов в поле

`ValidCheckSum` – рассчитанное по содержимому проверяемого поля значение контрольной суммы

5.3.15. TSymbolEstimation

Структура **`TSymbolEstimation`** служит для хранения информации о результатах оценки параметров печати одного символа МСЗ.

```
struct TSymbolEstimation
{
  char  CharSymbol;
  RECT  SymbolBounds;
  uint32_t SYMBOL_PARAM;
  uint32_t EMPTINESS;
  uint32_t EDGE;
  uint32_t STAIN;
  uint32_t CONTRAST_PRINT;
  uint32_t CONTRAST_SYMBOL;
  uint32_t ALIGNMENT_NEAREST_SYMBOLS
  float SizeErrorAlignWithPrev;
  float SizeErrorAlignWithNext;
  uint32_t SYMBOLS_INTERVAL;
  float SizeErrorIntervWithPrev;
  float SizeErrorIntervWithNext;
  uint32_t SYMBOL_SIZE;
  float SizeErrorSymbolHeight;
  float SizeErrorSymbolWidth;
};
```

Объявление: PasspR.h

Поля:

CharSymbol – результат распознавания символа
SymbolBounds – границы символа на изображении
SYMBOL_PARAM – обобщенная оценка параметров печати символа
EMPTINESS – оценка пустот в символе
EDGE – оценка неровности края штриха символа
STAIN – оценка наличия пятен в знакоместе символа
CONTRAST_PRINT – оценка контраста печати символа
CONTRAST_SYMBOL – оценка изменения контраста печати в символе
ALIGNMENT_NEAREST_SYMBOLS – оценка выравнивания по вертикали смежных символов
SizeErrorAlignWithPrev – величина ошибки выравнивания (выход за допуск, мм) по отношению к предыдущему символу
SizeErrorAlignWithNext – величина ошибки выравнивания (выход за допуск, мм) по отношению к следующему символу
SYMBOLS_INTERVAL – оценка знакового интервала в строке
SizeErrorIntervWithPrev – величина выхода значения интервала за допуск по отношению к предыдущему символу, мм
SizeErrorIntervWithNext – величина выхода значения интервала за допуск по отношению к следующему символу, мм
SYMBOL_SIZE – оценка размеров символа
SizeErrorSymbolHeight – величина выхода за допуск высоты символа, мм
SizeErrorSymbolWidth – величина выхода за допуск ширины символа, мм

5.3.16. TCommandsMRZTestQuality

Структура **TCommandsMRZTestQuality** служит для хранения параметров контроля качества заполнения МСЗ.

```
struct TDocMRZTestQuality
{
  uint32_t TEST_CLASS_QUALITY;
  uint32_t EXEC_SYMBOLS_PARAM;
  uint32_t EXEC_PRINT_POSITION;
  uint32_t EXEC_TEXTUAL_FILLING;
  uint32_t EXEC_CHECK_SUMS;
  uint32_t RESERVED1;
  uint32_t RESERVED2;
};
```

Объявление: PasspR.h

Поля:

TEST_CLASS_QUALITY – класс проверки качества (одно из значений **eMRZClassQuality**)

EXEC_SYMBOLS_PARAM – TRUE или FALSE. Объединяет следующие проверки: контраст печати, изменение контраста в символе, неровность края штриха символа, наличие пустот внутри символов, наличие пятен, расположение символов в строке, размеры символов

EXEC_PRINT_POSITION – TRUE или FALSE. Проверка компоновки печати (расположение строк)

EXEC_TEXTUAL_FILLING – TRUE или FALSE. Проверка правильности текстового заполнения МСЗ (выполнение данной проверки установлено по умолчанию)

EXEC_CHECK_SUMS – TRUE или FALSE. Проверка контрольных сумм МСЗ (выполнение данной проверки установлено по умолчанию)

5.3.17. TCandidatesListContainer

Структура **TCandidatesListContainer** служит для хранения и передачи в пользовательское приложение информации о документах-кандидатах при проведении процедуры распознавания типа. Тип результата – **RPRM_ResultType_Document TypesCandidates**.

```
struct TCandidatesListContainer
{
  uint32_t          RecResult;
  uint32_t          Count;
  TOneCandidate *Candidates;
};
```

Объявление: PasspR.h

Поля:

RecResult – результат определения типа документа (одно из значений **eRPRM_RCTP_Result_RecType**)

Count – количество элементов в массиве Candidates

Candidates – массив кандидатов распознавания

5.3.18. TOneCandidate

Структура **TOneCandidate** является базовой структурой-контейнером для списка **TCandidatesListContainer** и содержит информацию об одном документе-кандидате при определении типа документа.

```
struct TOneCandidate
{
  char                *DocumentName;
  int                 ID;
  double              P;
  WORD                Rotated180;
  WORD                RotationAngle;
  uint32_t               NecessaryLights;
  TRawImageContainer *preview;
  uint32_t               RFID_Presence;
  uint32_t               CheckAuthenticity;
  uint32_t               UVExp;
  WORD                OVIExp;
```

```
  uint32_tAuthenticityNecessaryLights;
  TFDSIDList          *extendedInfo;
};
```

Объявление: PasspR.h

Поля:

DocumentName	– наименование типа документа
ID	– числовой код типа документа
P	– мера вероятности правильного распознавания при анализе данного типа документа
Rotated180	– «true», если определено, что документ данного типа повернут на 180 градусов
NecessaryLights	– комбинация идентификаторов схем освещения (**eRPRM_Lights**), необходимых для проведения OCR для данного типа документа
preview	– уменьшенное изображение образца документа
RFID_Presence	– признак наличия в документе RFID-микросхемы (одно из значений **eRFID_Presence**)
CheckAuthenticity	– предусмотренный для данного типа документа набор опций контроля подлинности (комбинация значений **eRPRM_Authenticity**)
UVExp	– необходимое значение экспозиции видеокамеры при получении изображений документа данного типа для схемы УФ-освещения
OVIExp	– необходимое значение экспозиции видеокамеры при получении изображений документа данного типа для схемы аксиального освещения
AuthenticityNecessaryLights	– комбинация идентификаторов схем освещения (**eRPRM_Lights**), необходимых для проведения контроля подлинности данного типа документа
extendedInfo	– указатель на структуру TFDSIDList, содержащую дополнительную информацию о документе и его связи с ИСС

5.3.19. TFDSIDList

Структура **TFDSIDList** служит для хранения дополнительной информации о документе и его связи с документами в ИСС.

```
struct TFDSIDList
{
  char    ICAOCode[4];
  uint32_t     Count;
  uint32_t     *List;
  uint32_t     dType;
  uint32_t     dFormat;
  bool    dMRZ;
```

```
  char    *dDescription;
  char    *dYear;
  char    *dCountryName;
  char    *dStateCode;
  char    *dStateName;
};
```

Объявление: PasspR.h

Поля:

ICAOCode – код ICAO страны, выдавшей документ

Count – количество элементов в массиве List

List – массив идентификаторов документов в ИСС

dType – тип документа, одно из значений перечисления **diDocType**

dFormat – формат документа, одно из значений перечисления **CDocFormat**

dMRZ – флаг, обозначающий наличие МСЗ на документе

dDescription – текст описания документа

dYear – год издания документа

dCountryName – название страны, выдавшей документ

dStateCode – код штата страны, выдавшей документ

dStateName – название штата страны, выдавшей документ

5.3.20. TListDocsInfo

Структура **TListDocsInfo** служит для хранения и передачи в пользовательское приложение полного списка документов, хранящихся в текущей базе данных документов. Тип результата – **RPRM_ResultType_DocumentsInfoList**.

```
struct TListDocsInfo
{
  uint32_t          Count;
  TOCRDocInfo  *ArrayOfDocs;
};
```

Объявление: PasspR.h

Поля:

Count – количество элементов в массиве ArrayOfDocs

ArrayOfDocs – массив описаний документов

5.3.21. TOCRDocInfo

Структура **TOCRDocInfo** является базовой структурой-контейнером для списка **TListDocsInfo** и содержит информацию об одном документе.

```
struct TOCRDocInfo
{
  char  DocName[256];
  uint32_t DocID;
  char  DocTxtID[256];
  uint32_t DocFormat;
```

```
  uint32_t NecessaryLights;
  uint32_t nFields;
  uint32_t RFID_Presence;
  uint32_t reserved1;
  uint32_t reserved2;
  uint32_t reserved3;
};
```

Объявление: PasspR.h

Поля:

DocName – наименование документа

DocID – числовой код типа документа

DocTxtID – символьный код типа документа

NecessaryLight – комбинация идентификаторов схем освещения (**eRPRM_Lights**), необходимых для проведения OCR для данного типа документа

nFields – общее количество текстовых и графических полей, присутствующих в описании документа

RFID_Presence – признак наличия в документе RFID-микросхемы (одно из значений **eRFID_Presence**)

Reserved1 – не используется

Reserved2 – не используется

Reserved3 – не используется

5.3.22. TDocBarCodeInfo

Структура **TDocBarCodeInfo** служит для хранения и передачи в пользовательское приложение результатов поиска на сканируемой странице документа областей штрихкодов и их чтения в двоичном неформатированном виде. Тип результата – **RPRM_ResultType_BarCodes**.

```
struct TDocBarCodeInfo
{
  int               nFields;
  TDocBarCodeField *pArrayFields;
};
```

Объявление: PasspR.h

Поля:

nFields – количество элементов в массиве pArrayFields

pArrayFields – массив результатов чтения отдельных штрихкодов

5.3.23. TDocBarCodeField

Структура **TDocBarCodeField** является базовой структурой-контейнером для списка **TDocBarCodeInfo** и содержит данные чтения штрихкода.

```
struct TDocBarCodeField
{
   long                bcCodeResult;
   long                bcType_DETECT;
   RECT                bcROI_DETECT;
   float               bcAngle_DETECT;
   long                bcType_DECODE;
   long                bcCountModule;
   TIP_DECODE_MODULE   *bcDataModule;
   TIP_PDF417_INFO     bcPDF417INFO;
   long                bcReserved1;
   long                bcReserved2;
   long                bcReserved3;
};
```

Объявление: `PasspR.h`

Поля:

`bcCodeResult` – результат чтения штрихкода (одно из значений **`eBarCodeResultCodes`** либо другое значение, являющееся внутренним кодом ошибки SDK)

`bcType_DETECT` – тип найденного штрихкода (`1` – линейный, `2` – двухмерный: `PDF417`)

`bcROI_DETECT` – координаты области штрихкода на изображении

`bcAngle_DETECT` – угол разворота штрихкода, радианы

`bcType_DECODE` – тип декодированного штрихкода (одно из значений **`eBarCodeType`**)

`bcCountModule` – число прочитанных модулей штрихкода (количество элементов массива `bcDataModule`)

`bcDataModule` – результаты чтения данных из модулей штрихкода

`bcPDF417INFO` – информация о параметрах кода `PDF417` (только для `PDF417`)

`Reserved1` – не используется

`Reserved2` – не используется

`Reserved3` – не используется

5.3.24. TIP_DECODE_MODULE

Структура **`TIP_DECODE_MODULE`** используется для хранения результатов чтения одного модуля штрихкода.

```
struct TIP_DECODE_MODULE
{
   long mType;
   long mLength;
   BYTE *mData;
   long mReserved1;
   long mReserved2;
};
```

Объявление: PasspR.h
Поля:

mType – тип модуля (одно из значений **eBarCodeModuleType**)
mLength – длина массива данных модуля (количество значимых элементов массива mData)
mData – прочитанные данные модуля
mReserved1 – не используется
mReserved2 – не используется

5.3.25. TIP_PDF417_INFO

Структура **TIP_PDF417_INFO** используется для хранения дополнительной информации о параметрах штрихкода формата PDF417.

```
struct TIP_PDF417_INFO
{
   long  bcColumn;
   long  bcRow;
   long  bcErrorLevel;
   float minX;
   float minY;
   float Angle;
};
```

Объявление: PasspR.h
Поля:

bcColumn – число столбцов в штрихкоде
bcRow – число строк в штрихкоде
bcErrorLevel – уровень коррекции ошибок штрихкода
minX – ширина минимального элемента штрихкода на изображении, точки
minY – высота минимального элемента штрихкода на изображении, точки
Angle – угол поворота кода при декодировании, радианы

5.3.26. TListVerifiedFields

Структура **TListVerifiedFields** служит для хранения и передачи в пользовательское приложение результатов сравнения текстовых данных МСЗ, зоны заполнения документа, штрихкодов и данных из памяти RFID-микросхемы (при совместной работе с «*SDK для считывателей бесконтактных идентификационных микросхем*»). Тип результата – **RPRM_ResultType_OCRLexicalAnalyze**.

```
struct TListVerifiedFields
{
   uint32_t             Count;
   TVerifiedFieldMap *pFieldMaps;
};
```

Объявление: PasspR.h
Поля:

Count – количество элементов в массиве pFieldMaps
pFieldMaps – массив результатов анализа данных текстовых полей заполнения документа

5.3.27. TVerifiedFieldMap

Структура **TVerifiedFieldMap** является базовой структурой-контейнером для списка **TListVerifiedFields** и содержит информацию о результатах сравнения данных, полученных из разных источников, для одного и того же логического типа поля.

```
struct TVerifiedFieldMap
{
  WORD FieldType;
  WORD LCID;
  char *Field_MRZ;
  char *Field_RFID;
  char *Field_Visual;
  char *Field_Barcode;
  BYTE Matrix[10];
};
```

Объявление: PasspR.h
Поля:
FieldType – логический тип текстового поля (одно из значений **eVisualFieldType**)
LCID – идентификатор языка, служит для различия полей одного типа (например, 31 страница белорусского паспорта – белорусское и русское поле одного типа)
Field_MRZ – данные поля, полученные операцией чтения МСЗ
Field_RFID – данные поля, полученные операцией чтения памяти RFID-микросхемы
Field_Visual – данные поля, полученные операцией чтения текстовых полей заполнения документа
Field_Barcode – данные поля, полученные операцией чтения штрихкодов
Matrix – матрица сравнения результатов

Результаты чтения разных строк многострочного поля разделяются символом «^».
Элементы матрицы Matrix принимают одно из значений eRPRM_FieldVerificationResult. Элементы матрицы с индексами 0, 1, 2, 3 принимают одно из значений RCF_Disabled, RCF_Verified или RCF_Not_Verified, элементы с индексами 4, 5, 6, 7, 8 – одно из значений RCF_Disabled, RCF_Compare_True или RCF_Compare_False.
Элементы матрицы Matrix имеют следующее смысловое значение:

- элемент с индексом 0 – результат верификации данных из МСЗ;
- » 1 – результат верификации данных из RFID-микросхемы;
- » 2 – результат верификации данных из текстовых зон заполнения документа;
- » 3 – результат верификации данных из штрихкодов;
- » 4 – результат сравнения данных МСЗ и RFID-микросхемы;
- » 5 – результат сравнения данных МСЗ и текстовых зон заполнения документа;
- » 6 – результат сравнения данных МСЗ и штрихкодов;

- » 7 – результат сравнения данных текстовых зон заполнения документа и RFID-микросхемы;
- » 8 – результат сравнения данных текстовых зон заполнения документа и штрихкодов;
- » 9 – результат сравнения данных RFID-микросхемы и штрихкодов.

Примечание. Даты, содержащиеся в структуре **TVerifiedFieldMap**, могут являться автоматически преобразованными к единому формату представления дат, заданному в системе. Например, дата рождения, представленная в МСЗ строкой «681013» (с маской YYMMDD, где YY – год, MM – месяц, DD – день), а в поле заполнения документа строкой «13 OCT 1968», будет преобразована к формату DD.MM.YY («13.10.68») и уже именно в этом формате помещена в **TVerifiedFieldMap**.
Представление элементов **Matrix** в формате XML начинается с 1.

5.3.28. TAuthenticityCheckList

Структура **TAuthenticityCheckList** служит для хранения и передачи в пользовательское приложение результатов проверки подлинности документа по изображениям для различных схем освещения. Тип результата – **RPRM_ResultType_Authenticity**.

```
struct TAuthenticityCheckList
{
  int                       Count;
  TAuthenticityCheckResult **List;
};
```

Объявление: PasspR.h

Поля:

Count – количество элементов массива List

List – массив указателей на структуры, содержащие результаты проведения проверок элементов подлинности различных типов

5.3.29. TAuthenticityCheckResult

Структура **TAuthenticityCheckResult** служит для хранения результатов проверки элементов подлинности документа одного типа.

```
struct TAuthenticityCheckResult
{
  int  Type;
  int  Result;
  int  Count;
  void **List;
};
```

Объявление: PasspR.h

Поля:

Type – тип проверки (**eRPRM_Authenticity**)

Result – общий результат проверки (**eCheckResult**)

`Count` – число значимых элементов в `List`

`List` – массив указателей на структуры данных с результатами проверки данного типа

5.3.30. TFibersType

Структура **`TFibersType`** служит для хранения результатов проверки изображения для схемы УФ-освещения для одного типа люминесцирующих волокон (для проверки `RPRM_Authenticity_UV_Fibers`).

```
struct TFibersType
{
  union
  {
    WORD  ErrorCode;
    struct
    {
      WORD ElementResult;
      WORD ElementDiagnose;
    };
  };
  uint32_t RectCount;
  RECT  *RectArray;
  uint32_t *Width;
  uint32_t *Length;
  uint32_t *Area;
  BYTE  ColorValues[3];
  uint32_t ExpectedCount;
};
```

Объявление: `PasspR.h`

Поля:

`ErrorCode` – результат проверки (**`eCheckResult`**)

`ElementResult` – элемент, отвечающий за результат проверок (**`eCheckResult`**)

`ElementDiagnose` – элемент, при помощи которого проверяются ошибки `ElementResult` (**`eCheckDiagnose`**)

`RectCount` – число элементов `RectArray`, `Width`, `Length`, `Area`

`RectArray` – координаты найденных на изображении областей с волокнами данного типа

`Width` – ширина волокон из областей `RectArray`, точки

`Length` – длины волокон из областей `RectArray`, точки

`Area` – площади волокон из областей `RectArray`, точки

`ColorValues` – цвет волокон данного типа (`B`, `G`, `R`)

`ExpectedCount` – ожидаемое число волокон

5.3.31. TSecurityFeatureCheck

Структура **`TSecurityFeatureCheck`** служит для хранения результата проверки одного элемента изображения для одного из типов проверок:

- `RPRM_Authenticity_IR_B900` – контраст МСЗ по изображению для схемы ИК-освещения;

- RPRM_Authenticity_UV_Luminescence – контроль люминесценции материала документа для схемы УФ-освещения;
- RPRM_Authenticity_Axial_Protection – поиск областей с нарушением целостности ламинирующей пленки для схемы белого коаксиального освещения.

```
struct TSecurityFeatureCheck
{
  union
  {
    int Result;
    struct
    {
      WORD ElementResult;
      WORD ElementDiagnose;
    };
  };
  int         ElementType;
  RECT        ElementRect;
  uint32_t       Visibility;
  uint32_t       CriticalFlag;
  TAreaArray *AreaList;
  uint32_t       Reserved2;
};
```

Объявление: PasspR.h

Поля:

Result – результат проверки (**eCheckResult**)

ElementResult – элемент, отвечающий за результат проверок (**eCheckResult**)

ElementDiagnose – элемент, при помощи которого проверяются ошибки ElementResult (**eCheckDiagnose**)

ElementType – тип элемента (**eRPRM_SecurityFeatureType**)

ElementRect – область элемента

Visibility – флаг видимости элемента (**eIR_Visibility_Flag**)

CriticalFlag – флаг критичности проверки (**eSecurityCriticalFlag**)

5.3.32. TIdentResult

Структура **TIdentResult** служит для хранения результата проверки одного элемента изображения для одного из типов проверок:

- RPRM_Authenticity_IR_Visibility – контроль видимости/невидимости элементов бланка для схемы ИК-освещения;
- RPRM_Authenticity_Image_Pattern – для схемы УФ-освещения (контроль наличия люминесцирующего объекта).

```
struct TIdentResult
{
  union
  {
    uint32_t Result;
    struct
    {
      WORD ElementResult;
      WORD ElementDiagnose;
    };
  };
  uint32_t              LightIndex;
```

```
  RECT                Area;
  TRawImageContainer  Image;
  TRawImageContainer  EtalonImage;
  uint32_t               PercentValue;
  TAreaArray          *AreaList;
  uint32_t               ElementType;
};
```

Объявление: PasspR.h

Поля:

Result – результат проверки (**eCheckResult**)

ElementResult – элемент, отвечающий за результат проверок (**eCheckResult**)

ElementDiagnose – элемент, при помощи которого проверяются ошибки ElementResult (**eCheckDiagnose**)

LightIndex – идентификатор схемы освещения (**eRPRM_Lights**)

Area – область фрагмента на общем изображении документа

Image – найденный на общем изображении документа фрагмент

EtalonImage – ожидаемый фрагмент изображения

PercentValue – процент совпадения эталона и образца для **RPRM_Authenticity_Image_Pattern** или флаг видимости элемента (**eIR_Visibility_Flag**) для **RPRM_Authenticity_IR_Visibility**

ElementType – тип элемента (**eRPRM_SecurityFeatureType**)

5.3.33. TOCRSecurityTextResult

Структура **TOCRSecurityTextResult** служит для хранения результата OCR скрытого текста и его сравнения с заданным источником аналогичной текстовой информации для одного элемента изображения, например, проверка номера документа, напечатанного люминесцирующей в УФ-лучах краской (для проверки RPRM_Authenticity_OCRSecurityText).

```
struct TOCRSecurityTextResult
{
  union
  {
    uint32_t ResultCode;
    struct
    {
      WORD ElementResult;
      WORD ElementDiagnose;
    };
  };
  uint32_t  CriticalFlag;
  uint32_t  LightType;
  RECT   FieldRect;
  uint32_t  EtalonResultType;
  uint32_t  EtalonFieldType;
  uint32_t  EtalonLightType;
  RECT   EtalonFieldRect;
  char   SecurityTextResultOCR[256];
  char   EtalonResultOCR[256];
  uint32_t  Reserved1;
  uint32_t  Reserved2;
```

```
};
```

Объявление: PasspR.h

Поля:

ResultCode	– код результата проверки (**eCheckResult**)
ElementResult	– элемент, отвечающий за результат проверок (**eCheckResult**)
ElementDiagnose	– элемент, при помощи которого проверяются ошибки ElementResult (**eCheckDiagnose**)
CriticalFlag	– флаг критичности проверки (1 – текст присутствует всегда; 0 – текст может отсутствовать)
LightType	– тип освещения, при котором получено изображение (**eRPRM_Light**)
FieldRect	– координаты области элемента изображения
EtalonResultType	– тип результата для сравнения (допустимы значения из **eRPRM_ResultType**: **RPRM_ResultType_MRZ_OCR_Extended**, **RPRM_ResultType_Visual_OCR_Extended**, **RPRM_ResultType_BarCodes_TextData**)
EtalonFieldType	– тип поля для сравнения (**eRPRMFieldType**)
EtalonLightType	– тип освещения
EtalonFieldRect	– координаты области поля для сравнения
SecurityTextResultOCR	– результат распознавания
EtalonResultOCR	– результат распознавания, выбранный для сравнения в качестве эталонного

5.3.34. TPhotoIdentResult

Структура **TPhotoIdentResult** служит для хранения результатов визуализации одного внедренного скрытого изображения (для проверки RPRM_Authenticity_IPI).

```
struct TPhotoIdentResult
{
  union
  {
    int Result;
    struct
    {
      WORD ElementResult;
      WORD ElementDiagnose;
    };
  };
  int                   LightIndex;
  RECT                  Area;
  TRawImageContainer    SourceImage;
  TRawImageContainerList ResultImages;
  uint32_t                   FieldTypesCount;
  uint32_t*                  FieldTypesList;
  int                   Reserved1;
  int                   Reserved2;
  int                   Reserved3;
};
```

Объявление: PasspR.h

Поля:

`Result` – код результата проверки (**eMRZCheckResult**)
`ElementResult` – элемент, отвечающий за результат проверок (**eCheckResult**)
`ElementDiagnose` – элемент, при помощи которого проверяются ошибки `ElementResult` (**eCheckDiagnose**)
`LightIndex` – идентификатор схемы освещения (**eRPRM_Lights**)
`Area` – координаты области
`SourceImage` – исходное изображение
`ResultImages` – массив выходных изображений
`FieldTypesCount` – количество типов полей, текст которых закодирован в IPI. Зарезервировано для внутреннего использования
`FieldTypesList` – список констант типов полей, текст которых закодирован в IPI. Зарезервировано для внутреннего использования
`Reserved1` – не используется
`Reserved2` – не используется
`Reserved3` – не используется

5.3.35. TRegulaDeviceProperties

Структура **TRegulaDeviceProperties** служит для хранения информации о считывателе документов.

```
struct TRegulaDeviceProperties
{
   uint32_t DeviceID;
   uint32_t Lights;
   uint32_t SerialNumber;
   uint32_t Features;
   uint32_t DeviceCtrl;
   LPSTR DirectShowName;
   LPSTR DirectShowUID;
   LPSTR Name;
   uint32_t VideoModes;
   uint64_t LabelSerialNumber;
   LPSTR LabelSerialNumberStr;
   uint64_t CameraSerialNumber;
   GUID CameraGuid;
   uint32_t Capabilities;
   uint32_t Authenticity;
   uint32_t Database;
   time_t ValidUntil;
   bool WillConnect;
};
```

Объявление: `PasspR.h`

Поля:

`DeviceID` – идентификатор модели считывателя (одно из значений **eRPRM_DeviceTypes**)
`Lights` – комбинация идентификаторов всех доступных схем освещения считывателя (`OR`-комбинация значений **eRPRM_Lights**)
`SerialNumber` – серийный номер считывателя
`Features` – дополнительные свойства считывателя (`OR`-комбинация значений **eRPRM_DeviceAdditionalFeatures**)
`DeviceCtrl` – тип управления считывателем (одно из значений **eRPRM_DeviceControlTypes**)

DirectShowName – символьное имя считывателя (для устройств с управлением посредством DirectShow)
DirectShowUID – уникальный символьный идентификатор считывателя, формируемый ОС при подключении устройства в порт USB (для устройств с управлением посредством DirectShow)
Name – символьное имя считывателя (для устройств с прямым управлением)
VideoModes – поддерживаемые считывателем размеры сканируемых изображений (комбинация значений **eRPRM_VideoModes**)
LabelSerialNumber – серийный номер считывателя, нанесенный на его корпус
LabelSerialNumberStr – строковое представление серийного номера считывателя, нанесенного на его корпус
CameraSerialNumber – серийный номер камеры считывателя
CameraGuid – GUID камеры считывателя
Capabilities – набор доступных возможностей
Authenticity – набор доступных проверок подлинности
Database – тип базы данных
ValidUntil – дата окончания действия лицензии (timestamp)
WillConnect – индикация о возможности подключения считывателя после окончания действия лицензии.

5.3.36. TIndicationLED

Структура **TIndicationLED** служит для задания алгоритма поведения индикаторных светодиодов считывателя документов.

```
struct TIndicationLED
{
  WORD wColorLed;
  WORD wFreq;
};
```

Объявление: PasspR.h
Поля:
wColorLed – выбор индикаторного светодиода и цвета свечения. Младший байт (LOBYTE) должен содержать одно из значений **eLED_Color**, выбирающее цвет индикатора, старший байт (HIBYTE) – порядковый номер индикатора, начиная с индекса 0 (для считывателей с несколькими индикаторными светодиодами)
wFreq – выбор режима поведения индикатора. Младший и старший байты должны содержать значения в диапазоне 1–7. Значение младшего байта определяет длительность промежутка времени, в течение которого индикатор выключен, старший байт – длительность промежутка времени, в течение которого индикатор включен. Значение 1 – кратчайший интервал, 7 – наибольший. При

задании нулевого (0) значения для какого-либо параметра мигание светодиода прекращается

5.3.37. TPointArray

Структура **TPointArray** служит для хранения областей.

```
struct TPointArray
{
   int    PointCount;
   POINT  *PointsList;
};
```

Объявление: PasspR.h

Поля:

PointCount – количество точек в массиве PointsList

PointsList – массив координат точек

5.3.38. TAreaArray

Структура **TAreaArray** служит для хранения областей и контуров.

```
struct TAreaArray
{
   int         Count;
   RECT        *List;
   TPointArray *Points;
};
typedef TAreaArray* PAreaArray;
```

Объявление: PasspR.h

Поля:

Count – количество элементов в поле List и Points

List – массив областей

Points – массив точек

5.3.39. TIRVisibilityElement

Структура **TIRVisibilityElement** служит для хранения результата проверки IRVisibility.

```
struct TIRVisibilityElement
{
   RECT Field;
   long Visibility;
   long CheckResult;
   long ElementType;
   long Reserved;
};
```

Объявление: PasspR.h

Поля:

Field – координаты элемента

`Visibility` – флаг видимости/невидимости
`CheckResult` – результат проверки
`ElementType` – тип проверяемого элемента

5.3.40. TDwordArray

Структура **`TDwordArray`** служит для хранения массива значений типа `uint32_t`.

```
struct TDwordArray
{
  uint32_t Count;
  uint32_t *List;
};
```

Объявление: `PasspR.h`
Поля:
`Count` – количество элементов
`List` – массив элементов

5.3.41. TLexDateFormat

Структура **`TLexDateFormat`** служит для передачи формата даты командами **`RPRM_Command_Options_Get_LexDateFormat`** и **`RPRM_Command_Options_Set_LexDateFormat`**.

```
struct TLexDateFormat
{
  uint32_t format;
  char customFormatString[32];
};
```

Объявление: `PasspR.h`
Поля:
`format` – формат даты, одно из значений **`eLexDateFormat`**
`customFormatString` – специальная строка формата, обязательна для **`LDF_Custom`**

5.3.42. TBoundsResult

Структура **`TBoundsResult`** служит для хранения результатов поиска границ.

```
struct TBoundsResult
{
  int docFormat;
  int Width;
  int Height;
  POINT Center;
  float Angle;
  POINT LeftTop;
  POINT LeftBottom;
```

```
   POINT RightTop;
   POINT RightBottom;
   int Inverse;
   BYTE PerspectiveTr;
   BYTE ResultStatus;
   BYTE Reserve1;
   BYTE Reserve2;
   BYTE ObjArea;
   BYTE ObjIntAngleDev;
   int Dpi;
};
```

Объявление: `PasspR.h`

Поля:

`docFormat` – формат документа, одно из значений **`CDocFormat`**
`Width` – ширина документа
`Height` – высота документа
`Center` – координаты центра документа
`Angle` – угол поворота документа
`LeftTop` – координаты левого верхнего угла документа
`LeftBottom` – координаты левого нижнего угла документа
`RightTop` – координаты правого верхнего угла документа
`RightBottom` – координаты правого нижнего угла документа
`Inverse` – параметр для внутреннего использования
`PerspectiveTr` – параметр для внутреннего использования
`ResultStatus` – параметр для внутреннего использования
`Reserve1` – параметр для внутреннего использования
`Reserve2` – параметр для внутреннего использования
`ObjArea` – параметр для внутреннего использования
`ObjIntAngleDev` – параметр для внутреннего использования
`Dpi` – разрешение в точках на дюйм

5.3.43. TImageQualityCheck

Структура **`TImageQualityCheck`** служит для хранения результата проверки качества входного изображения.

```
struct TImageQualityCheck
{
   eImageQualityCheckType type;
   eCheckResult result;
   eRPRM_SecurityFeatureType featureType;
   TAreaArray *areas;
   float mean;
   float std_dev;
   int probability;
};
```

Объявление: `PasspR.h`

Поля:

type	–	тип проверки качества, одно из значений **eImageQualityCheckType**
result	–	результат проверки, одно из значений **eCheckResult**
featureType	–	результат проверки, одно из значений **eRPRM_SecurityFeatureType**
areas	–	аномальные области изображения
mean	–	среднее значение проверки
std_dev	–	значение отклонения
probability	–	вероятность проверки

5.3.44. TImageQualityCheckList

Структура **TImageQualityCheckList** служит для хранения списка результатов проверки качества входного изображения.

```
struct TImageQualityCheckList
{
  uint32_t Count;
  eCheckResult result;
  TImageQualityCheck ** List;
};
```

Объявление: PasspR.h

Поля:

Count	–	количество результатов в списке
result	–	общий результат проверки, одно из значений **eCheckResult**
List	–	массив указателей отдельных проверок

5.3.45. TVideodetectionNotification

Структура **TVideodetectionNotification** служит для хранения информации видеодетекции.

```
struct TVideodetectionNotification
{
    uint32_t sensorState;
    TRawImageContainer* image;
    TBoundsResult* mrzPosition;
};
```

Объявление: PasspR.h

Поля:

sensorState	–	состояние датчика
image	–	изображение видеодетекции
mrzPosition	–	местоположение МСЗ на изображении

5.3.46. TStatus

Структура **TStatus** служит для предоставления статуса проверки документов.

```
struct TStatus
{
eCheckResult overallStatus;
TDetailsOptical detailsOptical;
eCheckResult optical;
TDetailsRFID detailsRFID;
eCheckResult rfid;
eCheckResult portrait;
eCheckResult stopList;
};
```

Объявление: PasspR.h

Поля:

`overallStatus` – общий статус
`detailsOptical` – подробное описание оптической проверки
`optical` – оптическая проверка
`detailsRFID` – подробное описание RFID
`rfid –` – проверка RFID
`portrait` – проверка портрета
`stopList` – проверка по стоплисту

5.3.47. TDetailsRFID

Структура **TDetailsRFID** служит для предоставления подробностей проверки электронного документа.

```
struct TDetailsRFID
{
eCheckResult overallStatus;
eCheckResult PA;
eCheckResult AA;
eCheckResult CA;
eCheckResult TA;
eCheckResult BAC;
eCheckResult PACE;
};
```

Объявление: PasspR.h

Поля:

`overallStatus` – общий статус
`PA` – статус пассивной аутентификации
`AA` – статус активной аутентификации
`CA` – статус чиповой аутентификации
`TA` – статус терминальной аутентификации
`BAC` – Basics access control

PACE – Password Authenticated Connection Establishment

5.3.48. TDetailsOptical

Структура **TDetailsOptical** служит для предоставления подробностей оптической проверки.

```
struct TDetailsOptical
{
eCheckResult overallStatus;
eCheckResult mrz;
eCheckResult text;
eCheckResult docType;
eCheckResult security;
eCheckResult imageQA;
eCheckResult expiry;
uint32_t pagesCount;
};
```

Объявление: PasspR.h

Поля:

overallStatus	– общий статус
mrz	– статус МСЗ
text	– статус текста
docType	– статус типа документа
security	– статус проверки подлинности
imageQA	– статус качества входных изображений
expiry	– статус срока действия
pagesCount	– количество страниц

5.3.49. TTextResult

Структура **TTextResult** служит для хранения текстовых результатов.

```
struct TTextResult
{
uint32_t status;
uint32_t validityStatus;
uint32_t comparisonStatus;
char* dateFormat;
uint32_t fieldCount;
TTextField* fieldList;
uint32_t availableSourceCount;
TTextSource* availableSourceList;
};
```

Объявление: PasspR.h

Поля:

status	– общий статус
validityStatus	– статус достоверности
comparisonStatus	– статус сравнения

`dateFormat` – формат даты
`fieldCount` – количество полей
`fieldList` – список полей
`availableSourceCount` – количество источников
`availableSourceList` – список источников

5.3.50. TTextValidity

Структура **`TTextValidity`** служит для хранения результатов достоверности текстового поля указанного источника.

```
struct TTextValidity
{
char* source;
uint32_t status;
};
```

Объявление: PasspR.h
Поля:
`source` – имя источника
`status` – статус достоверности

5.3.51. TTextComparison

Структура **`TTextComparison`** служит для сравнения текстовых результатов.

```
struct TTextComparison
{
char* sourceLeft;
char* sourceRight;
uint32_t status;
};
```

Объявление: PasspR.h
Поля:
`sourceLeft, sourceRight` – имена источников
`status` – статус сравнения

5.3.52. TTextSource

Структура **`TTextSource`** служит для описания источника текстовых данных.

```
struct TTextSource
{
char* source;
uint32_t containerType;
uint32_t validityStatus;
};
```

Объявление: PasspR.h
Поля:
`source` – имя источника

containerType – тип контейнера
validityStatus - статус достоверности источника

5.3.53. TTextSymbol

Структура **TTextSymbol** служит для хранения результата чтения символа.

```
struct TTextSymbol
{
uint32_t code;
RECT rect;
uint32_t probability;
};
```

Объявление: PasspR.h
Поля:

code – код символа
rect – область символа
probability – вероятность распознавания

5.3.54. TTextFieldValue

Структура **TTextFieldValue** служит для хранения значений текстового поля.

```
struct TTextFieldValue
{
char* value;
char* originalValue;
uint32_t originalValidity;
char* source;
uint32_t containerType;
uint32_t pageIndex;
RECT fieldRect;
TRfidOrigin rfidOrigin;
uint32_t probability;
uint32_t originalSymbolsCount;
TTextSymbol* originalSymbols;
};
```

Объявление: PasspR.h
Поля:

value – значение поля в текущем формате предоставления данных
originalValue – оригинальное значение поля
originalValidity – достоверность оригинала
source – источник
containerType – тип контейнера
pageIndex – индекс страницы
fieldRect – область поля
rfidOrigin – источник поля из электронного документа
probability – вероятность распознавания поля
originalSymbolsCount – количество оригинальных символов
originalSymbols – оригинальные символы

5.3.55. TTextField

Структура **TTextField** служит для хранения текстового поля.

```
struct TTextField
{
uint16_t fieldType;
char* fieldName;
uint16_t lcid;
char* lcidName;
uint32_t status;
char* value;
uint32_t valueCount;
TTextFieldValue* valueList;
uint32_t validityStatus;
uint32_t validityCount;
TTextValidity* validityList;
uint32_t comparisonStatus;
uint32_t comparisonCount;
TTextComparison* comparisonList;
};
```

Объявление: PasspR.h

Поля:

fieldType	– тип поля
fieldName	– имя поля
lcid	– код LCID
lcidName	– имя LCID
status	– статус поля
value	– значение поля
valueCount	– количество значений
valueList	– список значений
validityStatus	– статус достоверности
validityCount	– количество результатов достоверности
validityList	– список результатов достоверности
comparisonStatus	– статус сравнения
comparisonCount	– количество результатов сравнения
comparisonList	– список результатов сравнения

5.3.56. TImagesResult

Структура **TImagesResult** служит для представления всех графических результатов в одном контейнере.

```
struct TImagesResult
{
uint32_t fieldCount;
TImageField* fieldList;
uint32_t availableSourceCount;
TImageSource* availableSourceList;
};
```

Объявление: PasspR.h
Поля:

`fieldCount` – количество полей
`fieldList` – список полей
`availableSourceCount` – количество доступных источников
`availableSourceList` – список доступных источников

5.3.57. TImageSource

Структура **`TImageSource`** служит для описания источника графической информации.

```
struct TImageSource
{
char* source;
uint32_t containerType;
};
```

Объявление: PasspR.h
Поля:

`source` – источник
`containerType` – тип контейнера

5.3.58. TImageField

Структура **`TImageField`** служит для предоставления одного изображения или графического поля.

```
struct TImageField
{
uint32_t fieldType;
char* fieldName;
uint32_t valueCount;
TImageFieldValue* valueList;
};
```

Объявление: PasspR.h
Поля:

`fieldType` – тип поля
`fieldname` – наименование поля
`valueCount` – количество значений
`valueList` – список значений

5.3.59. TImageFieldValue

Структура **`TImageFieldValue`** служит для представления изображения.

```
struct TImageFieldValue
{
char* value;
```

```
char* originalValue;
char* source;
uint32_t containerType;
uint32_t pageIndex;
uint32_t lightIndex;
RECT fieldRect;
TRfidOrigin rfidOrigin;
uint32_t originalPageIndex;
};
```

Объявление: PasspR.h

Поля:

`value` – файл изображения в Base64

`originalValue` – файл оригинального изображения в Base64

`source` – источник

`containerType` – тип контейнера

`pageIndex` – индекс страницы

`lightIndex` – идентификатор схемы освещения

`fieldRect` – область поля

`rfidOrigin` – источник поля из электронного документа

`originalPageIndex` – индекс страницы входного изображения, на котором был найден результат

5.3.60. TRfidOrigin

Структура **`TRfidOrigin`** служит для описания источника данных электронного документа.

```
struct TRfidOrigin
{
int32_t dg;
int32_t dgTag;
int32_t tagEntry;
int32_t entryView;
};
```

Объявление: PasspR.h

Поля:

`Dg` – data group

`dgTag` – DG tag

`tagEntry` – tag entry

`entryView` – entry view

5.4. Перечисления (наборы констант)

5.4.1. eRPRM_ResultType

Перечисление **eRPRM_ResultType** содержит набор констант, которые определяют тип данных, хранящихся в структуре-контейнере **TResultContainer**.

```
enum eRPRM_ResultType
{
   RPRM_ResultType_Empty                              = 0,
   RPRM_ResultType_RawImage                           = 1,
   RPRM_ResultType_FileImage                          = 2,
   RPRM_ResultType_MRZ_OCR_Extended                   = 3,
   RPRM_ResultType_BarCodes                           = 5,
   RPRM_ResultType_Graphics                           = 6,
   RPRM_ResultType_MRZ_TestQuality                    = 7,
   RPRM_ResultType_DocumentTypesCandidates            = 8,
   RPRM_ResultType_ChosenDocumentTypeCandidate        = 9,
   RPRM_ResultType_DocumentsInfoList                  = 10,
   RPRM_ResultType_OCRLexicalAnalyze                  = 15,
   RPRM_ResultType_RawUncroppedImage                  = 16,
   RPRM_ResultType_Visual_OCR_Extended                = 17,
   RPRM_ResultType_BarCodes_TextData                  = 18,
   RPRM_ResultType_BarCodes_ImageData                 = 19,
   RPRM_ResultType_Authenticity                       = 20,
   RPRM_ResultType_ExpertAnalyze                      = 21,
   RPRM_ResultType_OCRLexicalAnalyzeEx                = 22,
   RPRM_ResultType_EOSImage                           = 23,
   RPRM_ResultType_Bayer                              = 24,
   RPRM_ResultType_MagneticStripe                     = 25,
   RPRM_ResultType_MagneticStripe_TextData            = 26,
   RPRM_ResultType_FieldFileImage                     = 27,
   RPRM_ResultType_DatabaseCheck                      = 28,
   RPRM_ResultType_FingerprintTemplateISO             = 29,
   RPRM_ResultType_InputImageQuality                  = 30,
   RPRM_ResultType_DeviceInfo                         = 31,
   RPRM_ResultType_LivePortrait                       = 32,
   RPRM_ResultType_Status                             = 33,
   RPRM_ResultType_Portrait_Comparison                = 34,
   RPRM_ResultType_ExtPortrait                        = 35,
   RPRM_ResultType_Text                               = 36,
   RPRM_ResultType_Images                             = 37,
   RPRM_ResultType_FingerPrints                       = 38,
   RPRM_ResultType_BSI_XML_v2                         = 73,
   RPRM_ResultType_DocumentPosition                   = 85,
   RPRM_ResultType_BSI_XML                            = 92,
   RPRM_ResultType_Custom                             = 100,
};
```

Эти значения записываются в поле `result_type` структуры **TResultContainer** и однозначно определяют тип структуры данных, указателем на которую является значение поля `buffer`:

- **RPRM_ResultType_Empty** – результат отсутствует, пустой контейнер;
- **RPRM_ResultType_RawImage** – результат представлен в виде структуры `TRawImageContainer`;
- **RPRM_ResultType_FileImage** – двоичного массива длиной `buf_length`, в котором содержится образ графического файла изображения;
- **RPRM_ResultType_MRZ_OCR_Extended** – структуры `TDocVisualExtendedInfo`;
- **RPRM_ResultType_BarCodes** – структуры `TDocBarCodeInfo`;
- **RPRM_ResultType_Graphics** – структуры `TDocGraphicsInfo`;
- **RPRM_ResultType_MRZ_TestQuality** – структуры `TDocMRZTestQuality`;
- **RPRM_ResultType_DocumentTypesCandidates** – структуры `TCandidatesListContainer`;
- **RPRM_ResultType_ChosenDocumentTypeCandidate** – структуры `TOneCandidate`;
- **RPRM_ResultType_DocumentsInfoList** – структуры `TListDocsInfo`. НЕ ИСПОЛЬЗУЕТСЯ;
- **RPRM_ResultType_OCRLexicalAnalyze** – структуры `TListVerifiedFields`;
- **RPRM_ResultType_RawUncroppedImage** – структуры `TRawImageContainer`. НЕ ИСПОЛЬЗУЕТСЯ;
- **RPRM_ResultType_Visual_OCR_Extended** – структуры `TDocVisualExtendedInfo`;
- **RPRM_ResultType_BarCodes_TextData** – структуры `TDocVisualExtendedInfo`;
- **RPRM_ResultType_BarCodes_ImageData** – структуры `TDocGraphicsInfo`;
- **RPRM_ResultType_Authenticity** – структуры `TAuthenticityCheckList`;
- **RPRM_ResultType_ExpertAnalyze** – НЕ ИСПОЛЬЗУЕТСЯ;
- **RPRM_ResultType_OCRLexicalAnalyzeEx** – НЕ ИСПОЛЬЗУЕТСЯ;
- **RPRM_ResultType_EOSImage** – структуры `TRawImageContainer`;
- **RPRM_ResultType_Bayer** – структуры `TRawImageContainer`;
- **RPRM_ResultType_MagneticStripe** – двоичного массива длиной `buf_length`, в котором содержится массив прочитанных из магнитной полосы данных;
- **RPRM_ResultType_MagneticStripe_TextData** – структуры `TDocVisualExtendedInfo`;

- **RPRM_ResultType_FieldFileImage** – двоичного массива длиной `buf_length`, в котором содержится образ графического файла изображения графического поля;
- **RPRM_ResultType_DatabaseCheck** – структуры TDatabaseCheck;
- **RPRM_ResultType_FingerprintTemplateISO** - двоичного массива длиной `buf_length`, в котором содержится шаблон отпечатка пальца в формате ISO;
- **RPRM_ResultType_DeviceInfo** – структуры `TRegulaDeviceProperties`, содержащей описание лицензии подключенного считывателя;
- **RPRM_ResultType_LivePortrait** – структуры `TDocGraphicsInfo`, содержащей изображение портрета владельца документа с внешней камеры;
- **RPRM_ResultType_Status** - структуры `TStatus`;
- **RPRM_ResultType_Portrait_Comparison** – структуры `TAuthenticityCheckList`, содержащей результаты сравнения портрета из пришедших извне изображений и портрета на документе. Результаты сравнения с `RPRM_ResultType_ExtPortrait` и `RPRM_ResultType_LivePortrait` попадают в контейнер с этим типом;
- **RPRM_ResultType_ExtPortrait** – структуры `TDocGraphicsInfo`, содержащей изображение портрета владельца документа из внешнего источника;
- **RPRM_ResultType_Text –** структуры `TTextResult`;
- **RPRM_ResultType_Images** – структуры `TImagesResult`;
- **RPRM_ResultType_FingerPrints** структуры `TDocGraphicsInfo`;
- **RPRM_ResultType_DocumentPosition** – результат поиска границ документа в виде структуры `TBoundsResult`;
- **RPRM_ResultType_InputImageQuality** – результат проверки качества входного изображения в виде структуры `TImageQualityCheckList`;
- **RPRM_ResultType_BSI_XML** – Result in XML format according to BSI TR-03135 v1;
- **RPRM_ResultType_BSI_XML_v2** – Result in XML format according to BSI TR-03135 v2;
- **RPRM_ResultType_Custom** – НЕ ИСПОЛЬЗУЕТСЯ.

5.4.2. eRPRM_DeviceAdditionalFeatures

Перечисление **eRPRM_DeviceAdditionalFeatures** содержит набор констант, которые определяют дополнительные свойства считывателя документов.

```
enum eRPRM_DeviceAdditionalFeatures
{
  RPRM_DeviceAdditionalFeature_None                = 0x00000000,
  RPRM_DeviceAdditionalFeature_Accumulator         = 0x00000001,
  RPRM_DeviceAdditionalFeature_Indicators_Triple = 0x00000002,
  RPRM_DeviceAdditionalFeature_VideoDetection      = 0x00000004,
  RPRM_DeviceAdditionalFeature_IRFilter            = 0x00000008,
  RPRM_DeviceAdditionalFeature_Indicators_Single   = 0x00000010,
  RPRM_DeviceAdditionalFeature_Indicators_Double   = 0x00000020,
  RPRM_DeviceAdditionalFeature_Indicators_Button   = 0x00000040,
  RPRM_DeviceAdditionalFeature_Indicators_Four     = 0x00000080,
  RPRM_DeviceAdditionalFeature_2SidedWhite         = 0x00000100,
  RPRM_DeviceAdditionalFeature_2SidedIR            = 0x00000200,
  RPRM_DeviceAdditionalFeature_2SidedUV            = 0x00000400,
  RPRM_DeviceAdditionalFeature_MagneticStripe      = 0x00001000,
  RPRM_DeviceAdditionalFeature_JPEGCompression     = 0x00002000,
  RPRM_DeviceAdditionalFeature_IntegratedDisplay   = 0x00004000,
  RPRM_DeviceAdditionalFeature_KeyboardLight       = 0x00008000,
  RPRM_DeviceAdditionalFeature_ExternalLight       = 0x00010000,
  RPRM_DeviceAdditionalFeature_RESERVED            = 0x00020000,
  RPRM_DeviceAdditionalFeature_DocumentSensor      = 0x00040000,
  RPRM_DeviceAdditionalFeature_DocSizeMode         = 0x00080000,
  RPRM_DeviceAdditionalFeature_LiveView            = 0x00100000,
};
```

Значение констант:

- **RPRM_DeviceAdditionalFeature_Accumulator** Считыватель документов оборудован аккумулятором;

- **RPRM_DeviceAdditionalFeature_Indicators_Triple** Считыватель документов имеет три диода индикации;

- **RPRM_DeviceAdditionalFeature_VideoDetection** Считыватель документов может работать в режиме полностраничной видеодетекции;

- **RPRM_DeviceAdditionalFeature_IRFilter** Считыватель документов имеет ИК-фильтр;

- **RPRM_DeviceAdditionalFeature_Indicators_Single** Считыватель документов имеет один диод индикации;

- **RPRM_DeviceAdditionalFeature_Indicators_Double** Считыватель документов имеет два диода индикации;

- **RPRM_DeviceAdditionalFeature_Indicators_Button** Считыватель документов имеет интегрированный в кнопку диод индикации;

- **RPRM_DeviceAdditionalFeature_Indicators_Four** Считыватель документов имеет четыре диода индикации;

- **RPRM_DeviceAdditionalFeature_Indicators_2SidedWhite** Считыватель документов способен снимать два белых изображения обеих сторон документа одновременно;

- **RPRM_DeviceAdditionalFeature_Indicators_2SidedIR** Считыватель документов способен снимать два ИК-изображения обеих сторон документа одновременно;

- **RPRM_DeviceAdditionalFeature_Indicators_2SidedUV** Считыватель документов способен снимать два УФ-изображения обеих сторон документа одновременно;

- **RPRM_DeviceAdditionalFeature_MagneticStripe** Считыватель документов способен считывать магнитную полосу;

- **RPRM_DeviceAdditionalFeature_JPEGCompression** Считыватель документов способен получать изображения в JPEG вместо RAW;

- **RPRM_DeviceAdditionalFeature_IntegratedDisplay** Считыватель документов имеет встроенный экран;

- **RPRM_DeviceAdditionalFeature_KeyboardLight** Считыватель документов имеет встроенную подсветку клавиатуры;

- **RPRM_DeviceAdditionalFeature_ExternalLight** Считыватель документов имеет внешний источник света;

- **RPRM_DeviceAdditionalFeature_RESERVED** Зарезервировано для внутреннего использования;

- **RPRM_DeviceAdditionalFeature_DocumentSensor** Считыватель документов имеет датчик документа (для считывателей, обычно работающих в режиме видеодетекции);

- **RPRM_DeviceAdditionalFeature_DocSizeMode** Для внутреннего использования.

- • **RPRM_DeviceAdditionalFeature_LiveView** Считыватель документов имеет доступность функции «Live View».

5.4.3. eRPRM_DeviceControlTypes

Перечисление **eRPRM_DeviceControlTypes** содержит набор констант, которые определяют тип управления считывателем документов.

```
enum eRPRM_DeviceControlTypes
{
  RPRM_DeviceControlType_DirectShow = 1,
  RPRM_DeviceControlType_DirectIO   = 2,
  RPRM_DeviceControlType_Virtual    = 3
};
```

Значение констант:

- **RPRM_DeviceControlType_DirectShow** Считыватель документов управляется средствами `DirectShow`

- **RPRM_DeviceControlType_DirectIO**

Считыватель документов обладает прямым управлением

- **RPRM_DeviceControlType_Virtual** Виртуальный считыватель документов

5.4.4. eRPRM_DeviceTypes

Перечисление **eRPRM_DeviceTypes** содержит набор уникальных идентификаторов, определяющих тип модели считывателя документов.

```
enum eRPRM_DeviceTypes
{
  RPRM_DeviceType_Unknown    =  0x00000000, // Unknown device
  RPRM_DeviceType_Virtual    =  0xFFFFFFFF, // Virtual Device
                                            // (Security Key)

  // 83x3
  RPRM_DeviceType_FX_8313_115  =  0x08313115, // OV 1.3
  RPRM_DeviceType_FX_8333_115  =  0x08333115, // OV 3.0
  RPRM_DeviceType_FX_8353_115  =  0x08353115, // OV 5.1
  RPRM_DeviceType_FX_8383_115  =  0x08383115, // Micron 3.1
  RPRM_DeviceType_FX_8883_115  =  0x08883115, // Micron 9.0
  RPRM_DeviceType_FX_8853_115  =  0x08853115, // Micron 5.0


  // 83x3M
  RPRM_DeviceType_FX_8303_115 =  0x08323115, // Micron 3.1

  // 83x4 (83x3 + Bottom light table)
  RPRM_DeviceType_FX_8314_115  =  0x08314115, // OV 1.3
  RPRM_DeviceType_FX_8334_115  =  0x08334115, // OV 3.0
  RPRM_DeviceType_FX_8354_115  =  0x08354115, // OV 3.0
  RPRM_DeviceType_FX_8384_115  =  0x08384115, // Micron 3.1
  RPRM_DeviceType_FX_8884_115  =  0x08884115, // Micron 9.0
  RPRM_DeviceType_FX_8854_115  =  0x08854115, // Micron 5.0

  // 7007
  RPRM_DeviceType_FX_7107_115  =  0x07107115, // OV 1.3
  RPRM_DeviceType_FX_7117_115  =  0x07117115, // OV 1.3
                                            // Modification 2
  RPRM_DeviceType_FX_7307_115  =  0x07307115, // OV 3.0
  RPRM_DeviceType_FX_7317_115  =  0x07317115, // OV 3.1
                                            // Modification 2
  RPRM_DeviceType_FX_7507_115  =  0x07507115, // OV 5.1
```

```
RPRM_DeviceType_FX_7517_115  =  0x07517115, // OV 5.1
                                            // Modification 2
RPRM_DeviceType_FX_7387_115  =  0x07387115, // Micron 3.1
RPRM_DeviceType_FX_7397_115  =  0x07397115, // Micron 3.1
                                            // Modification 2
RPRM_DeviceType_FX_7887_115  =  0x07887115, // Micron 9.0
RPRM_DeviceType_FX_7857_115  =  0x07857115, // Micron 5.0

// 8307
RPRM_DeviceType_FX_8307      =  0x08307000, // Micron 3.1 Mp
                                            // (MRZ-only reader)

// 7024
RPRM_DeviceType_FX_7104_115  =  0x07104115, // OV 1.3
RPRM_DeviceType_FX_7304_115  =  0x07304115, // OV 3.0
RPRM_DeviceType_FX_7504_115  =  0x07504115, // OV 5.0
RPRM_DeviceType_FX_7384_115  =  0x07384115, // Micron 3.1
RPRM_DeviceType_FX_7884_115  =  0x07884115, // Micron 9.0
RPRM_DeviceType_FX_7854_115  =  0x07854115, // Micron 5.0
RPRM_DeviceType_FX_78A4_115  =  0x078A4115, // Micron 10.0
RPRM_DeviceType_FX_78E4_115  =  0x078E4115, // Micron 14.0

// 7024 Lite functionality
RPRM_DeviceType_FX_7104_Lite =  0x07104333, // OV 1.3
RPRM_DeviceType_FX_7304_Lite =  0x07304333, // OV 3.0
RPRM_DeviceType_FX_7504_Lite =  0x07504333, // OV 5.1
RPRM_DeviceType_FX_7384_Lite =  0x07384333, // Micron 3.1
RPRM_DeviceType_FX_7884_Lite =  0x07884333, // Micron 9.0
RPRM_DeviceType_FX_7854_Lite =  0x07854333, // Micron 5.0
RPRM_DeviceType_FX_78A4_Lite =  0x078A4333, // Micron 10.0
RPRM_DeviceType_FX_78E4_Lite =  0x078E4333, // Micron 14.0

// 70x3
RPRM_DeviceType_FX_7103_115  =  0x07103115, // OV 1.3
RPRM_DeviceType_FX_7303_115  =  0x07303115, // OV 3.0
RPRM_DeviceType_FX_7503_115  =  0x07503115, // OV 5.1
RPRM_DeviceType_FX_7383_115  =  0x07383115, // Micron 3.1
RPRM_DeviceType_FX_7883_115  =  0x07883115, // Micron 9.0
RPRM_DeviceType_FX_7853_115  =  0x07853115, // Micron 5.0

// 4820
RPRM_DeviceType_FX_4821      =  0x04821115, // OV 1.3
RPRM_DeviceType_FX_4823      =  0x04823115, // OV 3.0
RPRM_DeviceType_FX_4825      =  0x04825115, // OV 5.1
RPRM_DeviceType_FX_4822      =  0x04822115, // Micron 3.1
RPRM_DeviceType_FX_4828      =  0x04828115, // Micron 9.0
RPRM_DeviceType_FX_4858      =  0x04858115, // Micron 5.0

// 7008
RPRM_DeviceType_FX_7038      =  0x07038115, //Micron 3.1
RPRM_DeviceType_FX_7038_VB   =  0x17038115, //Micron 3.1
RPRM_DeviceType_FX_7058      =  0x07058115, //Micron 5.0

// 70x8M
RPRM_DeviceType_FX_7058M     =  0x07058110, //OV 5 Mp
RPRM_DeviceType_FX_7058M_VB  =  0x17058110, //OV 5 Mp

// 70x4M small reader
RPRM_DeviceType_FX_73x4M     =  0x07364115, //Micron 3.1
RPRM_DeviceType_FX_75x4M     =  0x07564115, //Micron 5.0
RPRM_DeviceType_FX_71x4M     =  0x07A64115, //Micron 10.0
RPRM_DeviceType_FX_76x4M     =  0x07664115, //OV 5Mp
RPRM_DeviceType_FX_78x4M     =  0x07864115, //Micron 18Mp

// 7308
RPRM_DeviceType_FX_7338      =  0x07338115, //Micron 3.1

// 72x3  ID1 reader with 2 cameras
RPRM_DeviceType_FX_7253      =  0x07253115, //OV 5.0 Mp
                             // with 2 cameras

RPRM_DeviceType_FX_7517      =  0x07517000, //Micron 5Mp
RPRM_DeviceType_FX_7017      =  0x07017000, //OV 5Mp
```

```
RPRM_DeviceType_FX_7027    = 0x07027000, //OV 5Mp
RPRM_DeviceType_FX_7017D_M = 0x07017100, //OV 5Mp
RPRM_DeviceType_FX_7017D_S = 0x07017200, //OV 5Mp

// 7074
RPRM_DeviceType_EOS_7074_550 = 0x07074550, // Cannon EOS 550D

// 7084
RPRM_DeviceType_EOS_7084_7 = 0x07084007, // Cannon EOS 7D

// 8803 - Banknote Reader
RPRM_DeviceType_EOS_8803_100 = 0x08803100, // Cannon EOS 100D

// 8850 - Passport Reader 1/2 pages
RPRM_DeviceType_EOS_8850_5 = 0x08850005, // Cannon EOS 5DS (5DSR)

// 8824 - Banknote Reader
RPRM_DeviceType_EOS_8824_80= 0x08824080, // Cannon EOS 80D

// 8850M - Passport Reader 1/2 pages
RPRM_DeviceType_EOS_8850M_9= 0x08880009, // Panasonic G9

// 8850F - Passport Reader 2 pages
RPRM_DeviceType_EOS_8850F_9= 0x0888000A, // Panasonic G9

// 8880 - Passport Reader 1/2 pages
RPRM_DeviceType_EOS_8880_1 = 0x08850001, // Panasonic S1R

// 8880F - Passport Reader 2 pages
RPRM_DeviceType_EOS_8880F_1 = 0x08850002,// Panasonic S1R

// Camera Canon EOS
RPRM_DeviceType_CanonEOS   = 0x20000001, // Canon EOS series

// Camera Panasonic
RPRM_DeviceType_Panasonic  = 0x30000009, // Panasonic series

// 4107
RPRM_DeviceType_FX_4137    = 0x04137115, //Micron 3.1
RPRM_DeviceType_FX_4157    = 0x04157115, //Micron 5.0

// 7310
RPRM_DeviceType_Mobile_7310  = 0x07310000, // 1110 Torch + Mobile
device = mobile complex 7310

// ARH
RPRM_DeviceType_ARH        = 0x10000001,

// 3M
RPRM_DeviceType_3M         = 0x10000002,

// TWAIN
RPRM_DeviceType_TWAIN      = 0x10000004,

// Bisys Korea
RPRM_DeviceType_BK         = 0x10000005,

// Bisys Korea New - 7303
RPRM_DeviceType_73xx       = 0x10000006,

// e-seek M500
RPRM_DeviceType_M500       = 0x10000007,

//unused ------------------------------------------------------
RPRM_DeviceType_USB20_1    = 0x00200001, // RdrPassport
                                         // with USB2.0 camera
RPRM_DeviceType_7004s      = 0x00200002, // PR7004S (W+, IR+)
RPRM_DeviceType_7003_01    = 0x00200003, // PR7003_01
                                         // (Wt, Ws, W+, IR+)
RPRM_DeviceType_7003_110   = 0x07003110, // PR7003_111
                                         // (Wt, Ws, W+, IRt,
                                         // IRs, IR+, UV365)
RPRM_DeviceType_7003_111   = 0x07003111, // PR7003_111 (Wt, Ws,
                                         // W+, IRt, IRs, IR+,
```

```
                                                // UV365, 3MW1, 3MW2,
                                              // 3MW+)
  RPRM_DeviceType_7004_100   =  0x07004100, // PR7004_100 /
                                                // PR7005_100 (W+, IR+)
  RPRM_DeviceType_7004_110   =  0x07004110, // PR7004_110 /
                                                // PR7005_110 (W+, IR+,
                                                // UV)
  RPRM_DeviceType_70x4_111   =  0x07004111, // PR70x4_111 /
                                                // PR70x5_111 (W+, IR+,
                                                // UV, 3MW+)
  RPRM_DeviceType_70x4_114   =  0x07004114, // PR70x4_114 /
                                                // PR70x5_114 (Wt, Ws,
                                                // W+, IRt, IRs, IR+,
                                                // UV365, 3MW1, 3MW2,
                                            // 3MW+, 3MIR1, 3MIR2,
                                                // 3MIR+)
  RPRM_DeviceType_70x4_115   =  0x07004115, // PR70x4_115 /
                                                // PR70x5_115 (W(m),
                                                // IR(m), UV365, 3MW1,
                                                // 3MW2, 3MW+)
  RPRM_DeviceType_8303_100   =  0x08303100, // PR8303_100 (W+, IR+)
  RPRM_DeviceType_8303_110   =  0x08303110, // PR8303_110 (W+, IR+,
                                                // UV)
  RPRM_DeviceType_8303_111   =  0x08303111, // PR8303_111 (W+, IR+,
                                                // UV, 3M1, 3M2)
  RPRM_DeviceType_8303_114   =  0x08303114, // PR8303_114 (W_down,
                                              // W_up, W+, IR_down,
                                                // IR_up, IR+, UV365,
                                              // 3MW1, 3MW2, 3MW+)
  RPRM_DeviceType_8303_115   =  0x08303115,
  RPRM_DeviceType_8305       =  0x08305000, // Mobile complex 8305
                                              // (Logitech QuickCam
                                              // Pro web-camera)
  //unused ------------------------------------------------------
};
```

Ниже перечислены значения констант, использующиеся в настоящее время:

- **RPRM_DeviceType_Unknown** Неизвестный тип считывателя документов;

- **RPRM_DeviceType_Virtual** Виртуальный ридер, позволяет обрабатывать файлы изображений;

- **RPRM_DeviceType_FX_8313_115** Считыватель документов 83x3 с камерой OV 1,3 Мп;

- **RPRM_DeviceType_FX_8383_115** Считыватель документов 83x3 с камерой Micron 3,1 Мп;

- **RPRM_DeviceType_FX_8314_115** Считыватель документов 83x3 с камерой OV 1,3 Мп;

- **RPRM_DeviceType_FX_8334_115** Считыватель документов 83x4 с камерой OV 1,3 Мп;

- **RPRM_DeviceType_FX_8384_115** Считыватель документов 83x4 с камерой Micron 3,1 Мп;

• **RPRM_DeviceType_FX_7107_115** Считыватель документов 7007 с камерой OV 1,3 Мп;

• **RPRM_DeviceType_FX_7117_115** Считыватель документов 7007, модифицированный с камерой OV 1,3 Мп;

• **RPRM_DeviceType_FX_8307** Считыватель документов 8307 с камерой Micron 3,1 Мп;

• **RPRM_DeviceType_FX_7104_115** Считыватель документов 70x4 с камерой OV 1,3 Мп;

• **RPRM_DeviceType_FX_7304_115** Считыватель документов 70x4 с камерой OV 3,0 Мп;

• **RPRM_DeviceType_FX_7384_115** Считыватель документов 70x4 с камерой Micron 3,1 Мп;

• **RPRM_DeviceType_FX_7854_115** Считыватель документов 70x4 с камерой Micron 5,0 Мп;

• **RPRM_DeviceType_FX_7103_115** Считыватель документов 70x3 с камерой OV 1,3 Мп;

• **RPRM_DeviceType_FX_7383_115** Считыватель документов 70x3 с камерой Micron 3,1 Мп;

• **RPRM_DeviceType_FX_4821** Считыватель документов 4820 с камерой OV 1,3 Мп;

• **RPRM_DeviceType_FX_4822** Считыватель документов 4820 с камерой Micron 3,1 Мп;

• **RPRM_DeviceType_FX_73x4M** Считыватель документов 70x4M с камерой 3 Мп;

• **RPRM_DeviceType_FX_75x4M** Считыватель документов 70x4M с камерой 5 Мп;

• **RPRM_DeviceType_FX_76x4M** Считыватель документов 70x4M с камерой OV 5 Мп;

• **RPRM_DeviceType_FX_7253** Считыватель документов 72x3 с двумя камерами OV 5 Мп;

- **RPRM_DeviceType_FX_7017** Считыватель документов 7017 с камерой OV 5 Мп;

- **RPRM_DeviceType_FX_7027** Считыватель документов 7027 с камерой OV 5 Мп;

- **RPRM_DeviceType_FX_7017D_M** Считыватель документов 7017D (ведущий) с камерой OV 5 Мп;

- **RPRM_DeviceType_FX_7017D_S** Считыватель документов 7017D (ведомый) с камерой OV 5 Мп;

- **RPRM_DeviceType_FX_7058M** Считыватель документов 70x8M с камерой OV 5 Мп для встраивания;

- **RPRM_DeviceType_EOS_7074_550** Считыватель документов 7074 с камерой 18 Мп;

- **RPRM_DeviceType_FX_78x4M** Считыватель документов 78x4M с камерой 18 Мп;

- **RPRM_DeviceType_Mobile_7310** Мобильный комплекс 7310;

5.4.5. eRPRM_Lights

Перечисление **eRPRM_Lights** содержит набор идентификаторов схем освещения, которые используются для идентификации возможностей считывателя документов, указания схем освещения для сканирования и т. п.

```
enum eRPRM_Lights
{
   RPRM_Light_OFF              =  0x00000000,
   RPRM_Light_OVI              =  0x00000001,
   RPRM_Light_White_Top        =  0x00000002,
   RPRM_Light_White_Side       =  0x00000004,
   RPRM_Light_White_Front      =  0x00800000,
   RPRM_Light_IR_Top           =  0x00000008,
   RPRM_Light_IR_Side          =  0x00000010,
   RPRM_Light_IR_Front         =  0x01000000,
   RPRM_Light_ White_Gray      =  0x02000000,
   RPRM_Light_OVD              =  0x04000000,
   RPRM_Light_Videodetection   =  0x08000000,
   RPRM_Light_UV               =  0x00000080,
   RPRM_Light_IR_Luminescence  =  0x000000100,
   RPRM_Light_AXIAL_White_Left =  0x00000400,
   RPRM_Light_AXIAL_White_Right=  0x00000800,
   RPRM_Light_AXIAL_White_Front=  0x00000200,
   RPRM_Light_IR_720           =  0x00001000,
   RPRM_Light_IR_940           =  0x00002000,
   RPRM_Light_White_Full       =  RPRM_Light_White_Top|
                                  RPRM_Light_White_Side,
   RPRM_Light_IR_Full          =  RPRM_Light_IR_Top|RPRM_Light_IR_Side,
   RPRM_Light_AXIAL_White_Full =  RPRM_Light_AXIAL_White_Left|
                                  RPRM_Light_AXIAL_White_Right,
   RPRM_Light_RAW_Data         =  0x80000000,
   RPRM_Light_RAW_Data_GRBG    =  0x90000000,
   RPRM_Light_RAW_Data_GBGR    =  0xA0000000,
```

```
    RPRM_Light_RAW_Data_RGGB    = 0xB0000000,
    RPRM_Light_RAW_Data_BGGR    = 0xC0000000,
    RPRM_Light_Transmitted      = 0x00000020,
    RPRM_Light_Transmitted_IR   = 0x00000040,
    RPRM_Light_AntiStokes       = 0x00010000,
    RPRM_Light_Transmitted_IR940
                                = 0x00004000,
    RPRM_Light_OVD_Right        = 0x00040000,
    RPRM_Light_OVD_Left         = 0x00020000,
    RPRM_Light_IR_700           = 0x00008000,
    RPRM_Light_IR_870           = RPRM_Light_IR_Front, // Light IR 870
    (mod. 8803),
    RPRM_Light_White_Obl        = RPRM_Light_White_Side,    //    Light
    White oblique (mod. 8850),
    RPRM_Light_Holo             = RPRM_Light_OVD,    //    Light    OVD
    (hologram visualization) - 8850,
    RPRM_Light_IR_870_Obl       = RPRM_Light_IR_Side, // Light IR 870
    oblique (mod. 8850),
    RPRM_Light_IR_Bottom        = RPRM_Light_Transmitted_IR, // Light
    IR Bottom (mod. 8304),
    RPRM_Light_White_Bottom     = RPRM_Light_Transmitted,   //   Light
    White Bottom (mod. 8304)
    RPRM_Light_UVC              = 0x00080000, // Light UVC 254 (88X0
    devices)
    RPRM_Light_UVB              = 0x00100000, // Light UVB 313 (88X0
    devices)
};
```

Значение констант:

- **RPRM_Light_OFF** Отсутствие включенной схемы освещения;
- **RPRM_Light_OVI** Схема OVI;
- **RPRM_Light_White_Top** Верхний/нижний осветители схемы белого света;
- **RPRM_Light_White_Side** Боковые осветители схемы белого света;
- **RPRM_Light_White_Front** Схема общего белого света, без возможности раздельного управления боковыми и верхним/нижним осветителями;
- **RPRM_Light_IR_Top** Верхний/нижний осветители схемы ИК-света;
- **RPRM_Light_IR_Side** Боковые осветители схемы ИК-света;
- **RPRM_Light_IR_Front** Схема общего ИК-света без возможности раздельного управления боковыми и верхним/нижним осветителями;
- **RPRM_Light_White_Gray** Схема общего белого света, переведенного в градации серого;
- **RPRM_Light_OVD** Схема OVD для визуализации голограмм;
- **RPRM_Light_Videodetection** Схема видеодетекции, только для служебного использования;

- **RPRM_Light_UV** Схема общего УФ-света;
- **RPRM_Light_IR_Luminescence** ИК-люминесценция;
- **RPRM_Light_AXIAL_White_Left** Левый осветитель схемы белого коаксиального света;
- **RPRM_Light_AXIAL_White_Right** Правый осветитель схемы белого коаксиального света;
- **RPRM_Light_AXIAL_White_Front** Схема общего белого коаксиального света без возможности раздельного управления левым и правым осветителями;
- **RPRM_Light_IR_720** ИК720;
- **RPRM_Light_IR_940** ИК940;
- **RPRM_Light_White_Full** Верхний/нижний и боковые осветители схемы белого света;
- **RPRM_Light_IR_Full** Верхний/нижний и боковые осветители схемы ИК света;
- **RPRM_Light_AXIAL_White_Full** Левый и правый осветители схемы белого коаксиального света;
- **RPRM_Light_IR_Bottom** Зарезервировано для внутреннего использования;
- **RPRM_Light_White_Bottom** Зарезервировано для внутреннего использования;
- **RPRM_Light_RAW_Data_xxx** Зарезервировано для внутреннего использования;
- **RPRM_Light_Transmitted** Просветный;
- **RPRM_Light_Transmitted_IR** Просветный ИК;
- **RPRM_Light_AntiStokes** Просветный Антистокс;
- **RPRM_Light_Transmitted_IR940** Просветный ИК940;

- **RPRM_Light_OVD_Right** OVD правый;
- **RPRM_Light_OVD_Left** OVD левый;
- **RPRM_Light_IR_700** ИК700;
- **RPRM_Light_IR_870** Фронтальный ИК870 для модели 8803;
- **RPRM_Light_White_Obl** Белый косопадающий для модели 8850;
- **RPRM_Light_Holo** OVD свет (визуализация голограмм) для модели 8850;
- **RPRM_Light_IR_870_Obl** Косопадающий ИК870 для модели 8850;
- **RPRM_Light_UVC** УФС 254 для модели 88X0;
- **RPRM_Light_UVB** УФВ 313 для модели 88X0.

Если считыватель оборудуется схемами освещения белого и ИК-света с матричным управлением, последующие константы определяют расположение активизируемых источников освещения. Для указания схемы освещения (белого или ИК), к которой они будут применены, их необходимо использовать в комбинации с одним из вышеописанных значений.

5.4.6. eRPRM_VideoModes

Перечисление **eRPRM_VideoModes** содержит набор идентификаторов, определяющих поддерживаемые видеокамерой устройства размеры получаемых изображений.

```
enum eRPRM_VideoModes
{
   RPRM_VM_UNDEFINED = 0,
   RPRM_VM_1MP       =  0x00000001,
   RPRM_VM_3MP       =  0x00000002,
   RPRM_VM_5MP       =  0x00000004,
   RPRM_VM_3MP_MRZ   =  0x00000010,
   RPRM_VM_9MP_2     =  0x00000020,
   RPRM_VM_9MP       =  0x00000040,
   RPRM_VM_5MP_1_3   =  0x00000080,
   RPRM_VM_50MP      =  0x01000000,
   RPRM_VM_50MP_3    =  0x02000000,
   RPRM_VM_24MP      =  0x04000000,
   RPRM_VM_24MP_1_5  =  0x08000000,
   RPRM_VM_20MP      =  0x00100000,
   RPRM_VM_80MP      =  0x00200000,
   RPRM_VM_ID1xx_600 =  0x10000000,
   RPRM_VM_300DPI    =  0x20000000,
```

```
   RPRM_VM_5MP_EOS    =  0x40000000,
   RPRM_VM_14MP       =  0x00000100,
   RPRM_VM_FULL_HD    =  0x00000200,
   RPRM_VM_10MP       =  0x00000400,
   RPRM_VM_10MP_2_5   =  0x00000800,
   RPRM_VM_MAX        =  RPRM_VM_14MP,
};
```

Значение констант:

- **RPRM_VM_UNDEFINED** – режим не определен;
- **RPRM_VM_1MP** – поддерживается получение кадров размером 1280×1024 или 1024×768;
- **RPRM_VM_3MP** – поддерживается получение кадров размером 2048×1536;
- **RPRM_VM_5MP** – поддерживается получение кадров размером 2592x1944;
- **RPRM_VM_3MP_MRZ** – режим работы прибора типа `RPRM_DeviceType_FX_8307` (1536×500);
- **RPRM_VM_9MP_2** – поддерживается получение кадров размером 1744x1308;
- **RPRM_VM_9MP** – поддерживается получение кадров размером 3488x2616;
- **RPRM_VM_5MP_1_3** – поддерживается получение кадров размером 1296x972 (1296x968 для 7017 и 72x3 в режиме JPEG);
- **RPRM_VM_50MP** – поддерживается получение кадров в разрешении 50 МП;
- **RPRM_VM_50MP_3** – поддерживается получение кадров, уменьшенных до 3 МП;
- **RPRM_VM_24MP** – поддерживается получение кадров в разрешении 24 МП;
- **RPRM_VM_24MP_1_5** – поддерживается получение кадров, уменьшенных до 1,5 МП;
- **RPRM_VM_20MP** – поддерживается получение кадров в разрешении 20 МП;
- **RPRM_VM_80MP** – поддерживается получение кадров в разрешении 80 МП;
- **RPRM_VM_ID1xx_600** – поддерживается получение кадров размером 600 DPI со сканеров ID1;
- **RPRM_VM_300DPI** – поддерживается получение кадров размером 300 DPI со сканеров ID1;
- **RPRM_VM_5MP_EOS** – поддерживается получение кадров размером 2592x1728;
- **RPRM_VM_14MP** – поддерживается получение кадров размером 4384x3288;
- **RPRM_VM_FULL_HD** – поддерживается получение кадров размером 1920x1080;
- **RPRM_VM_10MP** – поддерживается получение кадров размером 3664x2744;
- **RPRM_VM_10MP_2_5** – поддерживается получение кадров размером 1832x1372;
- **RPRM_VM_MAX** – для внутреннего использования.

Остальные значения зарезервированы для внутреннего использования.

5.4.7. CDocFormat

Перечисление **CDocFormat** содержит набор идентификаторов, определяющих геометрический формат документов в соответствии с ISO/IEC 7810.

```
enum CDocFormat
{
   dfID1        =  0,
```

```
   dfID2        = 1,
   dfID3        = 2,
   dfNON        = 3,
   dfA4         = 4,
   dfID3_x2     = 5,
   dfID2_Turkey= 6,
   dfID1_90     = 10,
   dfID1_180    = 11,
   dfID1_270    = 12,
   dfID2_180    = 13,
   dfID3_180    = 14,
   dfCustom     = 1000,
   dfPhoto      = 1001,
   dfFlexible   = 1002
   dfUnknown    = -1
};
```

Значение констант:

- **dfID1** – документ формата ID1;
- **dfID2** – документ формата ID2;
- **dfID3** – документ формата ID3;
- **dfNON** – формат документа не определен;
- **dfA4** – документ формата A4;
- **dfID3_x2** – двойной документ формата ID3;
- **dfID2_Turkey**– турецкая карта формата ID2;
- **dfID1_90** – документ формата ID1, повернут на 90°;
- **dfID1_180** – документ формата ID1, повернут на 180°;
- **dfID1_270** – документ формата ID1, повернут на 27°;
- **dfID2_90** – документ формата ID2, повернут на 90°;
- **dfID3_180** – документ формата ID3, повернут на 180°;
- **dfCustom** – документ неопределенного формата, для внутреннего использования;
- **dfPhoto** – фотография;
- **dfFlexible** – неформатный документ, для внутреннего использования;
- **dfUnknown** – неизвестный формат документа.

5.4.8. eRPRM_Capabilities

Перечисление **eRPRM_Capabilities** содержит идентификаторы, определяющие комбинацию возможностей главной управляющей библиотеки SDK по получению и обработке данных.

```
enum eRPRM_Capabilities
{
   RPRM_Capabilities_Empty                        = 0x00000000,
   RPRM_Capabilities_Scan                         = 0x00000001,
   RPRM_Capabilities_SaveFiles                    = 0x00000002,
   RPRM_Capabilities_LocateDocument               = 0x00000004,
   RPRM_Capabilities_MRZ_OCR                      = 0x00000008,
   RPRM_Capabilities_Visual_OCR                   = 0x00000010,
   RPRM_Capabilities_BarCodes                     = 0x00000020,
   RPRM_Capabilities_MRZ_TestQuality              = 0x00000040,
   RPRM_Capabilities_FDS                          = 0x00000080,
   RPRM_Capabilities_ImageDistortionCompensation  = 0x00000100,
   RPRM_Capabilities_OCR_Analyze                  = 0x00000200,
   RPRM_Capabilities_Authenticity                 = 0x00000400,
   RPRM_Capabilities_RAW_ImageData                = 0x00000800,
   RPRM_Capabilities_RAW_CustomDemosaic           = 0x00001000,
```

```
   RPRM_Capabilities_DocumentType              = 0x00002000,
   RPRM_Capabilities_Visual_Graphics           = 0x00004000,
   RPRM_Capabilities_Expert_Analyze            = 0x00008000,
   RPRM_Capabilities_ColorCompensation         = 0x00010000,
   RPRM_Capabilities_BarcodesExtended          = 0x00020000,
   RPRM_Capabilities_GlareCompensation         = 0x00040000,
   RPRM_Capabilities_RFID                      = 0x00080000,
   RPRM_Capabilities_BankCard                  = 0x00100000,
   RPRM_Capabilities_LiveFaceComparison        = 0x00200000,
   RPRM_Capabilities_ProcessImages             = 0x10000000,
   RPRM_Capabilities_Custom2                   = 0x20000000,
   RPRM_Capabilities_Custom3                   = 0x40000000,
   RPRM_Capabilities_Custom4                   = 0x80000000,
   RPRM_Capabilities_Max                       = 0xFFFFFFFF
};
```

Значение констант:

- **RPRM_Capabilities_Empty** Набор функциональных возможностей пуст;

- **RPRM_Capabilities_Scan** Доступно сканирование изображений;

- **RPRM_Capabilities_SaveFiles** Доступно формирование образов графических файлов для получаемых изображений;

- **RPRM_Capabilities_LocateDocument** Доступна процедура нахождения документа на отсканированных изображениях и формирования результирующих изображений документа, вырезанных по найденным границам, с проведенной компенсацией неравномерности освещения и коррекцией цветового баланса;

- **RPRM_Capabilities_MRZ_OCR** Доступна процедура чтения МСЗ документа;

- **RPRM_Capabilities_Visual_OCR** Доступна процедура чтения зон заполнения документа;

- **RPRM_Capabilities_BarCodes** Доступна процедура поиска и чтения штрихкодов;

- **RPRM_Capabilities_MRZ_TestQuality** Доступна процедура оценки качества заполнения МСЗ;

- **RPRM_Capabilities_FDS** Доступна совместная работа с СИС;

- **RPRM_Capabilities_ImageDistortionCompensation** Доступна процедура коррекции геометрических искажений изображений;

- **RPRM_Capabilities_OCR_Analyze** Доступна процедура лексического анализа результатов чтения текстовых данных МСЗ, зоны заполнения документа, штрихкодов, данных из памяти RFID-микросхемы;

- **RPRM_Capabilities_Authenticity** Доступна процедура оценки подлинности документа по получаемым изображениям;

- **RPRM_Capabilities_RAW_ImageData** Есть возможность получать необработанные данные с видеочипа;

- **RPRM_Capabilities_RAW_CustomDemosaic** Доступна библиотека для проведения внешнего демозаика необработанных картинок с видеочипа;

- **RPRM_Capabilities_DocumentType** Доступна процедура определения типа документа;

- **RPRM_Capabilities_Visual_Graphics** Доступна процедура вырезания графических полей из визуальной зоны документа;

- **RPRM_Capabilities_Expert_Analyze** Не используется;

- **RPRM_Capabilities_ColorCompensation** Не используется;

- **RPRM_Capabilities_BarcodesExtended** Не используется;

- **RPRM_Capabilities_GlaresCompensation** Не используется;

- **RPRM_Capabilities_RFID** Не используется;

- **RPRM_Capabilities_BankCard** Не используется;

- **RPRM_Capabilities_LiveFaceComparison** Сравнение фото с камеры и или из внешнего источника с фото на документе. Необходим отдельный сервис;

- **RPRM_Capabilities_ProcessImages** Обработка внешних изображений (см. RPRM_Command_ProcessImagesList);

- **RPRM_Capabilities_Max** Все значения.

5.4.9. eRPRM_GetImage_Modes

Перечисление **eRPRM_GetImage_Modes** содержит идентификаторы, определяющие комбинацию функций получения и обработки изображений, которую необходимо выполнить в ходе цикла сканирования и обработки, и набор данных, которые необходимо получить в качестве результата.

```
enum eRPRM_GetImage_Modes
{
   RPRM_GetImage_Modes_Empty                = 0x00000000,
   RPRM_GetImage_Modes_GetUncroppedImages   = 0x00000001,
   RPRM_GetImage_Modes_ReceiveAllScannedImages  = 0x00000002,
   RPRM_GetImage_Modes_OCR_Visual_Graphics  = 0x00000004,
   RPRM_GetImage_Modes_GetImages            = 0x00000008,
   RPRM_GetImage_Modes_LocateDocument       = 0x00000010,
   RPRM_GetImage_Modes_DocumentType         = 0x00000020,
   RPRM_GetImage_Modes_OCR_MRZ              = 0x00000040,
   RPRM_GetImage_Modes_OCR_Visual_Text      = 0x00000080,
   RPRM_GetImage_Modes_OCR_Visual           =
   RPRM_GetImage_Modes_OCR_Visual_Graphics|
   RPRM_GetImage_Modes_OCR_Visual_Text,
   RPRM_GetImage_Modes_OCR_BarCodes         = 0x00000100,
   RPRM_GetImage_Modes_Authenticity         = 0x00000200,
   RPRM_GetImage_Modes_OCR_TestMRZQuality   = 0x00000400,
   RPRM_GetImage_Modes_RAW_Data             = 0x00000800,
   RPRM_GetImage_Modes_RAW_Data_Only        = 0x00001000,
   RPRM_GetImage_Modes_NoColorCompensation= 0x00002000,
   RPRM_GetImage_Modes_NoDistortionCompensation
                                            = 0x00004000,
   RPRM_GetImage_Modes_DetectDocument       = 0x00008000,
   RPRM_GetImage_Modes_ImageQA              = 0x00010000,
   RPRM_GetImage_Modes_Holo                 = 0x00020000,
   RPRM_GetImage_Modes_Reserved10           = 0x00040000,
   RPRM_GetImage_Modes_Reserved11           = 0x00080000,
   RPRM_GetImage_Modes_Reserved12           = 0x00100000,
   RPRM_GetImage_Modes_Reserved13           = 0x00200000,
   RPRM_GetImage_Modes_Reserved14           = 0x00400000,
   RPRM_GetImage_Modes_Reserved15           = 0x00800000,
   RPRM_GetImage_Modes_Reserved16           = 0x01000000,
   RPRM_GetImage_Modes_Reserved17           = 0x02000000,
   RPRM_GetImage_Modes_Reserved18           = 0x04000000,
   RPRM_GetImage_Modes_Reserved19           = 0x08000000,
   RPRM_GetImage_Modes_Custom1              = 0x10000000,
   RPRM_GetImage_Modes_Custom2              = 0x20000000,
   RPRM_GetImage_Modes_Custom3              = 0x40000000,
   RPRM_GetImage_Modes_Custom4              = 0x80000000
};
```

Значение констант:

- **RPRM_GetImage_Modes_Empty** «Холостой» цикл сканирования без получения каких-либо результатов;

- **RPRM_GetImage_Modes_GetUncroppedImages** НЕ ИСПОЛЬЗУЕТСЯ. Необходимо получить необработанные изображения документа;

- **RPRM_GetImage_Modes_ReceiveAllScannedImages** Необходимо получить все отсканированные изображения документа с учетом процесса *дополнительного* сканирования;

- **RPRM_GetImage_Modes_OCR_Visual_Graphics** Необходимо получить графические поля с визуальной страницы;

- **RPRM_GetImage_Modes_GetImages** Необходимо получить изображения документа, принятые для схем освещения, входящих в текущий *список схем освещения для сканирования;*

- **RPRM_GetImage_Modes_LocateDocument** Необходимо провести процедуру нахождения документа на отсканированных изображениях, формирования результирующих изображений документа, вырезанных по найденным границам, с проведенной компенсацией неравномерности освещения и коррекцией цветового баланса;

- **RPRM_GetImage_Modes_DocumentType** Необходимо провести процедуру определения типа документа;

- **RPRM_GetImage_Modes_OCR_MRZ** Необходимо провести процедуру чтения МСЗ документа;

- **RPRM_GetImage_Modes_OCR_Visual_Text** Необходимо провести процедуру чтения информации из текстовых полей заполнения документа;

- **RPRM_GetImage_Modes_OCR_Visual** Необходимо провести процедуру чтения информации из текстовых и графических полей заполнения документа;

- **RPRM_GetImage_Modes_OCR_BarCodes** Необходимо провести процедуру поиска и чтения штрихкодов;

- **RPRM_GetImage_Modes_Authenticity** Необходимо провести процедуру контроля подлинности документа по получаемым изображениям;

- **RPRM_GetImage_Modes_OCR_TestMRZQuality** Необходимо провести процедуру контроля качества заполнения МСЗ документа;

- **RPRM_GetImage_Modes_RAW_Data** НЕ ИСПОЛЬЗУЕТСЯ. Необходимо получить RAW-данные;

- **RPRM_GetImage_Modes_RAW_Data_Only** НЕ ИСПОЛЬЗУЕТСЯ. Необходимо получить только RAW-данные;

- **RPRM_GetImage_Modes_NoColorCompensation** Необходимо отключить компенсацию яркости;

- **RPRM_GetImage_Modes_NoDistortionCompensation** Необходимо отключить компенсацию дисторсии;

- **RPRM_GetImage_Modes_DetectDocument** Необходимо получить границы документа;

- **RPRM_GetImage_Modes_ImageQA** Необходимо провести проверку качества входного изображения (только при обработке ранее полученных изображений);

- **RPRM_GetImage_Modes_Holo** Зарезервировано.

5.4.10. eRPRM_FieldVerificationResult

Перечисление **eRPRM_FieldVerificationResult** содержит идентификаторы, определяющие статус проверки и сравнения текстовых полей.

```
enum eRPRM_FieldVerificationResult
{
   RCF_Disabled          = 0,
   RCF_Verified          = 1,
   RCF_Not_Verified      = 2,
   RCF_Compare_True      = 3,
   RCF_Compare_False     = 4
};
```

Значения имеют следующий смысл:

- `RCF_Disabled` – проверка не проводилась, результат не определен;
- `RCF_Verified` – проверка прошла успешно;
- `RCF_Not_Verified` – проверка не прошла;
- `RCF_Compare_True` – положительный результат сравнения;
- `RCF_Compare_False` – отрицательный результат сравнения.

5.4.11. eVisualFieldType

Перечисление **eVisualFieldType** содержит идентификаторы, определяющие логический тип текстовых данных, полученных при распознавании МСЗ или штрихкодов, чтении полей заполнения документа.

```
enum eVisualFieldType
{
   ft_Document_Class_Code                = 0,
   ft_Issuing_State_Code                 = 1,
   ft_Document_Number                    = 2,
   ft_Date_of_Expiry                     = 3,
   ft_Date_of_Issue                      = 4,
   ft_Date_of_Birth                      = 5,
```

```
    ft_Place_of_Birth                       =  6,
    ft_Personal_Number                      =  7,
    ft_Surname                              =  8,
    ft_Given_Names                          =  9,
    ft_Mothers_Name                         =  10,
    ft_Nationality                          =  11,
    ft_Sex                                  =  12,
    ft_Height                               =  13,
    ft_Weight                               =  14,
    ft_Eyes_Color                           =  15,
    ft_Hair_Color                           =  16,
    ft_Address                              =  17,
    ft_Donor                                =  18,
    ft_Social_Security_Number               =  19,
    ft_DL_Class                             =  20,
    ft_DL_Endorsed                          =  21,
    ft_DL_Restriction_Code                  =  22,
    ft_DL_Under_21_Date                     =  23,
    ft_Authority                            =  24,
    ft_Surname_And_Given_Names          =    25,
    ft_Nationality_Code                     =  26,
    ft_Passport_Number                      =  27,
    ft_Invitation_Number                    =  28,
    ft_Visa_ID                              =  29,
    ft_Visa_Class                           =  30,
    ft_Visa_SubClass                        =  31,
    ft_MRZ_String1                          =  32,
    ft_MRZ_String2                          =  33,
    ft_MRZ_String3                          =  34,
    ft_MRZ_Type                             =  35,
    ft_Optional_Data                        =  36,
    ft_Document_Class_Name                  =  37,
    ft_Issuing_State_Name                   =  38,
    ft_Place_of_Issue                       =  39,
    ft_Document_Number_Checksum             =  40,
    ft_Date_of_Birth_Checksum               =  41,
    ft_Date_of_Expiry_Checksum              =  42,
    ft_Personal_Number_Checksum             =  43,
    ft_FinalChecksum                        =  44,
    ft_Passport_Number_Checksum             =  45,
    ft_Invitation_Number_Checksum           =  46,
    ft_Visa_ID_Checksum                     =  47,
    ft_Surname_And_Given_Names_Checksum     =  48,
    ft_Visa_Valid_Until_Checksum            =  49,
    ft_Other                                =  50,
    ft_MRZ_Strings                          =  51,
    ft_Name_Suffix                          =  52,
    ft_Name_Prefix                          =  53,
    ft_Date_of_Issue_Checksum               =  54,
    ft_Date_of_Issue_CheckDigit             =  55,
    ft_Document_Series                      =  56,
    ft_RegCert_RegNumber                    =  57,
    ft_RegCert_CarModel                     =  58,
    ft_RegCert_CarColor                     =  59,
    ft_RegCert_BodyNumber                   =  60,
    ft_RegCert_CarType                      =  61,
    ft_RegCert_MaxWeight                    =  62,
    ft_Reg_Cert_Weight                      =  63,
    ft_Address_Area                         =  64,
    ft_Address_State                        =  65,
    ft_Address_Building                     =  66,
    ft_Address_House                        =  67,
    ft_Address_Flat                         =  68,
    ft_Place_of_Registration                =  69,
    ft_Date_of_Registration                 =  70,
    ft_Resident_From                        =  71,
    ft_Resident_Until                       =  72,
    ft_Authority_Code                       =  73,
    ft_Place_of_Birth_Area                  =  74,
    ft_Place_of_Birth_StateCode             =  75,
    ft_Address_Street                       =  76,
    ft_Address_City                         =  77,
    ft_Address_Jurisdiction_Code            =  78,
    ft_Address_Postal_Code                  =  79,
    ft_Document_Number_CheckDigit           =  80,
    ft_Date_of_Birth_CheckDigit             =  81,
    ft_Date_of_Expiry_CheckDigit            =  82,
    ft_Personal_Number_CheckDigit           =  83,
    ft_FinalCheckDigit                      =  84,
    ft_Passport_Number_CheckDigit           =  85,
    ft_Invitation_Number_CheckDigit         =  86,
    ft_Visa_ID_CheckDigit                   =  87,
    ft_Surname_And_Given_Names_CheckDigit   =  88,
    ft_Visa_Valid_Until_CheckDigit          =  89,
```

```
ft_Permit_DL_Class                    =  90,
ft_Permit_Date_of_Expiry              =  91,
ft_Permit_Identifier                  =  92,
ft_Permit_Date_of_Issue               =  93,
ft_Permit_Restriction_Code            =  94,
ft_Permit_Endorsed                    =  95,
ft_Issue_Timestamp                    =  96,
ft_Number_of_Duplicates               =  97,
ft_Medical_Indicator_Codes            =  98,
ft_Non_Resident_Indicator             =  99,
ft_Visa_Type                          =  100,
ft_Visa_Valid_From                    =  101,
ft_Visa_Valid_Until                   =  102,
ft_Duration_of_Stay                   =  103,
ft_Number_of_Entries                  =  104,
ft_Day                                =  105,
ft_Month                              =  106,
ft_Year                               =  107,
ft_Unique_Customer_Identifier         =  108,
ft_Commercial_Vehicle_Codes           =  109,
ft_AKA_Date_of_Birth                  =  110,
ft_AKA_Social_Security_Number         =  111,
ft_AKA_Surname                        =  112,
ft_AKA_Given_Names                    =  113,
ft_AKA_Name_Suffix                    =  114,
ft_AKA_Name_Prefix                    =  115,
ft_Mailing_Address_Street             =  116,
ft_Mailing_Address_City               =  117,
ft_Mailing_Address_Jurisdiction_Code  =  118,
ft_Mailing_Address_Postal_Code        =  119,
ft_Audit_Information                  =  120,
ft_Inventory_Number                   =  121,
ft_Race_Ethnicity                     =  122,
ft_Jurisdiction_Vehicle_Class         =  123,
ft_Jurisdiction_Endorsement_Code      =  124,
ft_Jurisdiction_Restriction_Code      =  125,
ft_Family_Name                        =  126,
ft_Given_Names_RUS                    =  127,
ft_Visa_ID_RUS                        =  128,
ft_Fathers_Name                       =  129,
ft_Fathers_Name_RUS                   =  130,
ft_Surname_And_Given_Names_RUS        =  131,
ft_Place_Of_Birth_RUS                 =  132,
ft_Authority_RUS                      =  133,
ft_Issuing_State_Code_Numeric         =  134,
ft_Nationality_Code_Numeric           =  135,
ft_Engine_Power                       =  136,
ft_Engine_Volume                      =  137,
ft_Chassis_Number                     =  138,
ft_Engine_Number                      =  139,
ft_Engine_Model                       =  140,
ft_Vehicle_Category                   =  141,
ft_Identity_Card_Number               =  142,
ft_Control_No                         =  143,
ft_Parrent_s_Given_Names              =  144,
ft_Second_Surname                     =  145,
ft_Middle_Name                        =  146,
ft_RegCert_VIN                        =  147,
ft_RegCert_VIN_CheckDigit             =  148,
ft_RegCert_VIN_Checksum               =  149,
ft_Line1_CheckDigit                   =  150,
ft_Line2_CheckDigit                   =  151,
ft_Line3_CheckDigit                   =  152,
ft_Line1_Checksum                     =  153,
ft_Line2_Checksum                     =  154,
ft_Line3_Checksum                     =  155,
ft_RegCert_RegNumber_CheckDigit       =  156,
ft_RegCert_RegNumber_Checksum         =  157,
ft_RegCert_Vehicle_ITS_Code           =  158,
ft_Card_Access_Number                 = 159,
ft_Marital_Status                     = 160,
ft_Company_Name                       = 161,
ft_Special_Notes                      = 162,
ft_Surname_of_Spose                   = 163,
ft_Tracking_Number                    = 164,
ft_Booklet_Number                     = 165,
ft_Children                           = 166,
ft_Copy                               = 167,
ft_Serial_Number                      = 168,
ft_Dossier_Number                     = 169,
ft_AKA_Surname_And_Given_Names        = 170,
ft_Territorial_Validity               = 171,
ft_MRZ_Strings_With_Correct_CheckSums = 172,
ft_DL_CDL_Restriction_Code            = 173,
```

```
ft_DL_Under_18_Date                    = 174,
ft_DL_Record_Created                   = 175,
ft_DL_Duplicate_Date                   = 176,
ft_DL_Iss_Type                         = 177,
ft_Military_Book_Number                = 178,
ft_Destination                         = 179,
ft_Blood_Group                         = 180,
ft_Sequence_Number                     = 181,
ft_RegCert_BodyType                    = 182,
ft_RegCert_CarMark                     = 183,
ft_Transaction_Number                  = 184,
ft_Age                                 = 185,
ft_Folio_Number                        = 186,
ft_Voter_Key                           = 187,
ft_Address_Municipality                = 188,
ft_Address_Location                    = 189,
ft_Section                             = 190,
ft_OCR_Number                          = 191,
ft_Federal_Elections                   = 192,
ft_Reference_Number                    = 193,
ft_Optional_Data_Checksum              = 194,
ft_Optional_Data_CheckDigit            = 195,
ft_Visa_Number                         = 196,
ft_Visa_Number_Checksum                = 197,
ft_Visa_Number_CheckDigit              = 198,
ft_Voter                               = 199,
ft_Previous_Type                       = 200,
ft_FieldFromMRZ                        = 220,
ft_CurrentDate                         = 221,
ft_Status_Date_of_Expiry               = 251,
ft_Bancnote_Number                     = 252,
ft_CSC_Code                            = 253,
ft_Artistic_Name                       = 254,
ft_Academic_Title                      = 255,
ft_Address_Country                     = 256,
ft_Address_Zipcode                     = 257,
ft_eID_Residence_Permit1               = 258,
ft_eID_Residence_Permit2               = 259,
ft_eID_PlaceOfBirth_Street             = 260,
ft_eID_PlaceOfBirth_City               = 261,
ft_eID_PlaceOfBirth_State              = 262,
ft_eID_PlaceOfBirth_Country            = 263,
ft_eID_PlaceOfBirth_Zipcode            = 264,
ft_CDL_Class                           = 265,
ft_DL_Under_19_Date                    = 266,
ft_Weight_pounds                       = 267,
ft_Limited_Duration_Document_Indicator = 268,
ft_Endorsement_Expiration_Date         = 269,
ft_Revision_Date                       = 270,
ft_Compliance_Type                     = 271,
ft_Family_name_truncation              = 272,
ft_First_name_truncation               = 273,
ft_Middle_name_truncation              = 274,
ft_Exam_Date                           = 275,
ft_Organization                        = 276,
ft_Department                          = 277,
ft_Pay_Grade                           = 278,
ft_Rank                                = 279,
ft_Benefits_Number                     = 280,
ft_Sponsor_Service                     = 281,
ft_Sponsor_Status                      = 282,
ft_Sponsor                             = 283,
ft_Relationship                        = 284,
ft_USCIS                               = 285,
ft_Category                            = 286,
ft_Conditions                          = 287,
ft_Identifier                          = 288,
ft_Configuration                       = 289,
ft_Discretionary_data                  = 290,
ft_Line1_Optional_Data                 = 291,
ft_Line2_Optional_Data                 = 292,
ft_Line3_Optional_Data                 = 293,
ft_EQV_Code                            = 294,
ft_ALT_Code                            = 295,
ft_Binary_Code                         = 296,
ft_Pseudo_Code                         = 297,
ft_Fee                                 = 298,
ft_Stamp_Number                        = 299,
ft_SBH_SecurityOptions                 = 300,
ft_SBH_IntegrityOptions                = 301,
ft_Date_of_Creation                    = 302,
ft_Validity_Period                     = 303,
ft_Patron_Header_Version               = 304,
ft_BDB_Type                            = 305,
```

```
ft_Biometric_Type                       = 306,
ft_Biometric_Subtype                    = 307,
ft_Biometric_ProductID                  = 308,
ft_Biometric_Format_Owner               = 309,
ft_Biometric_Format_Type                = 310,
ft_Phone                                = 311,
ft_Profession                           = 312,
ft_Title                                = 313,
ft_Personal_Summary                     = 314,
ft_Other_Valid_ID                       = 315,
ft_Custody_Info                         = 316,
ftOther_Name                            = 317,
ft_Observations                         = 318,
ft_Tax                                  = 319,
ft_Date_of_Personalization              = 320,
ft_Personalization_SN                   = 321,
ft_OtherPerson_Name                     = 322,
ft_PersonToNotify_Date_of_Record        = 323,
ft_PersonToNotify_Name                  = 324,
ft_PersonToNotify_Phone                 = 325,
ft_PersonToNotify_Address               = 326,
ft_DS_Certificate_Issuer                = 327,
ft_DS_Certificate_Subject               = 328,
ft_DS_Certificate_ValidFrom             = 329,
ft_DS_Certificate_ValidTo               = 330,
ft_VRC_DataObject_Entry                 = 331,
ft_TypeApprovalNumber                   = 332,
ft_AdministrativeNumber                 = 333,
ft_DocumentDiscriminator                = 334,
ft_DataDiscriminator                    = 335,
ft_ISO_Issuer_ID_Number                 = 336,
ft_GNIB_Number                          = 340,
ft_Dept_Number                          = 341,
ft_Telex_Code                           = 342,
ft_Allergies                            = 343,
ft_Sp_Code                              = 344,
ft_Court_Code                           = 345,
ft_Cty                                  = 346,
ft_Sponsor_SSN                          = 347,
ft_DoD_Number                           = 348,
ft_MC_Novice_Date                       = 349,
ft_DUF_Number                           = 350,
ft_AGY                                  = 351,
ft_PNR_Code                             = 352,
ft_From_Airport_Code                    = 353,
ft_To_Airport_Code                      = 354,
ft_Flight_Number                        = 355,
ft_Date_of_Flight                       = 356,
ft_Seat_Number                          = 357,
ft_Date_of_Issue_Boarding_Pass          = 358,
ft_CCW_Until                            = 359,
ft_Reference_Number_Checksum            = 360,
ft_Reference_Number_CheckDigit          = 361,
ft_Room_Number                          = 362,
ft_Religion                             = 363,
ft_RemainderTerm                        = 364,
ft_Electronic_Ticket_Indicator          = 365,
ft_Compartment_Code                     = 366,
ft_CheckIn_Sequence_Number              = 367,
ft_Airline_Designator_of_boarding_pass_issuer = 368,
ft_Airline_Numeric_Code                 = 369,
ft_Ticket_Number                        = 370,
ft_Frequent_Flyer_Airline_Designator    = 371,
ft_Frequent_Flyer_Number                = 372,
ft_Free_Baggage_Allowance               = 373,
ft_PDF417Codec                          = 374,
ft_Identity_Card_Number_Checksum        = 375,
ft_Identity_Card_Number_CheckDigit      = 376,
ft_Veteran                              = 377,
ft_DLClassCode_A1_From                  = 378,
ft_DLClassCode_A1_To                    = 379,
ft_DLClassCode_A1_Notes                 = 380,
ft_DLClassCode_A_From                   = 381,
ft_DLClassCode_A_To                     = 382,
ft_DLClassCode_A_Notes                  = 383,
ft_DLClassCode_B_From                   = 384,
ft_DLClassCode_B_To                     = 385,
ft_DLClassCode_B_Notes                  = 386,
ft_DLClassCode_C1_From                  = 387,
ft_DLClassCode_C1_To                    = 388,
ft_DLClassCode_C1_Notes                 = 389,
ft_DLClassCode_C_From                   = 390,
ft_DLClassCode_C_To                     = 391,
ft_DLClassCode_C_Notes                  = 392,
```

```
ft_DLClassCode_D1_From                = 393,
ft_DLClassCode_D1_To                  = 394,
ft_DLClassCode_D1_Notes               = 395,
ft_DLClassCode_D_From                 = 396,
ft_DLClassCode_D_To                   = 397,
ft_DLClassCode_D_Notes                = 398,
ft_DLClassCode_BE_From                = 399,
ft_DLClassCode_BE_To                  = 400,
ft_DLClassCode_BE_Notes               = 401,
ft_DLClassCode_C1E_From               = 402,
ft_DLClassCode_C1E_To                 = 403,
ft_DLClassCode_C1E_Notes              = 404,
ft_DLClassCode_CE_From                = 405,
ft_DLClassCode_CE_To                  = 406,
ft_DLClassCode_CE_Notes               = 407,
ft_DLClassCode_D1E_From               = 408,
ft_DLClassCode_D1E_To                 = 409,
ft_DLClassCode_D1E_Notes              = 410,
ft_DLClassCode_DE_From                = 411,
ft_DLClassCode_DE_To                  = 412,
ft_DLClassCode_DE_Notes               = 413,
ft_DLClassCode_M_From                 = 414,
ft_DLClassCode_M_To                   = 415,
ft_DLClassCode_M_Notes                = 416,
ft_DLClassCode_L_From                 = 417,
ft_DLClassCode_L_To                   = 418,
ft_DLClassCode_L_Notes                = 419,
ft_DLClassCode_T_From                 = 420,
ft_DLClassCode_T_To                   = 421,
ft_DLClassCode_T_Notes                = 422,
ft_DLClassCode_AM_From                = 423,
ft_DLClassCode_AM_To                  = 424,
ft_DLClassCode_AM_Notes               = 425,
ft_DLClassCode_A2_From                = 426,
ft_DLClassCode_A2_To                  = 427,
ft_DLClassCode_A2_Notes               = 428,
ft_DLClassCode_B1_From                = 429,
ft_DLClassCode_B1_To                  = 430,
ft_DLClassCode_B1_Notes               = 431,
ft_Surname_at_Birth                   = 432,
ft_Civil_Status                       = 433,
ft_Number_of_Seats                    = 434,
ft_Number_of_Standing_Places          = 435,
ft_Max_Speed                          = 436,
ft_Fuel_Type                          = 437,
ft_EC_Environmental_Type              = 438,
ft_Power_Weight_Ratio                 = 439,
ft_Max_Mass_of_Trailer_Braked         = 440,
ft_Max_Mass_of_Trailer_Unbraked       = 441,
ft_Transmission_Type                  = 442,
ft_Trailer_Hitch                      = 443,
ft_Accompanied_by                     = 444,
ft_Police_District                    = 445,
ft_First_Issue_Date                   = 446,
ft_Payload_Capacity                   = 447,
ft_Number_of_Axels                    = 448,
ft_Permissible_Axle_Load              = 449,
ft_Precinct                           = 450,
ft_Invited_by                         = 451,
ft_Purpose_of_Entry                   = 452,
ft_Skin_Color                         = 453,
ft_Complexion                         = 454,
ft_Airport_From                       = 455,
ft_Airport_To                         = 456,
ft_Airline_Name                       = 457,
ft_Airline_Name_Frequent_Flyer        = 458,
ft_License_Number                     = 459,
ft_In_Tanks                           = 460,
ft_Exept_In_Tanks                     = 461,
ft_Fast_Track                         = 462,
ft_Owner                              = 463,
ft_MRZ_Strings_ICAO_RFID              = 464,
ft_Number_of_Card_Issuance            = 465,
ft_Number_of_Card_Issuance_Checksum   = 466,
ft_Number_of_Card_Issuance_CheckDigit = 467,
ft_Century_Date_of_Birth              = 468,
ft_DLClassCode_A3_From                = 469,
ft_DLClassCode_A3_To                  = 470,
ft_DLClassCode_A3_Notes               = 471,
ft_DLClassCode_C2_From                = 472,
ft_DLClassCode_C2_To                  = 473,
ft_DLClassCode_C2_Notes               = 474,
ft_DLClassCode_B2_From                = 475,
ft_DLClassCode_B2_To                  = 476,
```

```
ft_DLClassCode_B2_Notes                = 477,
ft_DLClassCode_D2_From                 = 478,
ft_DLClassCode_D2_To                   = 479,
ft_DLClassCode_D2_Notes                = 480,
ft_DLClassCode_B2E_From                = 481,
ft_DLClassCode_B2E_To                  = 482,
ft_DLClassCode_B2E_Notes               = 483,
ft_DLClassCode_G_From                  = 484,
ft_DLClassCode_G_To                    = 485,
ft_DLClassCode_G_Notes                 = 486,
ft_DLClassCode_J_From                  = 487,
ft_DLClassCode_J_To                    = 488,
ft_DLClassCode_J_Notes                 = 489,
ft_DLClassCode_LC_From                 = 490,
ft_DLClassCode_LC_To                   = 491,
ft_DLClassCode_LC_Notes                = 492,
ft_BankCardNumber                      = 493,
ft_BankCardValidThru                   = 494,
ft_TaxNumber                           = 495,
ft_HealthNumber                        = 496,
ft_GrandfatherName                     = 497,
ft_Selectee_Indicator                  = 498,
ft_Mother_Surname                      = 499,
ft_Mother_GivenName                    = 500,
ft_Father_Surname                      = 501,
ft_Father_GivenName                    = 502,
ft_Mother_DateOfBirth                  = 503,
ft_Father_DateOfBirth                  = 504,
ft_Mother_PersonalNumber               = 505,
ft_Father_PersonalNumber               = 506,
ft_Mother_PlaceOfBirth                 = 507,
ft_Father_PlaceOfBirth                 = 508,
ft_Mother_CountryOfBirth               = 509,
ft_Father_CountryOfBirth               = 510,
ft_Date_First_Renewal                  = 511,
ft_Date_Second_Renewal                 = 512,
ft_PlaceOfExamination                  = 513,
ft_ApplicationNumber                   = 514,
ft_VoucherNumber                       = 515,
ft_AuthorizationNumber                 = 516,
ft_Faculty                             = 517,
ft_FormOfEducation                     = 518,
ft_DNINumber                           = 519,
ft_RetirementNumber                    = 520,
ft_ProfessionalIdNumber                = 521,
ft_Age_at_Issue                        = 522,
ft_Years_Since_Issue                   = 523,
ft_DLClassCode_BTP_From                = 524,
ft_DLClassCode_BTP_Notes               = 525,
ft_DLClassCode_BTP_To                  = 526,
ft_DLClassCode_C3_From                 = 527,
ft_DLClassCode_C3_Notes                = 528,
ft_DLClassCode_C3_To                   = 529,
ft_DLClassCode_E_From                  = 530,
ft_DLClassCode_E_Notes                 = 531,
ft_DLClassCode_E_To                    = 532,
ft_DLClassCode_F_From                  = 533,
ft_DLClassCode_F_Notes                 = 534,
ft_DLClassCode_F_To                    = 535,
ft_DLClassCode_FA_From                 = 536,
ft_DLClassCode_FA_Notes                = 537,
ft_DLClassCode_FA_To                   = 538,
ft_DLClassCode_FA1_From                = 539,
ft_DLClassCode_FA1_Notes               = 540,
ft_DLClassCode_FA1_To                  = 541,
ft_DLClassCode_FB_From                 = 542,
ft_DLClassCode_FB_Notes                = 543,
ft_DLClassCode_FB_To                   = 544,
ft_DLClassCode_G1_From                 = 545,
ft_DLClassCode_G1_Notes                = 546,
ft_DLClassCode_G1_To                   = 547,
ft_DLClassCode_H_From                  = 548,
ft_DLClassCode_H_Notes                 = 549,
ft_DLClassCode_H_To                    = 550,
ft_DLClassCode_I_From                  = 551,
ft_DLClassCode_I_Notes                 = 552,
ft_DLClassCode_I_To                    = 553,
ft_DLClassCode_K_From                  = 554,
ft_DLClassCode_K_Notes                 = 555,
ft_DLClassCode_K_To                    = 556,
ft_DLClassCode_LK_From                 = 557,
ft_DLClassCode_LK_Notes                = 558,
ft_DLClassCode_LK_To                   = 559,
ft_DLClassCode_N_From                  = 560,
```

```
ft_DLClassCode_N_Notes                = 561,
ft_DLClassCode_N_To                   = 562,
ft_DLClassCode_S_From                 = 563,
ft_DLClassCode_S_Notes                = 564,
ft_DLClassCode_S_To                   = 565,
ft_DLClassCode_TB_From                = 566,
ft_DLClassCode_TB_Notes               = 567,
ft_DLClassCode_TB_To                  = 568,
ft_DLClassCode_TM_From                = 569,
ft_DLClassCode_TM_Notes               = 570,
ft_DLClassCode_TM_To                  = 571,
ft_DLClassCode_TR_From                = 572,
ft_DLClassCode_TR_Notes               = 573,
ft_DLClassCode_TR_To                  = 574,
ft_DLClassCode_TV_From                = 575,
ft_DLClassCode_TV_Notes               = 576,
ft_DLClassCode_TV_To                  = 577,
ft_DLClassCode_V_From                 = 578,
ft_DLClassCode_V_Notes                = 579,
ft_DLClassCode_V_To                   = 580,
ft_DLClassCode_W_From                 = 581,
ft_DLClassCode_W_Notes                = 582,
ft_DLClassCode_W_To                   = 583,
ft_URL                                = 584,
ft_Caliber                            = 585,
ft_Model                              = 586,
ft_Make                               = 587,
ft_NumberOfCylinders                  = 588,
ft_SurnameOfHusbandAfterRegistration = 589,
ft_SurnameOfWifeAfterRegistration    = 590,
ft_DateOfBirthOfWife                  = 591,
ft_DateOfBirthOfHusband               = 592,
ft_CitizenshipOfFirstPerson           = 593,
ft_CitizenshipOfSecondPerson          = 594,
ft_CVV                                = 595,
ft_Date_of_Insurance_Expiry          = 596,
ft_Mortgage_by                        = 597,
ft_Old_Document_Number               = 598,
ft_Old_Date_of_Issue                 = 599,
ft_Old_Place_of_Issue                = 600,
ft_DLClassCode_LR_From                = 601,
ft_DLClassCode_LR_To                  = 602,
ft_DLClassCode_LR_Notes               = 603,
ft_DLClassCode_MR_From                = 604,
ft_DLClassCode_MR_To                  = 605,
ft_DLClassCode_MR_Notes               = 606,
ft_DLClassCode_HR_From                = 607,
ft_DLClassCode_HR_To                  = 608,
ft_DLClassCode_HR_Notes               = 609,
ft_DLClassCode_HC_From                = 610,
ft_DLClassCode_HC_To                  = 611,
ft_DLClassCode_HC_Notes               = 612,
ft_DLClassCode_MC_From                = 613,
ft_DLClassCode_MC_To                  = 614,
ft_DLClassCode_MC_Notes               = 615,
ft_DLClassCode_RE_From                = 616,
ft_DLClassCode_RE_To                  = 617,
ft_DLClassCode_RE_Notes               = 618,
ft_DLClassCode_R_From                 = 619,
ft_DLClassCode_R_To                   = 620,
ft_DLClassCode_R_Notes                = 621,
ft_DLClassCode_CA_From                = 622,
ft_DLClassCode_CA_To                  = 623,
ft_DLClassCode_CA_Notes               = 624,
ft_Citizenship_Status                 = 625,
};
```

Логические типы полей, определяемые константами этого перечисления:

ft_Document_Class_Code – код типа документа;

ft_Issuing_State_Code – буквенный код государства выдачи документа в соответствии со стандартом ISO 3166-1 (ICAO doc 9303);

ft_Document_Number – номер документа;

ft_Date_of_Expiry – дата окончания срока действия документа;

ft_Date_of_Issue – дата выдачи документа;

ft_Date_of_Birth – дата рождения;
ft_Place_of_Birth – место рождения;
ft_Personal_Number – персональный номер;
ft_Surname – фамилия;
ft_Given_Names – имя (имена);
ft_Mothers_Name – имя матери;
ft_Nationality – национальность;
ft_Sex – пол;
ft_Height – рост;
ft_Weight – вес;
ft_Eyes_Color – цвет глаз;
ft_Hair_Color – цвет волос
ft_Address – адрес;
ft_Donor – отметка о донорстве;
ft_Social_Security_Number – номер социального страхования;
ft_DL_Class – класс водительского удостоверения (ВУ);
ft_DL_Endorsed – код разрешения для ВУ;
ft_DL_Restriction_Code – код ограничения для ВУ;
ft_DL_Under_21_Date – дата, когда владельцу документа исполняется 21 год;
ft_Authority – орган выдачи документа;
ft_Surname_And_Given_Names – Ф.И.О;
ft_Nationality_Code – буквенный код национальности в соответствии со стандартом ISO 3166-1 (ICAO doc 9303);
ft_Passport_Number – номер паспорта (используется в визах);
ft_Invitation_Number – номер приглашения (используется в визах);
ft_Visa_ID – идентификационный номер визы;
ft_Visa_Class – класс визы;
ft_Visa_SubClass – подкласс визы;
ft_MRZ_String1 – не используется;
ft_MRZ_String2 – не используется;
ft_MRZ_String3 – не используется;
ft_MRZ_Type – тип МСЗ;
ft_Optional_Data – опциональные данные;
ft_Document_Class_Name – полное название класса документа;
ft_Issuing_State_Name – полное название государства выдачи документа;
ft_Place_of_Issue – место выдачи документа;
ft_Document_Number_Checksum – контрольная сумма для номера документа[1];

1

ft_Date_of_Birth_Checksum – контрольная сумма для даты рождения;
ft_Date_of_Expiry_Checksum – контрольная сумма для даты окончания срока действия документа;
ft_Personal_Number_Checksum – контрольная сумма для персонального номера;
ft_FinalChecksum – общая контрольная сумма;
ft_Passport_Number_Checksum – контрольная сумма для номера паспорта (используется в визах);
ft_Invitation_Number_Checksum – контрольная сумма для номера приглашения (используется в визах);
ft_Visa_ID_Checksum – контрольная сумма для идентификационного номера визы;
ft_Surname_And_Given_Names_Checksum – контрольная сумма для Ф.И.О.;
ft_Visa_Valid_Until_Checksum – контрольная сумма для даты окончания действия визы;
ft_Other – иная информация;
ft_MRZ_Strings – строки из МСЗ;
ft_Name_Suffix – суффикс имени;
ft_Name_Prefix – префикс имени;
ft_Date_of_Issue_Checksum – контрольная сумма для даты выдачи документа;
ft_Date_of_Issue_CheckDigit – контрольная цифра для даты выдачи документа[2];
ft_Document_Series – серия документа;
ft_RegCert_RegNumber – номер техпаспорта (ТС);
ft_RegCert_CarModel – модель ТС;
ft_RegCert_CarColor – цвет ТС;
ft_RegCert_BodyNumber – номер кузова ТС;
ft_RegCert_CarType – тип ТС;
ft_RegCert_MaxWeight – разрешенная максимальная масса;
ft_Reg_Cert_Weight – масса ТС без нагрузки;
ft_Address_Area – адрес (область);
ft_Address_State – адрес (район);
ft_Address_Building – адрес (номер дома);
ft_Address_House – адрес (номер корпуса);
ft_Address_Flat – адрес (номер квартиры);
ft_Place_of_Registration – место прописки;
ft_Date_of_Registration – дата прописки;

---

2 Контрольная сумма рассчитывается приложением.

Контрольная цифра берется из МСЗ документа.

ft_Resident_From – дата начала проживания;
ft_Resident_Until – дата окончания проживания;
ft_Authority_Code – код органа выдачи (введен для 2–й страницы паспорта Российской Федерации);
ft_Place_of_Birth_Area – адрес места рождения (область/провинция);
ft_Place_of_Birth_StateCode – адрес места рождения (код государства);
ft_Address_Street – адрес (улица);
ft_Address_City – адрес (город);
ft_Address_Jurisdiction_Code – адрес (код места налогообложения);
ft_Address_Postal_Code – адрес (почтовый индекс);
ft_Document_Number_CheckDigit – контрольная цифра для номера документа;
ft_Date_of_Birth_CheckDigit – контрольная цифра для даты рождения;
ft_Date_of_Expiry_CheckDigit – контрольная цифра для даты окончания срока действия документа;
ft_Personal_Number_CheckDigit – контрольная цифра для персонального номера;
ft_FinalCheckDigit – общая контрольная цифра (для всей МСЗ);
ft_Passport_Number_CheckDigit – контрольная цифра для номера (используется в визах);
ft_Invitation_Number_CheckDigit – контрольная цифра для номера приглашения (для виз);
ft_Visa_ID_CheckDigit – контрольная цифра для идентификационного номера визы;
ft_Surname_And_Given_Names_CheckDigit – контрольная цифра для Ф.И.О.;
ft_Visa_Valid_Until_CheckDigit – контрольная цифра для даты окончания действия визы;
ft_Permit_DL_Class – тип разрешения;
ft_Permit_Date_of_Expiry – дата окончания действия разрешения;
ft_Permit_Identifier – идентификатор разрешения;
ft_Permit_Date_of_Issue – дата выдачи разрешения;
ft_Permit_Restriction_Code – код ограничения для разрешения на вождение;
ft_Permit_Endorsed – код разрешения для разрешения на вождение;
ft_Issue_Timestamp – строка, которая используется для проверки документа в соответствии с базой данных;
ft_Number_of_Duplicates – количество дубликатов;
ft_Medical_Indicator_Codes – медицинские отметки;

ft_Non_Resident_Indicator – отметка об отсутствии гражданства;
ft_Visa_Type – тип визы;
ft_Visa_Valid_From – дата начала действия визы;
ft_Visa_Valid_Until – дата окончания действия визы;
ft_Duration_of_Stay – продолжительность пребывания по визе;
ft_Number_of_Entries – количество въездов по визе;
ft_Day – в дате обозначает день;
ft_Month – в дате обозначает месяц;
ft_Year – в дате обозначает год;
ft_Unique_Customer_Identifier – идентификационный номер;
ft_Commercial_Vehicle_Codes – коммерческий код ТС;
ft_AKA_Date_of_Birth – также известен как (дата рождения);
ft_AKA_Social_Security_Number – также известен как (номер социального страхования);
ft_AKA_Surname – также известен как (фамилия);
ft_AKA_Given_Names – также известен как (имя);
ft_AKA_Name_Suffix – также известен как (суффикс имени);
ft_AKA_Name_Prefix – также известен как (префикс имени);
ft_Mailing_Address_Street – почтовый адрес (улица);
ft_Mailing_Address_City – почтовый адрес (город);
ft_Mailing_Address_Jurisdiction_Code – почтовый адрес (код места налогообложения);
ft_Mailing_Address_Postal_Code – почтовый адрес (индекс);
ft_Audit_Information – номер для проверки валидности водительского удостоверения;
ft_Inventory_Number – инвентарный номер;
ft_Race_Ethnicity – расовая/этническая принадлежность;
ft_Jurisdiction_Vehicle_Class – юридический класс транспортного средства;
ft_Jurisdiction_Endorsement_Code – юридический код разрешения;
ft_Jurisdiction_Restriction_Code – юридический код ограничения;
ft_Family_Name – фамилия и (или) имя при рождении;
ft_Given_Names_RUS – имя (русская транскрипция);
ft_Visa_ID_RUS – идентификационный номер визы (русская транскрипция);
ft_Fathers_Name – отчество/имя отца;
ft_Fathers_Name_RUS – отчество/имя отца (русская транскрипция);
ft_Surname_And_Given_Names_RUS – Ф.И.О. (русская транскрипция);
ft_Place_Of_Birth_RUS – место рождения (русская транскрипция);
ft_Authority_RUS – орган выдачи документа (русская транскрипция);

ft_Issuing_State_Code_Numeric – цифровой код государства выдачи документа в соответствии со стандартом ISO 3166–1 (ICAO doc 9303);

ft_Nationality_Code_Numeric – цифровой код национальности в соответствии со стандартом ISO 3166–1 (ICAO doc 9303);

ft_Engine_Power – мощность двигателя ТС;

ft_Engine_Volume – объем двигателя ТС;

ft_Chassis_Number – номер шасси ТС;

ft_Engine_Number – номер двигателя ТС;

ft_Engine_Model – модель двигателя ТС;

ft_Vehicle_Category – категория ТС;

ft_Identity_Card_Number – номер идентификационной карточки;

ft_Control_No – контрольный номер;

ft_Parrent_s_Given_Names – имена родителей;

ft_Second_Surname – вторая фамилия;

ft_Middle_Name – второе имя;

ft_CurrentDate – зарезервировано для внутреннего использования;

ft_FieldFromMRZ – зарезервировано для внутреннего использования;

ft_RegCert_VIN – идентификационный номер кузова автомобиля;

ft_RegCert_VIN_CheckDigit – контрольная цифра для идентификационного номера кузова автомобиля;

ft_RegCert_VIN_Checksum – контрольная сумма для идентификационного номера кузова автомобиля;

ft_Line1_CheckDigit – контрольная цифра первой строки МСЗ;

ft_Line2_CheckDigit – контрольная цифра второй строки МСЗ;

ft_Line3_CheckDigit – контрольная цифра третьей строки МСЗ;

ft_Line1_Checksum – контрольная сумма первой строки МСЗ;

ft_Line2_Checksum – контрольная сумма второй строки МСЗ;

ft_Line3_Checksum – контрольная сумма третьей строки МСЗ;

ft_RegCert_RegNumber_CheckDigit – контрольная цифра для регистрационного номера автомобиля;

ft_RegCert_RegNumber_Checksum – контрольная сумма для регистрационного номера автомобиля;

ft_RegCert_Vehicle_ITS_Code – код ТС в соответствии с ИТС (Интеллектуальная транспортная система);

ft_Card_Access_Number – номер карты для доступа к RFID–чипу;

ft_Marital_Status – семейное положение;

ft_Company_Name – название компании;
ft_Special_Notes – особые примечания;
ft_Surname_of_Spose – фамилия супруга;
ft_Tracking_Number – номер для отслеживания статуса документа;
ft_Booklet_Number – номер буклета;
ft_Children – дети;
ft_Copy – номер копии;
ft_Serial_Number – серийный номер;
ft_Dossier_Number – номер досье;
ft_AKA_Surname_And_Given_Names – также известен как (Ф.И.О.);
ft_Territorial_Validity – территориальное действие;
ft_MRZ_Strings_With_Correct_CheckSums – МСЗ с правильными контрольными суммами;
ft_DL_CDL_Restriction_Code – код ограничения для коммерческого ВУ;
ft_DL_Under_18_Date – дата, когда владельцу документа исполняется 18 лет;
ft_DL_Record_Created – дата создания записи;
ft_DL_Duplicate_Date – дата создания дубликата;
ft_DL_Iss_Type – тип выданного ВУ;
ft_Military_Book_Number – номер военного билета;
ft_Destination – пункт назначения;
ft_Blood_Group – группа крови;
ft_Sequence_Number – порядковый номер;
ft_RegCert_BodyType – тип кузова;
ft_RegCert_CarMark – марка автомобиля;
ft_Transaction_Number – номер транзакции;
ft_Age – возраст;
ft_Folio_Number – регистрационный номер (документа по книге учёта);
ft_Voter_Key – личный номер избирателя;
ft_Address_Municipality – адрес (муниципалитет);
ft_Address_Location – адрес (населенный пункт);
ft_Section – сфера/сектор;
ft_OCR_Number – OCR номер;
ft_Federal_Elections – федеральные выборы;
ft_Reference_Number – уникальный номер;
ft_Visa_Number – номер визы;
ft_Voter – избиратель;
ft_Previous_Type – тип/номер предыдущего документа;
ft_Status_Date_of_Expiry – дата окончания действия статуса;
ft_Bancnote_Number – номер банкноты;
ft_CSC_Code – код отдела по обслуживанию клиентов (Customer Service Center);

ft_Artistic_Name – псевдоним;
ft_Academic_Title – научное звание;
ft_Address_Country – адрес (страна);
ft_Address_Zipcode – адрес (почтовый индекс);
ft_eID_Residence_Permit1 – сведения о разрешении на постоянное место жительства 1 (поле для eID);
ft_eID_Residence_Permit2 – сведения о разрешении на постоянное место жительства 2 (поле для eID);
ft_eID_PlaceOfBirth_Street – адрес места рождения: улица (поле для eID);
ft_eID_PlaceOfBirth_City – адрес места рождения: город (поле для eID);
ft_eID_PlaceOfBirth_State – адрес места рождения: штат (поле для eID);
ft_eID_PlaceOfBirth_Country – адрес места рождения: страна (поле для eID);
ft_eID_PlaceOfBirth_Zipcode – адрес места рождения: почтовый индекс (поле для eID);
ft_CDL_Class – класс коммерческого ВУ;
ft_DL_Under_19_Date – дата, когда владельцу документа исполняется 19 лет;
ft_Weight_pounds – вес (фунты);
ft_Limited_Duration_Document_Indicator – индикатор ограничения срока действия документа;
ft_Endorsement_Expiration_Date – дата окончания срока разрешения для ВУ;
ft_Revision_Date – дата редакции;
ft_Compliance_Type – тип соответствия;
ft_Family_name_truncation – сокращенная фамилия;
ft_First_name_truncation – сокращенное имя;
ft_Middle_name_truncation – сокращенное второе имя;
ft_Exam_Date – дата экзамена;
ft_Organization – организация;
ft_Department – департамент/отдел;
ft_Pay_Grade – уровень заработной платы;
ft_Rank – ранг/статус/титул;
ft_Benefits_Number – номер, подтверждающий право на получение льгот;
ft_Sponsor_Service – вид вооруженных сил, в которых служит спонсор;
ft_Sponsor_Status – статус спонсора;
ft_Sponsor – спонсор;
ft_Relationship – степень родства;
ft_USCIS – регистрационный номер иностранца Службе гражданства и иммиграции США (US Citizenship and Immigration Services);
ft_Category – категория;
ft_Conditions – условия;
ft_Identifier – идентификатор;

ft_Configuration – конфигурация;
ft_Discretionary_data – опциональные данные;
ft_Line1_Optional_Data – дополнительные данные из строки 1 в МСЗ;
ft_Line2_Optional_Data – дополнительные данные из строки 2 в МСЗ;
ft_Line3_Optional_Data – дополнительные данные из строки 3 в МСЗ;
ft_EQV_Code – EQV–код (защитный код);
ft_ALT_Code – ALT–код;
ft_Binary_Code – бинарный код;
ft_Pseudo_Code – псевдокод;
ft_Fee – плата;
ft_Stamp_Number – номер печати;
ft_SBH_SecurityOptions – параметры защиты биометрических данных;
ft_SBH_IntegrityOptions – параметры целостности биометрических данных;
ft_Date_of_Creation – дата создания записи биометрических данных;
ft_Validity_Period – срок действия записи с биометрическими данными;
ft_Patron_Header_Version – версия заголовка владельца формата биометрических данных;
ft_BDB_Type – тип записи биометрических данных;
ft_Biometric_Type – тип биометрических данных;
ft_Biometric_Subtype – подтип биометрических данных;
ft_Biometric_ProductID – идентификатор биометрических данных;
ft_Biometric_Format_Owner – идентификатор владельца формата биометрических данных;
ft_Biometric_Format_Type – формат биометрических данных;
ft_Phone – номер телефона;
ft_Profession – профессия;
ft_Title – должность;
ft_Personal_Summary – общие персональные данные;
ft_Other_Valid_ID – другой действительный идентификатор;
ft_Custody_Info – данные об опеке;
ft_Other_Name – другое имя;
ft_Observations – дополнительные данные;
ft_Tax – данные о налогах;
ft_Date_of_Personalization – дата персонализации документа;
ft_Personalization_SN – серийный номер персонализации;
ft_OtherPerson_Name – имя другого человека;
ft_PersonToNotify_Date_of_Record – дата создания записи о лицах для извещения в случае непредвиденных ситуаций;

ft_PersonToNotify_Name – имя лица для извещения в случае непредвиденных ситуаций;

ft_PersonToNotify_Phone – номер телефона лица для извещения в случае непредвиденных ситуаций;

ft_PersonToNotify_Address – адрес лица для извещения в случае непредвиденных ситуаций;

ft_DS_Certificate_Issuer – текстовые сведения об организации, выпустившей DS-сертификат;

ft_DS_Certificate_Subject – текстовые сведения об организации, выпустившей документ;

ft_DS_Certificate_ValidFrom – дата начала действия DS-сертификата;

ft_DS_Certificate_ValidTo – дата окончания действия DS-сертификата.

ft_VRC_DataObject_Entry – категория транспортного средства/ограничение/условия из группы данных DG1 приложения *eDL;*

ft_TypeApprovalNumber – номер подтверждения типа;

ft_AdministrativeNumber – административный номер

ft_DocumentDiscriminator – различитель документа

ft_DataDiscriminator – различитель даннных

ft_ISO_Issuer_ID_Number – идентификатор ISO эмитента

ft_GNIB_Number – регистрационный номер Национального бюро по вопросам иммиграции Комиссариата полиции (Garda National);

ft_Dept_Number – номер отдела;

ft_Telex_Code – телеграфный код;

ft_Allergies – аллергия;

ft_Sp_Code – Sp–код;

ft_Court_Code – код ограничения, установленного судом;

ft_Cty – код округа;

ft_Sponsor_SSN – номер социального страхования спонсора;

ft_DoD_Number – индивидуальный идентификационный номер военнослужащего;

ft_MC_Novice_Date – начинающий водитель до... (дата);

ft_DUF_Number – номер DUF (регистрационный номер Директората по иммиграции Норвегии);

ft_AGY – код подразделения дорожной полиции Филиппин;

ft_PNR_Code – PNR–код (номер бронирования);

ft_From_Airport_Code – код аэропорта отправления;

ft_To_Airport_Code – код аэропорта прибытия;

ft_Flight_Number – номер рейса;

ft_Date_of_Flight – дата вылета;

ft_Seat_Number – номер места;

ft_Date_of_Issue_Boarding_Pass – дата выдачи посадочного талона;
ft_CCW_Until – лицензия на скрытое ношение оружия, действительна до...;
ft_Reference_Number_Checksum – контрольная сумма для уникального номера;
ft_Reference_Number_CheckDigit – контрольная цифра для уникального номера;
ft_Room_Number – номер комнаты;
ft_Religion – религия;
ft_RemainderTerm – количество месяцев до срока окончания действия документа;
ft_Electronic_Ticket_Indicator – индикатор электронных билетов;
ft_Free_Baggage_Allowance – норма бесплатного провоза багажа;
ft_Frequent_Flyer_Number – номер постоянного пассажира авиакомпании;
ft_Frequent_Flyer_Airline_Designator – индикатор постоянного пассажира авиакомпании;
ft_Ticket_Number – номер билета;
ft_Airline_Numeric_Code – цифровой код авиакомпании;
ft_Airline_Designator_of_boarding_pass_issuer – код авиакомпании, выпустившей посадочный талон;
ft_CheckIn_Sequence_Number – порядковый номер при регистрации пассажира на рейс;
ft_Compartment_Code – код отсека;
ft_PDF417Codec – кодек для PDF417;
ft_Identity_Card_Number_Checksum – контрольная сумма для номера идентификационной карты;
ft_Identity_Card_Number_CheckDigit – контрольная цифра для номера идентификационной карты;
ft_Veteran – ветеран;
ft_DLClassCode_A1_From – дата начала действия ВУ категории A1;
ft_DLClassCode_A1_To – дата окончания действия ВУ категории A1;
ft_DLClassCode_A1_Notes – ограничения для категории A1 ВУ;
ft_DLClassCode_A_From – дата начала действия ВУ категории A;
ft_DLClassCode_A_To – дата окончания действия ВУ категории A;
ft_DLClassCode_A_Notes – ограничения для категории A ВУ;
ft_DLClassCode_B_From – дата начала действия ВУ категории B;
ft_DLClassCode_B_To – дата окончания действия ВУ категории B;
ft_DLClassCode_B_Notes – ограничения для категории B ВУ;
ft_DLClassCode_C1_From – дата начала действия ВУ категории C1;
ft_DLClassCode_C1_To – дата окончания действия ВУ категории C1;
ft_DLClassCode_C1_Notes – ограничения для категории C1 ВУ;
ft_DLClassCode_C_From – дата начала действия ВУ категории C;
ft_DLClassCode_C_To – дата окончания действия ВУ категории C;

ft_DLClassCode_C_Notes – ограничения для категории C ВУ;
ft_DLClassCode_D1_From – дата начала действия ВУ категории D1;
ft_DLClassCode_D1_To – дата окончания действия ВУ категории D1;
ft_DLClassCode_D1_Notes – ограничения для категории D1 ВУ;
ft_DLClassCode_D_From – дата начала действия ВУ категории D;
ft_DLClassCode_D_To – дата окончания действия ВУ категории D;
ft_DLClassCode_D_Notes – ограничения для категории D ВУ;
ft_DLClassCode_BE_From – дата начала действия ВУ категории BE;
ft_DLClassCode_BE_To – дата окончания действия ВУ категории BE;
ft_DLClassCode_BE_Notes – ограничения для категории BE ВУ;
ft_DLClassCode_C1E_From – дата начала действия ВУ категории C1E;
ft_DLClassCode_C1E_To – дата окончания действия ВУ категории C1E;
ft_DLClassCode_C1E_Notes – ограничения для категории C1E ВУ;
ft_DLClassCode_CE_From – дата начала действия ВУ категории CE;
ft_DLClassCode_CE_To – дата окончания действия ВУ категории CE;
ft_DLClassCode_CE_Notes – ограничения для категории CE ВУ;
ft_DLClassCode_D1E_From – дата начала действия ВУ категории D1E;
ft_DLClassCode_D1E_To – дата окончания действия ВУ категории D1E;
ft_DLClassCode_D1E_Notes – ограничения для категории D1E ВУ;
ft_DLClassCode_DE_From – дата начала действия ВУ категории DE;
ft_DLClassCode_DE_To – дата окончания действия ВУ категории DE;
ft_DLClassCode_DE_Notes – ограничения для категории DE ВУ;
ft_DLClassCode_M_From – дата начала действия ВУ категории M;
ft_DLClassCode_M_To – дата окончания действия ВУ категории M;
ft_DLClassCode_M_Notes – ограничения для категории M ВУ;
ft_DLClassCode_L_From – дата начала действия ВУ категории L;
ft_DLClassCode_L_To – дата окончания действия ВУ категории L;
ft_DLClassCode_L_Notes – ограничения для категории L ВУ;
ft_DLClassCode_T_From – дата начала действия ВУ категории T;
ft_DLClassCode_T_To – дата окончания действия ВУ категории T;
ft_DLClassCode_T_Notes – ограничения для категории T ВУ;
ft_DLClassCode_AM_From – дата начала действия ВУ категории AM;
ft_DLClassCode_AM_To – дата окончания действия ВУ категории AM;
ft_DLClassCode_AM_Notes – ограничения для категории AM ВУ;
ft_DLClassCode_A2_From – дата начала действия ВУ категории A2;
ft_DLClassCode_A2_To – дата окончания действия ВУ категории A2;
ft_DLClassCode_A2_Notes – ограничения для категории A2 ВУ;
ft_DLClassCode_B1_From – дата начала действия ВУ категории B1;
ft_DLClassCode_B1_To – дата окончания действия ВУ категории B1;
ft_DLClassCode_B1_Notes – ограничения для категории B1 ВУ;
ft_Surname_at_Birth – фамилия при рождении;
ft_Civil_Status – гражданский статус;
ft_Number_of_Seats – количество сидячих мест;
ft_Number_of_Standing_Places – количество стоячих мест;

ft_Max_Speed – максимальная скорость;
ft_Fuel_Type – тип топлива;
ft_EC_Environmental_Type – экологический класс автомобиля ;
ft_Power_Weight_Ratio – удельная мощность двигателя (на единицу веса);
ft_Max_Mass_of_Trailer_Braked – максимальная масса прицепа, оборудованного тормозами;
ft_Max_Mass_of_Trailer_Unbraked – максимальная масса прицепа, не оборудованного тормозами;
ft_Transmission_Type – тип трансмиссии;
ft_Trailer_Hitch – прицепное оборудование;
ft_Accompanied_by – сопровождающее лицо;
ft_Police_District – полицейский округ;
ft_First_Issue_Date – дата первой выдачи документа;
ft_Payload_Capacity – максимальная полезная нагрузка;
ft_Number_of_Axels – количество осей;
ft_Permissible_Axle_Load – допустимая нагрузка на ось;
ft_Precinct – избирательный участок;
ft_Invited_by – приглашающее лицо;
ft_Purpose_of_Entry – цель приезда;
ft_Skin_Color – цвет кожи;
ft_Complexion – цвет лица;
ft_Airport_From – аэропорт отправления;
ft_Airport_To – аэропорт прибытия;
ft_Airline_Name – название авиакомпании;
ft_Airline_Name_Frequent_Flyer – бонусная программа авиакомпании для постоянных пассажиров;
ft_License_Number – номер лицензии;
ft_In_Tanks – в цистернах;
ft_Exept_In_Tanks – за исключением цистерн;
ft_Fast_Track – пассажир с приоритетом fast track (экспресс–прохождение официальных процедур в аэропорту);
ft_Owner – владелец;
ft_MRZ_Strings_ICAO_RFID – строки МСЗ из ICAO RFID;
ft_Number_of_Card_Issuance – количество выпусков карточки с таким номером ;
ft_Number_of_Card_Issuance_Checksum – контрольная сумма количества выпусков;
ft_Number_of_Card_Issuance_CheckDigit – контрольная цифра количества выпусков;
ft_Century_Date_of_Birth – век рождения;
ft_DLClassCode_A3_From – дата начала действия ВУ категории А3;
ft_DLClassCode_A3_To – дата окончания действия ВУ категории А3;

ft_DLClassCode_A3_Notes – ограничения для категории A3 ВУ;
ft_DLClassCode_C2_From – дата начала действия ВУ категории C2;
ft_DLClassCode_C2_To – дата окончания действия ВУ категории C2;
ft_DLClassCode_C2_Notes – ограничения для категории C2 ВУ;
ft_DLClassCode_B2_From – дата начала действия ВУ категории B2;
ft_DLClassCode_B2_To – дата окончания действия ВУ категории B2;
ft_DLClassCode_B2_Notes – ограничения для категории B2 ВУ;
ft_DLClassCode_D2_From – дата начала действия ВУ категории D2;
ft_DLClassCode_D2_To – дата окончания действия ВУ категории D2;
ft_DLClassCode_D2_Notes – ограничения для категории D2 ВУ;
ft_DLClassCode_B2E_From – дата начала действия ВУ категории B2E;
ft_DLClassCode_B2E_To – дата окончания действия ВУ категории B2E;
ft_DLClassCode_B2E_Notes ограничения для категории B2E ВУ;
ft_DLClassCode_G_From – дата начала действия ВУ категории G;
ft_DLClassCode_G_To – дата окончания действия ВУ категории G;
ft_DLClassCode_G_Notes – ограничения для категории G ВУ;
ft_DLClassCode_J_From – дата начала действия ВУ категории J;
ft_DLClassCode_J_To – дата окончания действия ВУ категории J;
ft_DLClassCode_J_Notes – ограничения для категории J ВУ;
ft_DLClassCode_LC_From – дата начала действия ВУ категории LC;
ft_DLClassCode_LC_To – дата окончания действия ВУ категории LC;
ft_DLClassCode_LC_Notes – ограничения для категории LC ВУ;
ft_BankCardNumber – номер банковской карты;
ft_BankCardValidThru – срок действия банковской карты;
ft_TaxNumber – номер налогоплательщика;
ft_HealthNumber – номер страхования здоровья;
ft_GrandfatherName – имя дедушки;
ft_Selectee_Indicator – индикатор призывника;
ft_Mother_Surname – фамилия матери;
ft_Mother_GivenName – имя матери;
ft_Father_Surname – фамилия отца;
ft_Father_GivenName – имя отца;
ft_Mother_DateOfBirth – дата рождения матери;
ft_Father_DateOfBirth – дата рождения отца;
ft_Mother_PersonalNumber – личный номер матери;
ft_Father_PersonalNumber – личный номер отца;
ft_Mother_PlaceOfBirth – место рождения матери;
ft_Father_PlaceOfBirth – место рождения отца;
ft_Mother_CountryOfBirth – страна рождения матери;

ft_Father_CountryOfBirth – страна рождения отца;
ft_Date_First_Renewal – дата первого продления;
ft_Date_Second_Renewal – дата второго продления.
ft_PlaceOfExamination – место исследования;
ft_ApplicationNumber – номер заявки;
ft_VoucherNumber – номер ваучера;
ft_AuthorizationNumber – номер авторизации;
ft_Faculty – факультет;
ft_FormOfEducation – форма обучения;
ft_DNINumber – DNI номер;
ft_RetirementNumber – пенсионный номер;
ft_ProfessionalIdNumber – профессиональный идентификационный номер;
ft_Age_at_Issue – возраст при выдаче;
ft_Years_Since_Issue – лет с момента выдачи;
ft_DLClassCode_BTP_From – дата начала действия ВУ категории BTP;
ft_DLClassCode_BTP_Notes – ограничения для ВУ категории BTP;
ft_DLClassCode_BTP_To – дата окончания действия ВУ категории BTP;
ft_DLClassCode_C3_From – дата начала действия ВУ категории C3;
ft_DLClassCode_C3_Notes – ограничения для ВУ категории C3;
ft_DLClassCode_C3_To – дата окончания действия ВУ категории C3;
ft_DLClassCode_E_From – дата начала действия ВУ категории E;
ft_DLClassCode_E_Notes – ограничения для ВУ категории E;
ft_DLClassCode_E_To – дата окончания действия ВУ категории E;
ft_DLClassCode_F_From – дата начала действия ВУ категории F;
ft_DLClassCode_F_Notes – ограничения для ВУ категории F;
ft_DLClassCode_F_To – дата окончания действия ВУ категории F;
ft_DLClassCode_FA_From – дата начала действия ВУ категории FA;
ft_DLClassCode_FA_Notes – ограничения для ВУ категории FA;
ft_DLClassCode_FA_To – дата окончания действия ВУ категории FA;
ft_DLClassCode_FA1_From – дата начала действия ВУ категории FA1;
ft_DLClassCode_FA1_Notes – ограничения для ВУ категории FA1;
ft_DLClassCode_FA1_To – дата окончания действия ВУ категории FA1;
ft_DLClassCode_FB_From – дата начала действия ВУ категории FB;
ft_DLClassCode_FB_Notes – ограничения для ВУ категории FB;
ft_DLClassCode_FB_To – дата окончания действия ВУ категории FB;
ft_DLClassCode_G1_From – дата начала действия ВУ категории G1;
ft_DLClassCode_G1_Notes – ограничения для ВУ категории G1;
ft_DLClassCode_G1_To – дата окончания действия ВУ категории G1;
ft_DLClassCode_H_From – дата начала действия ВУ категории H;
ft_DLClassCode_H_Notes – ограничения для ВУ категории H;
ft_DLClassCode_H_To – дата окончания действия ВУ категории H;
ft_DLClassCode_I_From – дата начала действия ВУ категории I;
ft_DLClassCode_I_Notes – ограничения для ВУ категории I;

ft_DLClassCode_I_To – дата окончания действия ВУ категории I;
ft_DLClassCode_K_From – дата начала действия ВУ категории K;
ft_DLClassCode_K_Notes – ограничения для ВУ категории K;
ft_DLClassCode_K_To – дата окончания действия ВУ категории K;
ft_DLClassCode_LK_From – дата начала действия ВУ категории LK;
ft_DLClassCode_LK_Notes – ограничения для ВУ категории LK;
ft_DLClassCode_LK_To – дата окончания действия ВУ категории LK;
ft_DLClassCode_N_From – дата начала действия ВУ категории N;
ft_DLClassCode_N_Notes – ограничения для ВУ категории N;
ft_DLClassCode_N_To – дата окончания действия ВУ категории N;
ft_DLClassCode_S_From – дата начала действия ВУ категории S;
ft_DLClassCode_S_Notes – ограничения для ВУ категории S;
ft_DLClassCode_S_To – дата окончания действия ВУ категории S;
ft_DLClassCode_TB_From – дата начала действия ВУ категории TB;
ft_DLClassCode_TB_Notes – ограничения для ВУ категории TB;
ft_DLClassCode_TB_To – дата окончания действия ВУ категории TB;
ft_DLClassCode_TM_From – дата начала действия ВУ категории TM;
ft_DLClassCode_TM_Notes – ограничения для ВУ категории TM;
ft_DLClassCode_TM_To – дата окончания действия ВУ категории TM;
ft_DLClassCode_TR_From – дата начала действия ВУ категории TR;
ft_DLClassCode_TR_Notes – ограничения для ВУ категории TR;
ft_DLClassCode_TR_To – дата окончания действия ВУ категории TR;
ft_DLClassCode_TV_From – дата начала действия ВУ категории TV;
ft_DLClassCode_TV_Notes – ограничения для ВУ категории TV;
ft_DLClassCode_TV_To – дата окончания действия ВУ категории TV;
ft_DLClassCode_V_From – дата начала действия ВУ категории V;
ft_DLClassCode_V_Notes – ограничения для ВУ категории V;
ft_DLClassCode_V_To – дата окончания действия ВУ категории V;
ft_DLClassCode_W_From – дата начала действия ВУ категории W;
ft_DLClassCode_W_Notes – ограничения для ВУ категории W;
ft_DLClassCode_W_To – дата окончания действия ВУ категории W;
ft_URL – URL;
ft_Caliber – калибр;
ft_Model – модель;
ft_Make – производитель;
ft_NumberOfCylinders – количество цилиндров;
ft_SurnameOfHusbandAfterRegistration – фамилия мужа после регистрации;
ft_SurnameOfWifeAfterRegistration – фамилия жены после регистрации;
ft_DateOfBirthOfWife – дата рождения жены;
ft_DateOfBirthOfHusband – дата рождения мужа;
ft_CitizenshipOfFirstPerson – гражданство первой личности;
ft_CitizenshipOfSecondPerson – гражданство второй личности;
ft_CVV – CVV;
ft_Date_of_Insurance_Expiry – дата окончания действия страховки;

```
   ft_Mortgage_by                  – ипотека;
   ft_Old_Document_Number          – старый номер документа;
   ft_Old_Date_of_Issue            – старая дата выдачи;
   ft_Old_Place_of_Issue           – старое место выдачи;
   ft_DLClassCode_LR_From          – дата начала действия ВУ категории LR;
   ft_DLClassCode_LR_To            – дата окончания действия ВУ категории LR;
   ft_DLClassCode_LR_Notes         – ограничения для ВУ категории LR;
   ft_DLClassCode_MR_From          – дата начала действия ВУ категории MR;
   ft_DLClassCode_MR_To            – дата окончания действия ВУ категории MR;
   ft_DLClassCode_MR_Notes         – ограничения для ВУ категории MR;
   ft_DLClassCode_HR_From          – дата начала действия ВУ категории HR;
   ft_DLClassCode_HR_To            – дата окончания действия ВУ категории HR;
   ft_DLClassCode_HR_Notes         – ограничения для ВУ категории HR;
   ft_DLClassCode_HC_From          – дата начала действия ВУ категории HC;
   ft_DLClassCode_HC_To            – дата окончания действия ВУ категории HC;
   ft_DLClassCode_HC_Notes         – ограничения для ВУ категории HC;
   ft_DLClassCode_MC_From          – дата начала действия ВУ категории MC;
   ft_DLClassCode_MC_To            – дата окончания действия ВУ категории MC;
   ft_DLClassCode_MC_Notes         – ограничения для ВУ категории MC;
   ft_DLClassCode_RE_From          – дата начала действия ВУ категории RE;
   ft_DLClassCode_RE_To            – дата окончания действия ВУ категории RE;
   ft_DLClassCode_RE_Notes         – ограничения для ВУ категории RE;
   ft_DLClassCode_R_From           – дата начала действия ВУ категории R;
   ft_DLClassCode_R_To             – дата окончания действия ВУ категории R;
   ft_DLClassCode_R_Notes          – ограничения для ВУ категории R;
   ft_DLClassCode_CA_From          – дата начала действия ВУ категории CA;
   ft_DLClassCode_CA_To            – дата окончания действия ВУ категории CA;
   ft_DLClassCode_CA_Notes         – ограничения для ВУ категории CA;
   ft_Citizenship_Status           – гражданство
};
```

5.4.12. eGraphicFieldType

Перечисление **eGraphicFieldType** содержит идентификаторы, определяющие логический тип графических данных, полученных при чтении полей заполнения документа или распознавании штрихкодов.

```
enum eGraphicFieldType
{
   gf_Portrait              = 201,
   gf_Fingerprint           = 202,
   gf_Eye                   = 203,
   gf_Signature             = 204,
   gf_BarCode               = 205,
   gf_Proof_Of_Citizenship  = 206,
   gf_Document_Front        = 207,
   gf_Document_Rear         = 208,
   gf_ColorDynamic          = 209,
   gf_GhostPortrait         = 210,
   gf_Stamp                 = 211,
   gf_Portrait_Of_Child     = 212,
```

```
   gf_Other                = 250,
   gf_Finger_LeftThumb     = 300,
   gf_Finger_LeftIndex     = 301,
   gf_Finger_LeftMiddle    = 302,
   gf_Finger_LeftRing      = 303,
   gf_Finger_LeftLittle    = 304,
   gf_Finger_RightThumb    = 305,
   gf_Finger_RightIndex    = 306,
   gf_Finger_RightMiddle   = 307,
   gf_Finger_RightRing     = 308,
   gf_Finger_RightLittle   = 309
};
```

Логические типы полей, определяемые константами этого перечисления:

- **gf_Portrait** – фотография владельца документа;
- **gf_Fingerprint** – отпечаток пальца владельца документа;
- **gf_Eye** – изображение радужной оболочки глаза владельца документа;
- **gf_Signature** – подпись владельца документа;
- **gf_BarCode** – изображение штрихкода;
- **gf_Proof_Of_Citizenship** – изображение документа, подтверждающего гражданство владельца;
- **gf_Document_Front** – изображение лицевой стороны документа;
- **gf_Document_Rear** – изображение оборотной стороны документа;
- **gf_ColorDynamic** – область с динамическим изменением цвета;
- **gf_GhostPortrait** – дополнительный портрет;
- **gf_Stamp** – штамп;
- **gf_Portrait_Of_Child** - фотография ребенка;
- **gf_Other** – неопределенный тип изображения;
- **gf_Finger_LeftThumb** – отпечаток пальца (большой, левая рука);
- **gf_Finger_LeftIndex** – отпечаток пальца (указательный, левая рука);
- **gf_Finger_LeftMiddle** – отпечаток пальца (средний, левая рука);
- **gf_Finger_LeftRing** – отпечаток пальца (безымянный, левая рука);
- **gf_Finger_LeftLittle** – отпечаток пальца (мизинец, левая рука);
- **gf_Finger_RightThumb** – отпечаток пальца (большой, правая рука);
- **gf_Finger_RightIndex** – отпечаток пальца (указательный, правая рука);
- **gf_Finger_RightMiddle** – отпечаток пальца (средний, правая рука);
- **gf_Finger_RightRing** – отпечаток пальца (безымянный, правая рука);
- **gf_Finger_RightLittle** – отпечаток пальца (мизинец, правая рука).

5.4.13. eBarCodeType

Перечисление **eBarCodeType** содержит идентификаторы, определяющие тип штрихкода.

```
enum eBarCodeType
{
   bct_Unknown     = 0,
   bct_Code128     = 1,
   bct_Code39      = 2,
   bct_EAN8        = 3,
   bct_ITF         = 4,
   bct_PDF417      = 5,
   bct_STF         = 6,
   bct_MTF         = 7,
   bct_IATA        = 8,
   bct_CODABAR     = 9,
   bct_UPCA        = 10,
```

```
   bct_CODE93     = 11,
   bct_UPCE       = 12,
   bct_EAN13      = 13,
   bct_QRCODE     = 14,
   bct_AZTEC      = 15,
   bct_DATAMATRIX = 16,
   bct_ALL_1D     = 17,
   bct_CODE11     = 18,
   eBarCodeType_END
};
```

Типы штрихкодов, определяемые константами:

- **bct_Unknown** – неизвестный тип;
- **bct_Code128** – одномерный штрихкод, ISO 15417 (ANSI/AIMBC4-1999 Code 128);
- **bct_Code39** – одномерный штрихкод, ISO 16388 (ANSI/AIM BC1-1995 Code 39);
- **bct_EAN8** – одномерный штрихкод EAN8, ISO 15418;
- **bct_ITF** – одномерный штрихкод «Interleaved 2 of 5», ISO 16390 (ANSI/AIM BC2-1995 Interleaved 2 of 5);
- **bct_PDF417** – двумерный штрихкод, ISO 15438 (AIM USS PDF417);
- **bct_STF** – одномерный штрихкод «Standard 2 of 5» (Industrial);
- **bct_MTF** – одномерный штрихкод «Matrix 2 of 5»;
- **bct_IATA** – одномерный штрихкод «IATA 2 of 5» (Airline);
- **bct_CODABAR** – одномерный штрихкод (ANSI/AIM BC3-1995, USS – Codabar);
- **bct_UPCA** – одномерный штрихкод UPC-A;
- **bct_CODE93** – одномерный штрихкод (ANSI/AIM BC5-1995, USS – Code 93);
- **bct_UPCE** – одномерный штрихкод UPC-E;
- **bct_EAN13** – одномерный штрихкод EAN13, ISO 15418;
- **bct_QRCODE** – двумерный штрихкод QRCODE;
- **bct_AZTEC** – двумерный штрихкод AZTEC;
- **bct_DATAMATRIX** – двумерный штрихкод DATAMATRIX;
- **bct_ALL_1D** – тип для внутреннего использования;
- **bct_CODE11 -** одномерный штрихкод CODE11;
- **eBarCodeType_END -** для внутреннего использования.

5.4.14. eBarCodeResultCodes

Перечисление **eBarCodeResultCodes** содержит идентификаторы, определяющие результат чтения данных штрихкода.

```
enum eBarCodeResultCodes
{
   bcrc_NoErr                          = 0,
   bcrc_NullPtrErr                     = -6001,
   bcrc_BadArgErr                      = -6002,
   bcrc_SizeErr                        = -6003,
   bcrc_RangeErr                       = -6004,
   bcrc_InternalErr                    = -6005,
   bcrc_TryExceptErr                   = -6006,
   bcrc_BarCodeNotFound                = -6008,
   bcrc_BarCodeDecodeErr               = -6010,
   bcrc_NoUserDLLFound                 = -6019,
   bcrc_NoIPPDLLFound                  = -6020,
   bcrc_IppExecErr                     = -6024,
   bcrc_IppTryExceptErr                = -6025,
```

```
   bcrc_BARCODE_ERROR_Inputparam          = -11001,
   bcrc_BARCODE_ERROR_FInit               = -11006,
   bcrc_BARCODE_ERROR_NotLoadIpDecodedll  = -11012,
   bcrc_BARCODE_ERROR_InnerProblem        = -11100,
   bcrc_BARCODE_ERROR_Decode_1D_BadDecode = -11200,
   bcrc_BARCODE_ERROR_FindRowOrColumn     = -11201,
   bcrc_BARCODE_ERROR_Find3X8_2D_X        = -11202,
   bcrc_BARCODE_ERROR_Find3X8_2D_Y        = -11203,
   bcrc_BARCODE_ERROR_2D_UgolMax          = -11204,
   bcrc_BARCODE_ERROR_INDEFINITELY_DECODED = -11210,
   bcrc_BARCODE_ERROR_Dllnotinit          = -11300,
   bcrc_BARCODE_ERROR_IPDECODE_DLL_Try_Except = -11400,
   bcrc_IPDECODE_ERROR_LARGEERRORS        = -4503,
   bcrc_IPDECODE_ERROR_FAULTCOLUMNS       = -4504,
   bcrc_IPDECODE_ERROR_FAULTROWS          = -4505,
   bcrc_IPDECODE_ERROR_INCORRECT_ERROR_LEVEL = -4511,
   bcrc_IPDECODE_ERROR_LOADING_DEV_TABLE  = -4512
};
```

Значение констант:

- **bcrc_NoErr** Нет ошибок

ошибки RegCodes.dll:

- **bcrc_NullPtrErr** Нулевые параметры вызова функции
- **bcrc_BadArgErr** Ошибочные параметры вызова функции
- **bcrc_SizeErr** Ошибочная длина данных
- **bcrc_RangeErr** Ошибочный диапазон параметров
- **bcrc_InternalErr** Внутренняя ошибка
- **bcrc_TryExceptErr** Срабатывание блока `try-except`
- **bcrc_BarCodeNotFound**

Ошибка нахождения штрихкода

- **bcrc_BarCodeDecodeErr**

Ошибка раскодирования штрихкода

- **bcrc_NoUserDLLFound** Ошибка подключения `ImageProcess.dll`
- **bcrc_NoIPPDLLFound** Ошибка подключения `IPP Dll`
- **bcrc_IppExecErr** Ошибка выполнения функции `IPP`
- **bcrc_IppTryExceptErr** Срабатывание блока `try-except` в функции `IPP`

ошибки ImagProcess.dll:

- **bcrc_BARCODE_ERROR_Inputparam** Ошибка входных данных

- **bcrc_BARCODE_ERROR_FInit** Ошибка инициализации

- **bcrc_BARCODE_ERROR_NotLoadIpDecodedll** Ошибка подключения `IpDecode.dll`

- **bcrc_BARCODE_ERROR_InnerProblem** Внутренняя ошибка программы

- **bcrc_BARCODE_ERROR_Decode_1D_BadDecode**
Ошибка раскодирования одномерного штрихкода

- **bcrc_BARCODE_ERROR_FindRowOrColumn** Ошибка вычисления количества строк или столбцов (`PDF417`)

- **bcrc_BARCODE_ERROR_Find3X8_2D_X** Ошибка вычисления `MinX` (`PDF417`)

- **bcrc_BARCODE_ERROR_Find3X8_2D_Y** Ошибка вычисления `MinY` (`PDF417`)

- **bcrc_BARCODE_ERROR_2D_UgolMax** Некорректный угол перекоса штрихкода (> 3 градусов)

- **bcrc_BARCODE_ERROR_INDEFINITELY_DECODED** Результат может содержать ошибки раскодирования

- **bcrc_BARCODE_ERROR_Dllnotinit** Ошибка инициализации `Dll`

- **bcrc_BARCODE_ERROR_IPDECODE_DLL_Try_Except** Сработка блока `try-except` в функции `IPDECODE`

ошибки `IPDecode.dll`:
- **bcrc_IPDECODE_ERROR_LARGEERRORS** Слишком много ошибочных кодовых слов

- **bcrc_IPDECODE_ERROR_FAULTCOLUMNS** Определено неверное число столбцов

- **bcrc_IPDECODE_ERROR_FAULTROWS** Определено неверное число строк

- **bcrc_IPDECODE_ERROR_INCORRECT_ERROR_LEVEL** Ошибочный уровень коррекции ошибок

- **bcrc_IPDECODE_ERROR_LOADING_DEV_TABLE** Ошибка загрузки `DevTable.bin`

5.4.15. eBarCodeModuleType

Перечисление **eBarCodeModuleType** содержит идентификаторы, определяющие тип модуля штрихкода.

```
enum eBarCodeModuleType
{
  MODULETYPE_TEXT  = 0,
  MODULETYPE_BYTE  = 1,
  MODULETYPE_NUM   = 2,
  MODULETYPE_SHIFT = 3,
  MODULETYPE_ALL   = 4
```

```
};
```

Значения констант:

- **MODULETYPE_TEXT** – модуль содержит текстовые данные;
- **MODULETYPE_BYTE** – модуль содержит байтовые данные;
- **MODULETYPE_NUM** – модуль содержит числовые данные;
- **MODULETYPE_SHIFT** – буфер расположения позиций «Shift in byte compaction mode» в коде PDF417;
- **MODULETYPE_ALL** – модуль содержит любые данные.

5.4.16. eTestTextField

Перечисление **eTestTextField** содержит идентификаторы, определяющие результат проверки правильности заполнения текстового поля МСЗ документа.

```
enum eTestTextField
{
  tr_OK                          = 0x00000000,
  tr_Process_Error               = 0x00000001,
  tr_Invalid_Checksum            = 0x00000002,
  tr_Syntax_Error                = 0x00000004,
  tr_Logic_Error                 = 0x00000008,
  tr_SourcesComparison_Error     = 0x00000010,
  tr_FieldsComparison_LogicError = 0x00000020,
  tr_UnknownStatus               = 0x00000040,
  tr_GlaresOnField               = 0x00000080
};
```

Значение констант:

- **tr_OK** – нет ошибок заполнения;
- **tr_Process_Error** – ошибка выполнения проверки;
- **tr_Invalid_Checksum** – неверная контрольная сумма. Используется только для полей «контрольная цифра» (например, **ft_Document_Number_CheckDigit**). Поле, для которого рассчитывается контрольная сумма, определяется соответствующим типом, т. е. для **ft_Document_Number_CheckDigit** это будет поле типа **ft_Document_ Number**;
- **tr_Syntax_Error** – синтаксическая ошибка (например, использование недопустимого символа);
- **tr_Logic_Error** – логическая ошибка (неправильный формат заполнения поля, срок действия документа закончился и т. п.);
- **tr_SourcesComparison_Error** – зарезервировано для внутреннего использования;
- **tr_FieldsComparison_LogicError** – зарезервировано для внутреннего использования;

- **tr_UnknownStatus** – зарезервировано для внутреннего использования;
- **tr_GlaresOnField** – блики на поле.

5.4.17. eMRZClassQuality

Перечисление **eMRZClassQuality** содержит идентификаторы, определяющие параметры проверки качества печати МСЗ документа.

```
enum eMRZClassQuality
{
   mrz_CLASS_QUALITY_X          = 21,
   mrz_CLASS_QUALITY_Y          = 22,
   mrz_CLASS_QUALITY_Z          = 23,
   mrz_CLASS_QUALITY_CUSTOM     = 24
};
```

Установка параметров проверок осуществляется для различных классов качества печати в соответствии со стандартом *ISO 1831:1980* «*Характеристики печатного изображения для оптического распознавания символов*»:

- **mrz_CLASS_QUALITY_X** – параметры проверок для класса качества печати *X*;
- **mrz_CLASS_QUALITY_Y** – параметры проверок для класса качества печати *Y*;
- **mrz_CLASS_QUALITY_Z** – параметры проверок для класса качества печати *Z*;
- **mrz_CLASS_QUALITY_CUSTOM** – параметры проверок по умолчанию.

5.4.18. eCheckResult

Перечисление **eCheckResult** содержит идентификаторы, определяющие результат той или иной проверки документа.

```
enum eMRZCheckResult
{
   ch_Check_Error       =  0,
   ch_Check_OK          =  1,
   ch_Check_WasNotDone  =  2
};
```

Значение констант:

- **ch_Check_Error** – проверка не прошла, контролируемый параметр не в допуске;
- **ch_Check_OK** – проверка прошла, контролируемый параметр в допуске;
- **ch_Check_WasNotDone** – проверка не выполнялась.

5.4.19. ePhotoEmbedType

Перечисление **ePhotoEmbedType** содержит идентификаторы, определяющие технологию нанесения фото в документе.

```
enum ePhotoEmbedType
{
```

```
   PhotoEmbedType_AnyType     = 0,
   PhotoEmbedType_strPrinted  = 1,
   PhotoEmbedType_strSticked  = 2
};
```

Значение констант:

- **PhotoEmbedType_AnyType** – фото может быть как напечатано, так и наклеено;
- **PhotoEmbedType_strPrinted** – фото напечатано;
- **PhotoEmbedType_strSticked** – фото наклеено.

5.4.20. eCheckDiagnose

Перечисление **eCheckDiagnose** содержит идентификаторы, определяющие результат сравнения и анализа текстовых полей из различных источников.

```
enum eCheckDiagnose
{
   chd_Unknown                                = 0,
   chd_Pass                                   = 1,
   chd_InvalidInputData                       = 2,
   chd_InternalError                          = 3,
   chd_ExceptionInModule                      = 4,
   chd_UncertainVerification                  = 5,
   chd_NecessaryImageNotFound                 = 7,
   chd_PhotoSidesNotFound                     = 8,
   chd_InvalidChecksum                        = 10,
   chd_SyntaxError                            = 11,
   chd_LogicError                             = 12,
   chd_SourcesComparisonError                 = 13,
   chd_FieldsComparisonLogicError             = 14,
   chd_InvalidFieldFormat                     = 15,
   chd_TrueLuminiscenceError                  = 20,
   chd_FalseLuminiscenceError                 = 21,
   chd_FixedPatternError                      = 22,
   chd_LowContrastInIRLight                   = 23,
   chd_IncorrectBackgroundLight               = 24,
   chd_BackgroundComparisonError              = 25,
   chd_IncorrectTextColor                     = 26,
   chd_PhotoFalseLuminiscence                 = 27,
   chd_TooMuchShift                           = 28,
   chd_FibersNotFound                         = 30,
   chd_TooManyObjects                         = 31,
   chd_SpecksInUV                             = 33,
   chd_TooLowResolution                       = 34,
   chd_InvisibleElementPresent                = 40,
   chd_VisibleElementAbsent                   = 41,
   chd_ElementShouldBeColored                 = 42,
   chd_ElementShouldBeGrayscale               = 43,
   chd_UVDullPaper_MRZ                        = 50,
   chd_FalseLuminiscenceInMRZ                 = 51,
   chd_UVDullPaper_Photo                      = 52,
   chd_UVDullPaper_Blank                      = 53,
   chd_UVDullPaperError                       = 54,
   chd_FalseLuminiscenceInBlank               = 55,
   chd_BadAreaInAxial                         = 60,
   chd_FalseIPIParameters                     = 65,
   chd_FieldPosCorrector_Highlight_IR         = 80,
   chd_FieldPosCorrector_GlaresInPhotoArea    = 81,
   chd_OVI_IR_Invisible                       = 90,
   chd_OVI_InsufficientArea                   = 91,
   chd_OVI_ColorInvariable                    = 92,
   chd_OVI_BadColor_Front                     = 93,
   chd_OVI_BadColor_Side                      = 94,
   chd_OVI_Wide_Color_Spread                  = 95,
   chd_OVI_BadColor_Percent                   = 96,
   chd_HologramElementAbsent                  = 100,
   chd_Hologram_Side_Top_Images_Absent        = 101,
```

```
  chd_HologramElementPresent                = 102,
  chd_PhotoPattern_Interrupted              = 110,
  chd_PhotoPattern_Shifted                  = 111,
  chd_PhotoPattern_DifferentColors          = 112,
  chd_PhotoPattern_IR_Visible               = 113,
  chd_PhotoPattern_Not_Intersect            = 114,
  chd_PhotoSize_Is_Wrong                    = 115,
  chd_PhotoPattern_InvalidColor             = 116,
  chd_PhotoPattern_Shifted_Vert             = 117,
  chd_PhotoPattern_PatternNotFound          = 118,
  chd_PhotoPattern_DifferentLinesThickness = 119,
  chd_Photo_IsNot_Rectangle                 = 120,
  chd_Photo_Corners_Is_Wrong                = 121,
  chd_DocumentIsCancelling                  = 122,
  chd_TextColorShouldBeBlue                 = 130,
  chd_TextColorShouldBeGreen                = 131,
  chd_TextColorShouldBeRed                  = 132,
  chd_TextShouldBeBlack                     = 133,
  chd_BarcodeWasReadWithErrors              = 140,
  chd_BarcodeDataFormatError                = 141,
  chd_BarcodeSizeParamsError                = 142,
  chd_NotAllBarcodesRead                    = 143,
  chd_PortraitComparison_PortraitsDiffer    = 150,
  chd_PortraitComparison_NoServiceReply     = 151,
  chd_PortraitComparison_ServiceError       = 152,
  chd_PortraitComparison_NotEnoughImages    = 153,
  chd_PortraitComparison_NoLivePhoto        = 154,
  chd_PortraitComparison_NoServiceLicense   = 155,
  chd_PortraitComparison_NoPortraitDetected= 156,
  chd_MobileImages_UnsuitableLightConditions = 160,
  chd_MobileImages_WhiteUVNoDifference      = 161,
  chd_FingerprintsComparison_Mismatch       = 170,
  chd_HoloPhoto_FaceNotDetected             = 180,
  chd_HoloPhoto_FaceComparisonFailed        = 181,
  chd_HoloPhoto_GlareInCenterAbsent         = 182,
  chd_HoloPhoto_HoloElementShapeError       = 183,
  chd_HoloPhoto_AlgorithmStepsError         = 184,
  chd_HoloPhoto_HoloAreasNotLoaded          = 185,
  chd_HoloPhoto_FinishedByTimeout           = 186,
  chd_LastDiagnoseValue                     = 190
};
```

Значение констант:

- **chd_Unknown** – проверка не проводилась;
- **chd_Pass** – проверка прошла успешно;
- **chd_InvalidInputData** – некорректные входные данные;
- **chd_InternalError** – внутренняя ошибка модуля;
- **chd_ExceptionInModule** – поймано исключение;
- **chd_UncertainVerification** – невозможно принять достоверное решение;
- **chd_NecessaryImageNotFound** – не найдено изображение в нужном свете;
- **chd_PhotoSidesNotFound** – не найдена нужная сторона фотографии;
- **chd_InvalidChecksum** – неверная контрольная сумма;
- **chd_SyntaxError** – синтаксическая ошибка;
- **chd_LogicError** – логическая ошибка (например, дата выдачи документа больше, чем текущая дата);
- **chd_SourcesComparisonError** – несовпадение данных из различных источников;
- **chd_FieldsComparisonLogicError** – логическая ошибка проверки данных;
- **chd_InvalidFieldFormat** – неверный формат поля;
- **chd_TrueLuminiscenceError** – элемент люминесценции в УФ не соответствует эталону;
- **chd_FalseLuminiscenceError** – наличие лишней люминесценции в УФ;

- **chd_FixedPatternError** – шаблон не соответствует эталону;
- **chd_LowContrastInIRLight** – низкий контраст объекта в проходящем ИК;
- **chd_IncorrectBackgroundLight** – фон страницы слишком яркий или другого цвета;
- **chd_BackgroundComparisonError** – яркость фона двух страниц различается;
- **chd_IncorrectTextColor** – неверный цвет люминесценции текста в УФ;
- **chd_PhotoFalseLuminiscence** – некорректная люминесценция фотографии;
- **chd_TooMuchShift** – объект сильно смещен относительно стандартных координат;
- **chd_FibersNotFound** – не найдены защитные волокна в УФ;
- **chd_TooManyObjects** – ошибка нахождения волокон;
- **chd_SpecksInUV** – блики или засветка в УФ;
- **chd_TooLowResolution** – слишком низкое разрешение для поиска волокон;
- **chd_InvisibleElementPresent** – ошибочная видимость элемента в ИК;
- **chd_VisibleElementAbsent** – элемент отсутствует в ИК;
- **chd_ElementShouldBeColored** – элемент должен быть цветным;
- **chd_ElementShouldBeGrayscale** – элемент должен быть в градациях серого;
- **chd_UVDullPaper_MRZ** – свечение бумаги в МСЗ;
- **chd_FalseLuminiscenceInMRZ** – люминесценция символов в МСЗ;
- **chd_UVDullPaper_Photo** – свечение бумаги в области фото;
- **chd_UVDullPaper_Blank** – свечение бумаги бланка;
- **chd_UVDullPaperError** – свечение документа в УФ;
- **chd_FalseLuminiscenceInBlank** – элемент бланка люминесцирует;
- **chd_BadAreaInAxial** – нарушение ретрорефлективной защиты;
- **chd_FalseIPIParameters** – некорректные параметры для проверки IPI;
- **chd_FieldPosCorrector_Highlight_IR** – слишком яркое изображение в ИК;
- **chd_FieldPosCorrector_GlaresInPhotoArea** блики в зоне фото
- **chd_OVI_IR_Invisible** – объект OVI не виден в ИК;
- **chd_OVI_InsufficientArea** – недостаточная площадь объекта OVI;
- **chd_OVI_ColorInvariable** – цвет объекта OVI не меняется;
- **chd_OVI_BadColor_Front** – невозможно определить цвет на коаксиальном изображении;
- **chd_OVI_BadColor_Side** – невозможно определить цвет на белом изображении;
- **chd_OVI_Wide_Color_Spread** – большая разбежка по цвету;
- **chd_OVI_BadColor_Percent** – недостаточно информации о цвете;
- **chd_HologramElementAbsent** – отсутствует голограмма;
- **chd_Hologram_Side_Top_Images_Absent**

– нет бокового и верхнего изображений. Проверка отменена;

- **chd_HologramElementPresent** – присутствует голограмма;
- **chd_PhotoPattern_Interrupted** – паттерн прерывается;
- **chd_PhotoPattern_Shifted** – некоторые паттерны смещены относительно друг друга;
- **chd_PhotoPattern_DifferentColors** – некоторые части паттерна имеют другой цвет;
- **chd_PhotoPattern_IR_Visible** – паттерн виден в ИК;
- **chd_PhotoPattern_Not_Intersect** – край фотографии не пересекается с паттерном. Проверка отменена;
- **chd_PhotoSize_Is_Wrong** – размер фотографии не соответствует требованиям;
- **chd_PhotoPattern_InvalidColor** – некоторые части паттерна не того цвета;
- **chd_PhotoSize_Shifted_Vert** – некоторые паттерны смещены относительно друг друга;
- **chd_PhotoPattern_PatternNotFound** – паттерн не найден. Проверка отменена;
- **chd_PhotoPattern_DifferentLinesThickness** – различная толщина линий;
- **chd_Photo_IsNot_Rectangle** – фотограяия не прямоугольной формы;
- **chd_Photo_Corners_Is_Wrong** – углы фотографии не соответствуют требованиям;
- **chd_DocumentIsCancelling** – для внутреннего использования;
- **chd_TextColorShouldBeBlue** – текст должен быть синим;
- **chd_TextColorShouldBeGreen** – текст должен быть зеленым;
- **chd_TextColorShouldBeRed** – текст должен быть красным;
- **chd_TextShouldBeBlack** – текст должен быть черным;
- **chd_BarcodeWasReadWithErrors** – штрихкод прочитан с ошибками;
- **chd_BarcodeDataFormatError** – ошибка в формате данных штрихкода;
- **chd_BarcodeSizeParamsError** – ошибка в формате размера штрихкода;
- **chd_NotAllBarcodesRead** – не все штрихкоды прочитаны;
- **chd_PortraitComparison_PortraitsDiffer** – портреты различаются;
- **chd_PortraitComparison_NoServiceReply** – нет ответа от сервиса сравнения лиц;
- **chd_PortraitComparison_ServiceError** – ошибка сервиса сравнения лиц;
- **chd_PortraitComparison_NotEnoughImages** – недостаточно изображений;

- **chd_PortraitComparison_NoLivePhoto** – нет изображения с камеры;
- **chd_PortraitComparison_NoServiceLicense** – отсутствует лицензия на сервисе;
- **chd_PortraitComparison_NoPortraitDetected** – портреты не найдены;
- **chd_MobileImages_UnsuitableLightConditions** – неподходящие условия освещения;
- **chd_MobileImages_WhiteUVNoDifference** – нет различий между видимым и УФ изображением. Возможная неисправность УФ фонаря;
- **chd_FingerprintsComparison_Mismatch** – отпечатки пальцев не совпадают;
- **chd_HoloPhoto_FaceNotDetected** – лицо не найдено;
- **chd_HoloPhoto_FaceComparisonFailed** – сравнение лиц не выдержано;
- **chd_HoloPhoto_GlareInCenterAbsent** – отсутствует блик посередине;
- **chd_HoloPhoto_HoloElementShapeError** - ошибка формы элемента голограммы;
- **chd_HoloPhoto_AlgorithmStepsError** - ошибка шагов алгоритма;
- **chd_HoloPhoto_HoloAreasNotLoaded** - не загружены области голограммы;
- **chd_HoloPhoto_FinishedByTimeout** – анализ голограммы на фотографии завершен по таймауту;
- **chd_LastDiagnoseValue –** для внутреннего использования.

5.4.21. eRPRM_PostCalbackAction

Перечисление **eRPRM_PostCalbackAction** содержит идентификаторы, определяющие варианты продолжения процедуры чтения полей заполнения документа после возврата из callback-функции, в которой пользовательскому приложению были переданы кандидаты распознавания типа сканируемого документа.

```
enum eRPRM_PostCalbackAction
{
  RPRM_PostCalbackAction_Continue          = 0,
  RPRM_PostCalbackAction_Cancel            = 1,
  RPRM_PostCalbackAction_ProcessCandidate  = 2
};
```

Значение констант:

- **RPRM_PostCalbackAction_Continue** – продолжать чтение данных для документа, который определяется первым элементом из списка;
- **RPRM_PostCalbackAction_Cancel** – прервать дальнейшие операции, требующие определения типа документа;
- **RPRM_PostCalbackAction_ProcessCandidate** – продолжать чтение данных для документа, который определяется

элементом из списка кандидатов с индексом, содержащимся в параметре **PostActionParameter** callback-функции **ResultReceivingFunc**.

5.4.22. eRPRM_RCTP_Result_RecType

Перечисление **eRPRM_RCTP_Result_RecType** содержит идентификаторы, определяющие результат определения типа документа.

```
enum eRPRM_RCTP_Result_RecType
{
   RPRM_RCTP_Result_Ok                 =  0,
   RPRM_RCTP_Result_RecognClassConflict =  14,
   RPRM_RCTP_Result_RecognClassUnknown =  15
   RPRM_RCTP_Result_Need_Other_Image    =  29
};
```

Значение констант:

- **RPRM_RCTP_Result_Ok** – тип документа определен, и первый элемент списка кандидатов распознавания типа документа содержит результат;
- **RPRM_RCTP_Result_RecognClassConflict** – тип документа не определен, и пользовательское приложение должно произвести выбор одного из кандидатов;
- **RPRM_RCTP_Result_RecognClassUnknown** – тип документа не определен, и дальнейшая операция чтения данных невозможна.
- **RPRM_RCTP_Result_Need_Other_Image** – нужны дополнительные изображения для определения типа документа

5.4.23. eRFID_Presence

Перечисление **eRFID_Presence** содержит идентификаторы, определяющие наличие и расположение RFID-микросхемы в документе определенного типа.

```
enum eRFID_Presence
{
   rfpNone     = 0,
   rfpMainPage = 1,
   rfpBackPage = 2
};
```

Значение констант:

- **rfpNone** – в документе нет RFID-микросхемы;
- **rfpMainPage** – RFID-микросхема расположена в странице данных документа;
- **rfpBackPage** – RFID-микросхема расположена в заднем форзаце документа.

5.4.24. eRPRM_Authenticity

Перечисление **eRPRM_Authenticity** содержит идентификаторы, определяющие возможность проведения той или иной процедуры контроля подлинности документа по изображениям для определенных схем освещения.

```
enum eRPRM_Authenticity
{
   RPRM_Authenticity_None                 = 0x00000000,
   RPRM_Authenticity_UV_Luminescence      = 0x00000001,
   RPRM_Authenticity_IR_B900              = 0x00000002,
   RPRM_Authenticity_Image_Pattern        = 0x00000004,
   RPRM_Authenticity_Axial_Protection     = 0x00000008,
   RPRM_Authenticity_UV_Fibers            = 0x00000010,
   RPRM_Authenticity_IR_Visibility        = 0x00000020,
   RPRM_Authenticity_OCRSecurityText      = 0x00000040,
   RPRM_Authenticity_IPI                  = 0x00000080,
   RPRM_Authenticity_IR_Photo             = 0x00000100,
   RPRM_Authenticity_Photo_Embed_Type     = 0x00000200,
   RPRM_Authenticity_OVI                  = 0x00000400,
   RPRM_Authenticity_IR_Luminescence      = 0x00000800,
   RPRM_Authenticity_Holograms            = 0x00001000,
   RPRM_Authenticity_Photo_Area           = 0x00002000,
   RPRM_Authenticity_UV_Background        = 0x00004000,
   RPRM_Authenticity_Portrait_Comparison  = 0x00008000,
   RPRM_Authenticity_BarcodeFormatCheck   = 0x000010000,
   RPRM_Authenticity_Kinegram             = 0x00020000,
   RPRM_Authenticity_Letter_Screen        = 0x00040000,
   RPRM_Authenticity_Holograms_Detection  = 0x00080000,
   RPRM_Authenticity_Fingerprint_Comparison= 0x00100000,
   RPRM_Authenticity_CancellingDocumentDetector
                                          = 0x00200000,
   RPRM_Authenticity_UV                   =
     RPRM_Authenticity_UV_Luminescence |
   RPRM_Authenticity_Image_Pattern      | RPRM_Authenticity_UV_Fibers,
};
```

Значение констант:

- **RPRM_Authenticity_None** Для данного документа проведение процедуры контроля подлинности не предусмотрено

- **RPRM_Authenticity_UV_Luminescence** Для данного документа предусмотрено проведение процедуры контроля УФ-люминесценции материала документа

• **RPRM_Authenticity_IR_B900** » контроля контраста МСЗ по изображению для схемы ИК-освещения

• **RPRM_Authenticity_Image_Pattern** » контроля наличия люминесцирующих объектов на изображении для схемы УФ-освещения

• **RPRM_Authenticity_Axial_Protection** » контроля подлинности по изображениям для схемы белого коаксиального освещения

• **RPRM_Authenticity_UV_Fibers** » контроля УФ-люминесценции защитных волокон

• **RPRM_Authenticity_IR_Visibility** » контроля видимости/невидимости элементов бланка для схемы ИК-освещения

• **RPRM_Authenticity_OCRSecurityText** » OCR скрытого текста и его сравнения с заданным источником аналогичной текстовой информации

• **RPRM_Authenticity_IPI** » визуализации внедренных скрытых изображений

• **RPRM_Authenticity_IR_Photo** Зарезервировано для внутренних нужд

• **RPRM_Authenticity_Photo_Embed_Type** Для данного документа предусмотрено проведение процедуры проверки наклеена фотография или напечатана

• **RPRM_Authenticity_OVI** Зарезервировано для внутренних нужд

• **RPRM_Authenticity_IR_Luminescence** Зарезервировано для внутренних нужд

• **RPRM_Authenticity_Holograms** Зарезервировано для внутренних нужд

• **RPRM_Authenticity_Photo_Area** Зарезервировано для внутренних нужд

• **RPRM_Authenticity_UV_Background** Зарезервировано для внутренних нужд

• **RPRM_Authenticity_Portrait_Comparison** Сравнения портретов

• **RPRM_Authenticity_BarcodeFormatCheck** Проверки формата штрихкода

• **RPRM_Authenticity_Kinegram** Kinegram

- **RPRM_Authenticity_Letter_Screen** LetterScreen

- **RPRM_Authenticity_Fingerprint_Comparison** Сравнение отпечатков пальцев

- **RPRM_Authenticity_CancellingDocumentDetector**
Зарезервировано для внутренних нужд

- **RPRM_Authenticity_Holograms_Detection**
Зарезервировано для внутренних нужд

5.4.25. eRPRM_SecurityFeatureType

Перечисление **eRPRM_SecurityFeatureType** содержит идентификаторы, определяющие типы элементов проверки подлинности документа.

```
enum eRPRM_SecurityFeatureType
{
   SecurityFeatureType_Blank                          =  0,
   SecurityFeatureType_Fill                           =  1,
   SecurityFeatureType_Photo                          =  2,
   SecurityFeatureType_MRZ                            =  3,
   SecurityFeatureType_FalseLuminescence              =  4,
   SecurityFeatureType_HoloSimple                     =  5,
   SecurityFeatureType_HoloVerifyStatic               =  6,
   SecurityFeatureType_HoloVerifyMultiStatic          = 7,
   SecurityFeatureType_HoloVerifyDinamic              = 8,
   SecurityFeatureType_Pattern_NotInterrupted         = 9,
   SecurityFeatureType_Pattern_NotShifted             = 10,
   SecurityFeatureType_Pattern_SameColors             = 11,
   SecurityFeatureType_Pattern_IRInvisible            = 12,
   SecurityFeatureType_PhotoSize_Check                = 13,
   SecurityFeatureType_Portrait_Comparison_vsGhost    = 14,
   SecurityFeatureType_Portrait_Comparison_vsRFID     = 15,
   SecurityFeatureType_Portrait_Comparison_vsVisual   = 16,
   SecurityFeatureType_Barcode                        = 17,
   SecurityFeatureType_Pattern_DifferentLinesThicknes = 18,
   SecurityFeatureType_Portrait_Comparison_vsCamera   = 19,
   SecurityFeatureType_Portrait_Comparison_RFIDvsCamera = 20,
   SecurityFeatureType_GhostPhoto                     = 21,
   SecurityFeatureType_ClearGhostPhoto                = 22,
   SecurityFeatureType_InvisibleObject                = 23,
   SecurityFeatureType_LowContrastObject              = 24,
   SecurityFeatureType_Photo_Color                    = 25,
   SecurityFeatureType_Photo_Shape                    = 26,
   SecurityFeatureType_Photo_Corners                  = 27,
   SecurityFeatureType_Document_Cancelling_Detector   = 28,
   SecurityFeatureType_Portrait_Comparison_ExtvsVisual = 29,
   SecurityFeatureType_Portrait_Comparison_ExtvsRFID  = 30,
   SecurityFeatureType_Portrait_Comparison_ExtvsLive  = 31,
};
```

Значение констант:

- **SecurityFeatureType_Blank** – элемент бланка;
- **SecurityFeatureType_Fill** – элемент заполнения;
- **SecurityFeatureType_Photo** – фотография;
- **SecurityFeatureType_MRZ** – машиносчитываемая зона;

- **SecurityFeatureType_FalseLuminescence** – область вокруг фото;
- **SecurityFeatureType_HoloSimple** - зарезервировано для внутреннего использования;
- **SecurityFeatureType_HoloVerifyStatic** - зарезервировано для внутреннего использования;
- **SecurityFeatureType_HoloVerifyMultiStatic** - зарезервировано для внутреннего использования;
- **SecurityFeatureType_HoloVerifyDinamic** - зарезервировано для внутреннего использования;
- **SecurityFeatureType_Pattern_NotInterrupted** - зарезервировано для внутреннего использования;
- **SecurityFeatureType_Pattern_NotShifted** - зарезервировано для внутреннего использования;
- **SecurityFeatureType_Pattern_SameColors** - зарезервировано для внутреннего использования;
- **SecurityFeatureType_Pattern_IRInvisible** - зарезервировано для внутреннего использования;
- **SecurityFeatureType_PhotoSize_Check** - зарезервировано для внутреннего использования;
- **SecurityFeatureType_Portrait_Comparison_vsGhost** - сравнение основного портрета с дополнительным;
- **SecurityFeatureType_Portrait_Comparison_vsRFID** - сравнение основного портрета с портретом в RFID;
- **SecurityFeatureType_Portrait_Comparison_vsVisual** - сравнение основного портрета с портретом на другой странице.
- **SecurityFeatureType_Barcode** - проверка формата штрих кода;
- **SecurityFeatureType_Pattern_DifferentLinesThickness** – различная толщина линий;
- **SecurityFeatureType_Portrait_Comparison_vsCamera** – сравнение портрета с изображением с камеры;
- **SecurityFeatureType_Portrait_Comparison_RFIDvsCamera** – сравнение портрета с изображением из RFID;
- **SecurityFeatureType_GhostPhoto** – дополнительный портрет;
- **SecurityFeatureType_ClearGhostPhoto** – прозрачный дополнительный портрет;
- **SecurityFeatureType_InvisibleObject** – невидимый объект;
- **SecurityFeatureType_LowContrastObject** – неконтрастный объект;

- **SecurityFeatureType_Photo_Color** – цветность фотографии;
- **SecurityFeatureType_Photo_Shape** – квадратность фотографии;
- **SecurityFeatureType_Photo_Corners** – форма углов фотографии;
- **SecurityFeatureType_Document_Cancelling_Detector** – документ отменен;
- **SecurityFeatureType_Portrait_Comparison_ExtvsVisual** – сравнение портрета: внешнее изображение с визуальной зоной;
- **SecurityFeatureType_Portrait_Comparison_ExtvsRFID** – сравнение портрета: внешнее изображение с RFID;
- **SecurityFeatureType_Portrait_Comparison_ExtvsLive** – сравнение портрета: внешнее изображение с изображением с камеры.

5.4.26. eSecurityCriticalFlag

Перечисление **eSecurityCriticalFlag** содержит признак критичности проверки того или иного элемента подлинности документа.

```
enum eSecurityCriticalFlag
{
   CheckFeatureNotCritical   = 0,
   CheckFeatureIsCritical    = 1
};
```

Значение констант:

- **CheckFeatureNotCritical** – элемент защиты может отсутствовать на подлинных документах;
- **CheckFeatureIsCritical** – элемент защиты должен присутствовать на подлинных документах.

5.4.27. eIR_Visibility_Flag

Перечисление **eIR_Visibility_Flag** содержит признак видимости элемента изображения в ИК-диапазоне.

```
enum eIR_Visibility_Flag
{
   ElementInvisible = 0,
   ElementVisible   = 1,
   ElementColored   = 2,
   ElementGrayscale = 4,
};
```

Значение констант:

- **ElementInvisible** – элемент невидим;
- **ElementVisible** – элемент видим;
- **ElementColored** – элемент с цветом;

- **ElementGrayscale** – элемент в градациях серого.

5.4.28. eLED_Color

Перечисление **eLED_Color** содержит идентификаторы, определяющие цвет свечения индикаторов считывателя документов.

```
enum eLED_Color
{
   ledNone   = 0,
   ledRed    = 1,
   ledGreen  = 2,
   ledOrange = 3
};
```

Значение констант:

- **ledNone** – индикатор выключен;
- **ledRed** – красный свет;
- **ledGreen** – зеленый свет;
- **ledOrange** – оранжевый свет.

5.4.29. eFDS_Light

Перечисление **eFDS_Light** содержит идентификаторы, определяющие тип защиты документа, изображение для которого необходимо отобразить в главной информационной панели FDS.

```
enum eFDS_Light
{
   fdsWhite    = 1,
   fdsUV365    = 2,
   fdsIR       = 4,
   fdsMaterial = 6
};
```

Значение констант:

- **fdsWhite** – полиграфическая защита;
- **fdsUV365** – защита в области УФ-излучения (365 нм);
- **fdsIR** – защита в области ИК-излучения (900 нм);
- **fdsMaterial** – защита на уровне материалов.

5.4.30. eFDS_Panel

Перечисление **eFDS_Panel** содержит идентификаторы, определяющие тип панели СИС.

```
enum eFDS_Panel
{
   FDS_Panel_Main         = 0,
   FDS_Panel_Countries    = 1,
   FDS_Panel_Documents    = 2,
   FDS_Panel_Caption      = 3,
   FDS_Panel_Description  = 4,
   FDS_Panel_Illumination= 5,
   FDS_Panel_PageType     = 6
};
```

Значение констант:

- **FDS_Panel_Main** – панель с изображениями документа;
- **FDS_Panel_Countries** – панель со списком стран;
- **FDS_Panel_Documents** – панель со списком документов выбранной страны;
- **FDS_Panel_Caption** – информационная панель;
- **FDS_Panel_Description** – панель с общим описанием документа;
- **FDS_Panel_Illumination** – панель с кнопками, соответствующими различным типам защиты;
- **FDS_Panel_PageType** – панель с кнопками, активизирующими изображения для различных страниц документа.

5.4.31. eFDS_Panel_Position

Перечисление **eFDS_Panel_Position** содержит идентификаторы, определяющие расположение и видимость панели СИС.

```
enum eFDS_Panel_Position
{
   FDS_Panel_Hide      = 0,
   FDS_Panel_Show      = 1,
   FDS_Panel_ShowTop   = 2,
   FDS_Panel_ShowBottom = 3
};
```

Значение констант:

- **FDS_Panel_Hide** – скрыть панель;
- **FDS_Panel_Show** – показать панель;
- **FDS_Panel_ShowTop** – расположить панель у верхнего края родительского окна;
- **FDS_Panel_ShowBottom** – расположить панель у нижнего края родительского окна.

5.4.32. eRPRM_OutputFormat

Перечисление **eRPRM_OutputFormat** содержит идентификаторы, определяющие формат представления данных и механизм их передачи в пользовательское приложение при запросе результатов выполнения цикла сканирования и обработки с помощью функции **_CheckResult()**.

```
enum eRPRM_OutputFormat
{
   ofrDefault            = 0,
   ofrTransport_Clipboard = 0x00000002,
   ofrTransport_File     = 0x00000004,
   ofrFormat_XML         = 0x00010000,
   ofrFormat_FileBuffer  = 0x00020000,
   ofrFormat_ImagesXML   = 0x00040000,
   ofrFormat_JSON        = 0x00080000,
   ofrFileBuffer_File    = ofrTransport_File | ofrFormat_FileBuffer,
   ofrXML_Clipboard      = ofrTransport_Clipboard | ofrFormat_XML,
   ofrXML_File           = ofrTransport_File      | ofrFormat_XML,
   ofrFileBuffer_XML_Clipboard
                         = ofrXML_Clipboard | ofrFormat_FileBuffer,
   ofrFileBuffer_XML_File = ofrXML_File  | ofrFormat_FileBuffer,
   ofrJSON_Clipboard     = ofrTransport_Clipboard | ofrFormat_JSON,
```

```
   ofrJSON_File          = ofrTransport_File      | ofrFormat_JSON,
   ofrFileBuffer_JSON_Clipboard
                         = ofrJSON_Clipboard | ofrFormat_FileBuffer,
   ofrFileBuffer_JSON_File= ofrJSON_File  | ofrFormat_FileBuffer

};
```

Значение констант:

- **ofrDefault** – режим по умолчанию. Будет возвращен только указатель на структуру данных результата;
- **ofrTransport_Clipboard** – сформированное XML-представление результата или графическое изображение будет дополнительно помещено в буфер обмена ОС;
- **ofrTransport_File** – сформированное XML-представление результата или графическое изображение будет дополнительно записано в файл;
- **ofrFormat_XML** – сформировать XML-представление структуры данных результата;
- **ofrFormat_FileBuffer** – сформировать образ графического файла, содержащего запрашиваемое изображение;
- **ofrFormat_ImagesXML** – сформировать XML-представление образа графического файла, содержащего запрашиваемое изображение;
- **ofrFormat_JSON** – сформировать JSON- представление структуры данных результата.

5.4.33. eLexAnalysisDepth

Перечисление **eLexAnalysisDepth** содержит возможные значения настроек лексического анализа для команд **RPRM_Command_Options_Get_LexAnalysisDepth** и **RPRM_Command_Options_Set_LexAnalysisDepth**.

```
enum eLexAnalysisDepth
{
   eLAD_Default             = 0x0000,
   eLAD_ShowAllData         = 0x0001,
   eLAD_ShowDataAndResults  = 0x0002,
   eLAD_CheckStopListOFF    = 0x0004,
   eLAD_CheckDocNumZeroOFF  = 0x0008,
   eLAD_ICAOConvertionOFF   = 0x0010,
   eLAD_ConvertToCyrillicON= 0x0020,
   eLAD_GenerateRussiaMRZStrings  = 0x0040
   eLAD_CompareRussiaAuthority  = 0x0080,
   eLAD_CheckLowProbabilityEnable = 0x0100
};
```

Значение констант:

- **eLAD_Default** – режим по умолчанию, поля и результат с низкой вероятностью не выдаются, поля проверяются на присутствие в "стоп-листе", номер документа не должен быть нулевым;
- **eLAD_ShowAllData** – показывать все прочитанные данные, но отключать результаты сравнения/верификации;

- **eLAD_ShowDataAndResults** – показывать все данные и результаты вне зависимости от вероятности распознавания;
- **eLAD_CheckStopListOFF** – отключить проверку полей на присутствие в "стоп-листе";
- **eLAD_CheckDocNumZeroOFF** – отключить проверку на нулевой номер документа;
- **eLAD_ICAOConvertionOFF** – отключить конверсию значений текстовых полей в латинский алфавит для сравнения с МСЗ в соответствии с ICAO 9303;
- **eLAD_ConvertToCyrillicON** – включить конверсию значений текстовых полей на латинице в кириллицу;
- **eLAD_GenerateRussiaMRZStrings** – включить создание поля **ft_MRZ_Strings** для российских паспортов без МСЗ;
- **eLAD_CompareRussiaAuthority** - сравнить сгенерированные поля **ft_Authority_RUS** с **ft_Authority** в российском национальном паспорте;
- **eLAD_CheckLowProbabilityEnable -** включить фильтрацию результатов с низкой вероятностью.

5.4.34. eLexDateFormat

НЕ ИСПОЛЬЗУЕТСЯ

Перечисление **eLexDateFormat** содержит возможные значения форматов даты для структуры **TLexDateFormat** для команд **RPRM_Command_Options_Get_LexDateFormat** и **RPRM_Command_Options_Set_LexDateFormat**.

```
enum eLexDateFormat
{
   LDF_DefaultShort          = 0,
   LDF_DefaultLong           = 1,
   LDF_SystemShort           = 2,
   LDF_SystemLong            = 3,
   LDF_Universal             = 4,
   LDF_Custom                = 5

};
```

Значения констант:

- **LDF_DefaultShort** – режим по умолчанию, короткий формат даты текущего пользователя;
- **LDF_DefaultLong** – длинный формат даты текущего пользователя;
- **LDF_SystemShort** – короткий формат даты ОС;
- **LDF_SystemLong** – длинный формат даты ОС;
- **LDF_Universal** – универсальный формат даты по ISO 8601 вида “YYYY-MM-DD”;
- **LDF_Custom** – формат даты задается вручную.

5.4.35. eImageQualityCheckType

Перечисление **eImageQualityCheckType** содержит возможные значения типов проверок качества входных изображений.

```
enum eImageQualityCheckType
{
     IQC_ImageGlares      = 0,
     IQC_ImageFocus       = 1,
     IQC_ImageResolution  = 2,
     IQC_ImageColorness   = 3,
     IQC_Perspective      = 4,
     IQC_Bounds           = 5
};
```

Значения констант:

- **IQC_ImageGlares** – проверка наличия бликов;
- **IQC_ImageFocus** – проверка качества фокусировки;
- **IQC_ImageResolution** – проверка разрешения изображения;
- **IQC_ImageColorness** – проверка цвета изображения;
- **IQC_Perspective** – проверка изображения на предмет перспективы;
- **IQC_Bounds** – проверка изображения на предмет попадания документа в кадр целиком.

5.4.36. diDocType

Перечисление **diDocType** содержит возможные значения типов документов.

```
enum diDocType
{
     dtNotDefined                                    = 0,
     dtPassport                                      = 11,
     dtIdentityCard                                  = 12,
     dtDiplomaticPassport                            = 13,
     dtServicePassport                               = 14,
     dtSeamanIdentityDocument                        = 15,
     dtIdentityCardForResidence                      = 16,
     dtTravelDocument                                = 17,
     dtOther                                         = 99,
     dtVisaID2                                       = 29,
     dtVisaID3                                       = 30,
     dtNationalIdentityCard                          = 20,
     dtSocialIdentityCard                            = 21,
     dtAliensIdentityCard                            = 22,
     dtPrivilegedIdentityCard                        = 23,
     dtResidencePermitIdentityCard                   = 24,
     dtOriginCard                                    = 25,
     dtEmergencyPassport                             = 26,
     dtAliensPassport                                = 27,
     dtAlternativeIdentityCard                       = 28,
     dtAuthorizationCard                             = 32,
     dtBeginnerPermit                                = 33,
     dtBorderCrossingCard                            = 34,
     dtChauffeurLicense                              = 35,
     dtChauffeurLicenseUnder18                       = 36,
     dtChauffeurLicenseUnder21                       = 37,
     dtCommercialDrivingLicense                      = 38,
     dtCommercialDrivingLicenseInstructionalPermit   = 39,
     dtCommercialDrivingLicenseUnder18               = 40,
     dtCommercialDrivingLicenseUnder21               = 41,
     dtCommercialInstructionPermit                   = 42,
     dtCommercialNewPermit                           = 43,
     dtConcealedCarryLicense                         = 44,
     dtConcealedFirearmPermit                        = 45,
     dtConditionalDrivingLicense                     = 46,
     dtDepartmentOfVeteransAffairsIdentityCard       = 47,
     dtDiplomaticDrivingLicense                      = 48,
     dtDrivingLicense                                = 49,
     dtDrivingLicenseInstructionalPermit             = 50,
     dtDrivingLicenseInstructionalPermitUnder18      = 51,
     dtDrivingLicenseInstructionalPermitUnder21      = 52,
```

```
dtDrivingLicenseLearnersPermit                  = 53,
dtDrivingLicenseLearnersPermitUnder18           = 54,
dtDrivingLicenseLearnersPermitUnder21           = 55,
dtDrivingLicenseNovice                          = 56,
dtDrivingLicenseNoviceUnder18                   = 57,
dtDrivingLicenseNoviceUnder21                   = 58,
dtDrivingLicenseRegisteredOffender              = 59,
dtDrivingLicenseRestrictedUnder18               = 60,
dtDrivingLicenseRestrictedUnder21               = 61,
dtDrivingLicenseTemporaryVisitor                = 62,
dtDrivingLicenseTemporaryVisitorUnder18         = 63,
dtDrivingLicenseTemporaryVisitorUnder21         = 64,
dtDrivingLicenseUnder18                         = 65,
dtDrivingLicenseUnder21                         = 66,
dtEmploymentDrivingPermit                       = 67,
dtEnhancedChauffeurLicense                      = 68,
dtEnhancedChauffeurLicenseUnder18               = 69,
dtEnhancedChauffeurLicenseUnder21               = 70,
dtEnhancedCommercialDrivingLicense              = 71,
dtEnhancedDrivingLicense                        = 72,
dtEnhancedDrivingLicenseUnder18                 = 73,
dtEnhancedDrivingLicenseUnder21                 = 74,
dtEnhancedIdentityCard                          = 75,
dtEnhancedIdentityCardUnder18                   = 76,
dtEnhancedIdentityCardUnder21                   = 77,
dtEnhancedOperatorsLicense                      = 78,
dtFirearmsPermit                                = 79,
dtFullProvisionalLicense                        = 80,
dtFullProvisionalLicenseUnder18                 = 81,
dtFullProvisionalLicenseUnder21                 = 82,
dtGenevaConventionsIdentityCard                 = 83,
dtGraduatedDrivingLicenseUnder18                = 84,
dtGraduatedDrivingLicenseUnder21                = 85,
dtGraduatedInstructionPermitUnder18             = 86,
dtGraduatedInstructionPermitUnder21             = 87,
dtGraduatedLicenseUnder18                       = 88,
dtGraduatedLicenseUnder21                       = 89,
dtHandgunCarryPermit                            = 90,
dtIdentityAndPrivilegeCard                      = 91,
dtIdentityCardMobilityImpaired                  = 92,
dtIdentityCardRegisteredOffender                = 93,
dtIdentityCardTemporaryVisitor                  = 94,
dtIdentityCardTemporaryVisitorUnder18           = 95,
dtIdentityCardTemporaryVisitorUnder21           = 96,
dtIdentityCardUnder18                           = 97,
dtIdentityCardUnder21                           = 98,
dtIgnitionInterlockPermit                       = 100,
dtImmigrantVisa                                 = 101,
dtInstructionPermit                             = 102,
dtInstructionPermitUnder18                      = 103,
dtInstructionPermitUnder21                      = 104,
dtInterimDrivingLicense                         = 105,
dtInterimIdentityCard                           = 106,
dtIntermediateDrivingLicense                    = 107,
dtIntermediateDrivingLicenseUnder18             = 108,
dtIntermediateDrivingLicenseUnder21             = 109,
dtJuniorDrivingLicense                          = 110,
dtLearnerInstructionalPermit                    = 111,
dtLearnerLicense                                = 112,
dtLearnerLicenseUnder18                         = 113,
dtLearnerLicenseUnder21                         = 114,
dtLearnerPermit                                 = 115,
dtLearnerPermitUnder18                          = 116,
dtLearnerPermitUnder21                          = 117,
dtLimitedLicense                                = 118,
dtLimitedPermit                                 = 119,
dtLimitedTermDrivingLicense                     = 120,
dtLimitedTermIdentityCard                       = 121,
dtLiquorIdentityCard                            = 122,
dtNewPermit                                     = 123,
dtNewPermitUnder18                              = 124,
dtNewPermitUnder21                              = 125,
dtNonUsCitizenDrivingLicense                    = 126,
dtOccupationalDrivingLicense                    = 127,
dtOneidaTribeOfIndiansIdentityCard              = 128,
dtOperatorLicense                               = 129,
dtOperatorLicenseUnder18                        = 130,
dtOperatorLicenseUnder21                        = 131,
dtPermanentDrivingLicense                       = 132,
```

```
    dtPermitToReenter                                  = 133,
    dtProbationaryAutoLicense                          = 134,
    dtProbationaryDrivingLicenseUnder18                = 135,
    dtProbationaryDrivingLicenseUnder21                = 136,
    dtProbationaryVehicleSalespersonLicense            = 137,
    dtProvisionalDrivingLicense                        = 138,
    dtProvisionalDrivingLicenseUnder18                 = 139,
    dtProvisionalDrivingLicenseUnder21                 = 140,
    dtProvisionalLicense                               = 141,
    dtProvisionalLicenseUnder18                        = 142,
    dtProvisionalLicenseUnder21                        = 143,
    dtPublicPassengerChauffeurLicense                  = 144,
    dtRacingAndGamingComissionCard                     = 145,
    dtRefugeeTravelDocument                            = 146,
    dtRenewalPermit                                    = 147,
    dtRestrictedCommercialDrivingLicense               = 148,
    dtRestrictedDrivingLicense                         = 149,
    dtRestrictedPermit                                 = 150,
    dtSeasonalPermit                                   = 151,
    dtSeasonalResidentIdentityCard                     = 152,
    dtSeniorCitizenIdentityCard                        = 153,
    dtSexOffender                                      = 154,
    dtSocialSecurityCard                               = 155,
    dtTemporaryDrivingLicense                          = 156,
    dtTemporaryDrivingLicenseUnder18                   = 157,
    dtTemporaryDrivingLicenseUnder21                   = 158,
    dtTemporaryIdentityCard                            = 159,
    dtTemporaryInstructionPermitIdentityCard           = 160,
    dtTemporaryInstructionPermitIdentityCardUnder18    = 161,
    dtTemporaryInstructionPermitIdentityCardUnder21    = 162,
    dtTemporaryVisitorDrivingLicense                   = 163,
    dtTemporaryVisitorDrivingLicenseUnder18            = 164,
    dtTemporaryVisitorDrivingLicenseUnder21            = 165,
    dtUniformedServicesIdentityCard                    = 166,
    dtVehicleSalespersonLicense                        = 167,
    dtWorkerIdentificationCredential                   = 168,
    dtCommercialDrivingLicenseNovice                   = 169,
    dtCommercialDrivingLicenseNoviceUnder18            = 170,
    dtCommercialDrivingLicenseNoviceUnder21            = 171,
    dtPassportCard                                     = 172,
    dtPermanentResidentCard                            = 173,
    dtPersonalIdentificationVerification               = 174,
    dtTemporaryOperatorLicense                         = 175,
    dtDrivingLicenseUnder19                            = 176,
    dtIdentityCardUnder19                              = 177,
    dtVisa                                             = 178,
    dtTemporaryPassport                                = 179,
    dtVotingCard                                       = 180,
    dtHealthCard                                       = 181,
    dtCertificateOfCitizenship                         = 182,
    dtAddressCard                                      = 183,
    dtAirportImmigrationCard                           = 184,
    dtAlienRegistrationCard                            = 185,
    dtAPEHCard                                         = 186,
    dtCouponToDrivingLicense                           = 187,
    dtCrewMemberCertificate                            = 188,
    dtDocumentForReturn                                = 189,
    dtECard                                            = 190,
    dtEmploymentCard                                   = 191,
    dtHKSARImmigrationForm                             = 192,
    dtImmigrantCard                                    = 193,
    dtLabourCard                                       = 194,
    dtLaissezPasser                                    = 195,
    dtLawyerIdentityCertificate                        = 196,
    dtLicenseCard                                      = 197,
    dtPassportStateless                                = 198,
    dtPassportChild                                    = 199,
    dtPassportConsular                                 = 200,
    dtPassportDiplomaticService                        = 201,
    dtPassportOfficial                                 = 202,
    dtPassportProvisional                              = 203,
    dtPassportSpecial                                  = 204,
    dtPermissionToTheLocalBorderTraffic                = 205,
    dtRegistrationCertificate                          = 206,
    dtSEDESOLCard                                      = 207,
    dtSocialCard                                       = 208,
    dtTBCard                                           = 209,
    dtVehiclePassport                                  = 210,
    dtWDocument                                        = 211,
```

```
    dtDiplomaticIdentityCard                    = 212,
    dtConsularIdentityCard                      = 213,
    dtIncomeTaxCard                             = 214,
    dtResidencePermit                           = 215,
    dtDocumentOfIdentity                        = 216,
    dtBorderCrossingPermit                      = 217,
    dtPassportLimitedValidity                   = 218,
    dtSIMCard                                   = 219,
    dtTaxCard                                   = 220,
    dtCompanyCard                               = 221,
    dtDomesticPassport                          = 222,
    dtIdentityCertificate                       = 223,
    dtResidentIdCard                            = 224,
    dtArmedForcesIdentityCard                   = 225,
    dtProfessionalCard                          = 226,
    dtRegistrationStamp                         = 227,
    dtDriverCard                                = 228,
    dtDriverTrainingCertificate                 = 229,
    dtQualificationDrivingLicense               = 230,
    dtMembershipCard                            = 231,
    dtPublicVehicleDriverAuthorityCard          = 232,
    dtMarineLicense                             = 233,
    dtTemporaryLearnerDrivingLicense            = 234,
    dtTemporaryCommercialDrivingLicense         = 235,
    dtInterimInstructionalPermit                = 236,
    dtCertificateOfCompetency                   = 237,
    dtCertificateOfProficiency                  = 238,
};
```

Значение констант соответствует названию.

5.4.37. eRPRM_ResultStatus

Перечисление **eRPRM_ResultStatus** содержит возможные коды возврата из функций **_CheckResult()** и **_CheckResultFromList()**.

```
enum eRPRM_ResultStatus
{
  RPRM_ResultStatus_NotAvailable    = 0xffffffff,
  RPRM_ResultStatus_EndOfList       = 0xfffffffe,
  RPRM_ResultStatus_InvalidParameter = 0xfffffffd,
  RPRM_ResultStatus_IOError         = 0xfffffffb,
  RPRM_ResultStatus_InvalidFilename = 0xfffffffa,
  RPRM_ResultStatus_ClipboardError  = 0xfffffff9,
  RPRM_ResultStatus_NotEnoughMemory = 0xfffffff8,
  RPRM_ResultStatus_NotSupported    = 0xfffffff7
};
```

Значение констант:

- **RPRM_ResultStatus_NotAvailable** – запрошенный тип результата недоступен;
- **RPRM_ResultStatus_EndOfList** – на предыдущем шаге получения данных из полей результатов, представленных в виде структур-списков, был достигнут конец списка, и новых данных в обрабатываемом списке больше нет;
- **RPRM_ResultStatus_InvalidParameter** – неверный параметр вызова функции;
- **RPRM_ResultStatus_IOError** – ошибка файлового вывода;
- **RPRM_ResultStatus_InvalidFilename** – неверное имя файла;
- **RPRM_ResultStatus_ClipboardError** – невозможно выполнить операцию с буфером обмена ОС;
- **RPRM_ResultStatus_NotEnoughMemory** – не хватает памяти для выполнения операции;
- **RPRM_ResultStatus_NotSupported** – указанный формат представления данных или механизм их передачи

недоступен для заданного типа результата.

5.4.38. eRPRM_NotificationCodes

Перечисление **eRPRM_NotificationCodes** содержит возможные коды сообщений, получаемых пользовательским приложением через callback-функцию **NotifyFunc**.

```
enum eRPRM_NotificationCodes
{
   RPRM_Notification_Error                        =  0x00000000,
   RPRM_Notification_DeviceDisconnected           =  0x00000001,
   RPRM_Notification_DocumentReady                =  0x00000002,
   RPRM_Notification_Scanning                     =  0x00000004,
   RPRM_Notification_Calibration                  =  0x00000008,
   RPRM_Notification_CalibrationProgress          =  0x00000009,
   RPRM_Notification_EnumeratingDevices           =  0x0000000C,
   RPRM_Notification_ConnectingDevice             =  0x0000000D,
   RPRM_Notification_DocumentCanBeRemoved         =  0x0000000E,
   RPRM_Notification_LidOpen                      =  0x0000000F,
   RPRM_Notification_Processing                   =  0x00000010,
   RPRM_Notification_DownloadingCalibrationInfo   =  0x00000011,
   RPRM_Notification_LicenseExpired               =  0x00000012,
   RPRM_Notification_OperationProgress            =  0x00000013,
   RPRM_Notification_LatestAvailableSDK           =  0x00000014,
   RPRM_Notification_LatestAvailableDatabase      =  0x00000015,
   RPRM_Notification_VideoFrame                   =  0x00000016,
};
```

Значение констант:

- **RPRM_Notification_Error** Возникновение некритической ошибки. Код ошибки (одно из значений **eRPRM_ErrorCodes**) находится в параметре `value`. Обратите внимание, что отрицательные значения кодов ошибки служат для внутреннего использования и должны игнорироваться.

- **RPRM_Notification_DeviceDisconnected** Активный считыватель документов был отключен от USB-порта ПК, и дальнейшая работа с ним невозможна

- **RPRM_Notification_DocumentReady** Индикация срабатывания датчика присутствия документа. Значение в `value` показывает, был ли документ помещен в считыватель (`true`) либо удален из него (`false`)

- **RPRM_Notification_Scanning** Индикация выполнения операции сканирования изображений. Значение в `value` показывает начало (`false`) либо окончание операции (`true`)

- **RPRM_Notification_Calibration** Индикация выполнения операции калибровки считывателя документов. Значение в `value` показывает начало (`false`) либо окончание операции (`true`)

- **RPRM_Notification_CalibrationProgress** Индикация прогресса выполнения операции калибровки считывателя. Значение в `value` содержит значение в процентах от общей длительности операции

- **RPRM_Notification_EnumeratingDevices** Индикация выполнения операции поиска подключенных к ПК в текущий момент времени считывателей документов. Значение в `value` показывает начало (`false`) либо окончание (`true`) операции

- **RPRM_Notification_ConnectingDevice** Индикация выполнения операции подключения считывателя документов. Значение в `value` показывает начало (`false`) либо окончание (`true`) операции

- **RPRM_Notification_DocumentCanBeRemoved** Индикация момента времени, когда документ может быть извлечен из считывателя после проведения сканирования изображений.

- **RPRM_Notification_LidOpen**
Индикация срабатывания датчика открытия крышки прибора. Значение в `value` показывает, была ли крышка открыта (`true`) либо закрыта (`false`)

- **RPRM_Notification_Processing**
Индикация выполнения операции обработки изображений. Значение в `value` показывает начало (`false`) либо окончание операции (`true`)

- **RPRM_Notification_DownloadingCalibrationInfo**
Индикация прогресса скачивания калибровочной информации из считывателя. Значение в `value` содержит значение в процентах от общей длительности операции

- **RPRM_Notification_LicenseExpired**
Индикация того, что лицензия на обновление SDK для этого устройство истекла. `value` содержит количество дней с 1 января 1900, в течение которых лицензия была активна.

- **RPRM_Notification_OperationProgress**
Индикация прогресса операции. Значение в `value` содержит значение в процентах от общей длительности операции

- **RPRM_Notification_LatestAvailableSDK**
Индикация последней доступной версии SDK. Значение в `value` содержит версию в четырех байтах.

- **RPRM_Notification_LatestAvailableDatabase**
Индикация последней доступной версии базы данных. Значение в `value` содержит версию в четырех байтах.

- **RPRM_Notification_VideoFrame**
Индикация доступности нового изображения видеодетекции. Значение в `value` содержит `TVideodetectionNotification*`.

5.4.39. eRPRM_ErrorCodes

Перечисление **eRPRM_ErrorCodes** содержит возможные коды, возвращаемые функциями главной управляющей библиотеки SDK и получаемые через callback-функцию **NotifyFunc** с приходом сообщения **RPRM_Notification_Error**.

```
enum eRPRM_ErrorCodes
{
   RPRM_Error_NoError                          = 0x00000000,
   RPRM_Error_AlreadyDone                      = 0x00000001,
   RPRM_Error_NoGraphManager                   = 0x00000002,
   RPRM_Error_CantRegisterMessages             = 0x00000003,
   RPRM_Error_NoServiceManager                 = 0x00000004,
   RPRM_Error_CantConnectServiceManager        = 0x00000006,
   RPRM_Error_CantCreateDeviceLibraryEvent     = 0x00000009,
   RPRM_Error_InvalidParameter                 = 0x0000000C,
   RPRM_Error_NotInitialized                   = 0x0000000D,
   RPRM_Error_Busy                             = 0x0000000E,
   RPRM_Error_NotEnoughMemory                  = 0x00000011,
   RPRM_Error_BadVideo                         = 0x00000014,
   RPRM_Error_ScanAborted                      = 0x00000015,
   RPRM_Error_CantRecognizeDocumentType        = 0x00000016,
   RPRM_Error_CantSetupSensor                  = 0x00000018,
   RPRM_Error_NotTrueColorDesktop              = 0x00000019,
   RPRM_Error_NotAvailable                     = 0x0000001A,
   RPRM_Error_DeviceError                      = 0x0000001B,
   RPRM_Error_DeviceDisconnected               = 0x00000020,
   RPRM_Error_WrongThreadContext               = 0x00000030,
   RPRM_Error_COMServers                       = 0x00000031,
   RPRM_Error_NoDocumentReadersFound           = 0x00000032,
   RPRM_Error_NoTranslationMngr                = 0x00000033,
   RPRM_Error_NoActiveDevice                   = 0x00000034,
   RPRM_Error_ConnectingDevice                 = 0x00000035,
   RPRM_Error_Failed                           = 0x00000036,
   RPRM_Error_LightIsNotAllowed                = 0x00000037,
   RPRM_Error_ImageIOError                     = 0x00000038,
   RPRM_Error_CantStoreCalibrationData         = 0x00000039,
   RPRM_Error_DeviceNotCalibrated              = 0x0000003A,
   RPRM_Error_CantCompensateDistortion         = 0x0000003B,
   RPRM_Error_OperationCancelled               = 0x0000003C,
   RPRM_Error_CantLocateDocumentBounds         = 0x0000003D,
   RPRM_Error_CantRefineImages                 = 0x0000003E,
   RPRM_Error_CantCropRotateImages             = 0x0000003F,
   RPRM_Error_IncompleteImagesList             = 0x00000040,
   RPRM_Error_CantReadMRZ                      = 0x00000041,
   RPRM_Error_CantFindBarcodes                 = 0x00000042,
   RPRM_Error_DeviceIDNotSupported             = 0x00000043,
   RPRM_Error_DeviceIDNotStored                = 0x00000044,
   RPRM_Error_DeviceDriver                     = 0x00000045,
   RPRM_Error_CalibrationOpenLid               = 0x00000046,
   RPRM_Error_Calibration_Brightness           = 0x00000047,
   RPRM_Error_Calibration_WhiteBalance         = 0x00000048,
   RPRM_Error_Calibration_TargetPosition       = 0x00000049,
   RPRM_Error_Calibration_LightBlank           = 0x0000004A
   RPRM_Error_Calibration_LightDistortion      = 0x0000004B,
   RPRM_Error_Calibration_LightLevel           = 0x0000004C,
   RPRM_Error_Calibration_LightLevelHigh       = 0x0000004D,
   RPRM_Error_Calibration_LightLevelLow        = 0x0000004E,
   RPRM_Error_8305CameraAbsent                 = 0x00000050,
   RPRM_Error_NotImplemented                   = 0x00000051,
   RPRM_Error_RemoveDocument                   = 0x00000052,
   RPRM_Error_BadDataFile                      = 0x00000053, ,
   RPRM_Error_BadInputImage                    = 0x00000054
};
```

Значение констант:

- **RPRM_Error_NoError** Операция завершена успешно
- **RPRM_Error_AlreadyDone** Операция уже выполнялась ранее

• **RPRM_Error_NoGraphManager** Ошибка подключения к *DGraph.dll*. Дальнейшая работа невозможна

• **RPRM_Error_CantRegisterMessages** Ошибка регистрации управляющих сообщений Windows. Дальнейшая работа невозможна

• **RPRM_Error_NoServiceManager** Невозможно найти или инициализировать *RSrvMngr.exe*. Дальнейшая работа невозможна

• **RPRM_Error_CantConnectServiceManager** Ошибка подключения к *RSrvMngr.exe*. Дальнейшая работа невозможна

• **RPRM_Error_CantCreateDeviceLibraryEvent** Ошибка создания управляющих элементов библиотеки управления устройством. Дальнейшая работа невозможна

• **RPRM_Error_InvalidParameter** Неверный параметр вызова функции

• **RPRM_Error_NotInitialized** Главная управляющая библиотека не была проинициализирована

• **RPRM_Error_Busy** Выполняется предыдущая команда

• **RPRM_Error_NotEnoughMemory** Не хватает памяти для выполнения запрашиваемого действия

• **RPRM_Error_BadVideo** Видео работает в замедленном режиме – менее 3 кадров в секунду. Данная ситуация может возникнуть из-за проблем с правильным функционированием драйвера устройства и системных средств Windows. Дальнейшая работа с устройством невозможна. Это ошибка может возникнуть только при подключении устройств, управляемых посредством `DirectShow`

• **RPRM_Error_ScanAborted** Сканирование не завершено, поскольку документ был извлечен из устройства до момента получения всех необходимых изображений

• **RPRM_Error_CantRecognizeDocumentType** Не удалось определить тип документа

• **RPRM_Error_CantSetupSensor** Во время инициализации устройства обнаружено присутствие документа, что мешает проведению настройки чувствительности датчика. Необходимо извлечь документ и повторить операцию подключения

• **RPRM_Error_NotTrueColorDesktop** Текущие настройки цветности рабочего стола ОС препятствуют нормальной работе с устройством (для считывателей с

управлением посредством `DirectShow`). Рабочий стол должен иметь цветность 24 или 32 бита на цвет

- **RPRM_Error_NotAvailable** Запрошенная операция недоступна в текущий момент времени

- **RPRM_Error_DeviceError** Ошибка инициализации устройства. Дальнейшая работа с ним невозможна

- **RPRM_Error_DeviceDisconnected** Устройство было отключено от ПК и выполнение запрошенного действия невозможно

- **RPRM_Error_WrongThreadContext** Функция **_Free()** вызвана не из того рабочего потока приложения, из которого вызывалась функция **_Initialize()**

- **RPRM_Error_COMServers** Ошибка функционирования используемых COM-серверов

- **RPRM_Error_NoDocumentReadersFound** Не найдено ни одного считывателя документов, подключенного к ПК

- **RPRM_Error_NoTranslationMngr** Ошибка подключения к *RTrans.exe*

- **RPRM_Error_NoActiveDevice** Выполнение запрошенной операции невозможно из-за отсутствия подключенного (командой **RPRM_Command_Device_ Connect**) считывателя документов

- **RPRM_Error_ConnectingDevice** Ошибка подключения устройства. Дальнейшая работа с ним невозможна

- **RPRM_Error_Failed** Общая ошибка выполнения запрошенной операции

- **RPRM_Error_LightIsNotAllowed** Указанная комбинация схем освещения не может быть использована при проведении цикла сканирования

- **RPRM_Error_ImageIOError** Ошибка файлового ввода/вывода при манипуляциях с изображениями

- **RPRM_Error_CantStoreCalibrationData** Невозможно сохранить данные калибровки

- **RPRM_Error_DeviceNotCalibrated** Устройство не откалибровано, запрашиваемое действие невозможно

• **RPRM_Error_CantCompensateDistortion** Ошибка при проведении компенсации геометрических искажений изображений

• **RPRM_Error_OperationCancelled** Текущая операция прервана пользователем

• **RPRM_Error_CantLocateDocumentBounds** Не удалось определить границы документа на изображении

• **RPRM_Error_CantRefineImages** Ошибка при проведении компенсации неравномерности освещения и цветовой коррекции

• **RPRM_Error_CantCropRotateImages** Ошибка при повороте или обрезке изображения по найденным границам документа

• **RPRM_Error_IncompleteImagesList** Предоставленный для обработки список изображений не полон (для команды **RPRM_Command_ProcessImagesList**)

• **RPRM_Error_CantReadMRZ** Не удалось найти или прочитать МСЗ

• **RPRM_Error_CantFindBarcodes** Не удалось найти или прочитать штрихкод

• **RPRM_Error_DeviceIDNotSupported** Прошивка считывателя не поддерживает сохранение идентификационного номера (по умолчанию он равен `0x0000`)

• **RPRM_Error_DeviceIDNotStored** Идентификационный номер не прошит в память считывателя (по умолчанию он равен `0xFFFF`)

• **RPRM_Error_DeviceDriver** Необходимо обновить драйвер устройства (для считывателей, оборудованных цифровой камерой `Cypress` или `Micron`). Дальнейшая работа со считывателем невозможна

• **RPRM_Error_CalibrationOpenLid** Крышка считывателя открыта, что мешает проведению калибровки (для считывателей, оборудованных цифровой камерой `Cypress` или `Micron`). Необходимо закрыть крышку и повторить операцию

• **RPRM_Error_Calibration_Brightness** Калибровка яркости картинки неудовлетворительная

• **RPRM_Error_Calibration_WhiteBalance** Калибровка баланса белого неудовлетворительная

• **RPRM_Error_Calibration_TargetPosition**
Калибровка положения тест-объекта неудовлетворительная

• **RPRM_Error_Calibration_LightBlank** Свечение калибровочной картинки неудовлетворительное

• **RPRM_Error_Calibration_LightDistortion** Равномерность свечения калибровочной картинки неудовлетворительная

• **RPRM_Error_Calibration_LightLevel** Калибровочная картинка неудовлетворительная

• **RPRM_Error_Calibration_LightLevelHigh**
Калибровочная картинка слишком светлая

• **RPRM_Error_Calibration_LightLevelLow** Калибровочная картинка слишком темная

• **RPRM_Error_8305CameraAbsent** Не обнаружена видеокамера считывателя 8305 (при построении списка установленных в системе считывателей). Работа со считывателем 8305 невозможна

• **RPRM_Error_NotImplemented** Вызванная команда не реализована

• **RPRM_Error_RemoveDocument** Необходимо удалить документ до инициализации считывателя документов

• **RPRM_Error_BadDataFile** Файл данных отсутствует или не может быть прочитан

• **RPRM_Error_ BadInputImage** Изображение во входном файле не в фокусе и/или содержит блики

5.4.40. eRPRM_Commands

Перечисление **eRPRM_Commands** содержит набор команд главной управляющей библиотеки SDK.

```
enum eRPRM_Commands
{
  RPRM_Command_Device_Count                          = 0x00000001,
  RPRM_Command_Device_Features                       = 0x00000002,
  RPRM_Command_Device_RefreshList                    = 0x00000003,
  RPRM_Command_Device_ActiveIndex                    = 0x00000004,
  RPRM_Command_Device_Connect                        = 0x00000005,
  RPRM_Command_Device_Disconnect                     = 0x00000006,
  RPRM_Command_Device_Light_ScanList_Clear           = 0x00000007,
  RPRM_Command_Device_Light_ScanList_AddTo           = 0x00000008,
  RPRM_Command_Device_Light_ScanList_Default         = 0x00000016,
  RPRM_Command_Device_Light_ScanList_Count           = 0x00000017,
  RPRM_Command_Device_Light_ScanList_Item            = 0x00000018,
  RPRM_Command_Device_Light_TurnOn                   = 0x00000009,
  RPRM_Command_Device_LED                            = 0x0000000B,
  RPRM_Command_Device_PlaySound                 = 0x0000000F,
  RPRM_Command_Device_Set_ParamLowLight              = 0x0000000C,
  RPRM_Command_Device_Get_ParamLowLight              = 0x0000000D,
  RPRM_Command_Device_Calibration                    = 0x00000015,
  RPRM_Command_Process                               = 0x00000019,
  RPRM_Command_Options_GraphicFormat_Count           = 0x0000001A,
  RPRM_Command_Options_GraphicFormat_Name            = 0x0000001B,
  RPRM_Command_Options_GraphicFormat_Select          = 0x0000001C,
  RPRM_Command_Options_GraphicFormat_ActiveIndex     = 0x00000020,
  RPRM_Command_Options_GetSDKCapabilities            = 0x0000001E,
  RPRM_Command_Options_GetSDKAuthCapabilities  = 0x00000035,
```

```
  RPRM_Command_Options_Set_MRZTestQualityParams     = 0x00000022,
  RPRM_Command_Options_Get_MRZTestQualityParams     = 0x00000023,
  RPRM_Command_ProcessImagesList                    = 0x00000024,
  RPRM_Command_Options_Set_CurrentDocumentType      = 0x00000027,
  RPRM_Command_Options_Get_CurrentDocumentType      = 0x00000028,
  RPRM_Command_Options_Set_CustomDocTypeMode        = 0x00000029,
  RPRM_Command_Options_Get_CustomDocTypeMode        = 0x0000002A,
  RPRM_Command_Get_DocumentsInfoList                = 0x0000002B,
  RPRM_Command_OCRLexicalAnalyze                    = 0x0000002C,
  RPRM_Command_Device_IsCalibrated                  = 0x0000002D,
  RPRM_Command_Options_Set_CheckResultHeight        = 0x0000002E,
  RPRM_Command_Device_Set_WorkingVideoMode          = 0x00000030,
  RPRM_Command_Device_Get_WorkingVideoMode          = 0x00000031,
  RPRM_Command_Options_Set_AuthenticityCheckMode    = 0x00000032,
  RPRM_Command_Options_Get_AuthenticityCheckMode    = 0x00000033,
  RPRM_Command_Options_Get_BatteryStatus            = 0x00000034
  RPRM_Command_Options_BuildExtLog             = 0x00000040,
  RPRM_Command_Device_SetFrequencyDivider      = 0x00000041,
  RPRM_Command_Device_Get_DriverVersion             = 0x00000042,
  RPRM_Command_Device_APM_Mode                      = 0x00000044,
  RPRM_Command_Device_UseVideoDetection        = 0x00000045,
  RPRM_Command_ExpertAnalyze                        = 0x00000046,
  RPRM_Command_ClearResults                         = 0x00000047,
  RPRM_Command_Options_GraphicFormat_SetCompressionRatio   = 0x00000048
  RPRM_Command_Options_GraphicFormat_GetCompressionRatio   = 0x00000049,
  RPRM_Command_Process_Cancel                       = 0x0000004A,
  RPRM_Command_ExcludeCapabilities                  = 0x0000004B,
  RPRM_Command_ExcludeAuthCapabilities              = 0x0000004C,
  RPRM_Command_MakeSingleShot                       = 0x0000004D,
  RPRM_Command_Device_GetFrequencyDivider           = 0x0000004E,
  RPRM_Command_ComplexAuthenticityCheck             = 0x0000004F,
  RPRM_Command_Options_Set_GlareCompensation        = 0x00000050,
  RPRM_Command_Options_Set_ExtendProcessingModes    = 0x00000051,
  RPRM_Command_Options_Get_AppendVisa               = 0x00000052,
  RPRM_Command_Options_Set_AppendVisa               = 0x00000053,
  RPRM_Command_Options_Set_MultiPageProcessingMode = 0x00000054,
  RPRM_Command_Device_Get_Calibration_FrequencyDivider   = 0x00000055,
  RPRM_Command_PortraitGraphicalAnalyze             = 0x00000056,
  RPRM_Command_Options_Set_SmartUV                  = 0x00000057,
  RPRM_Command_Options_Set_RotateResultImages       = 0x00000058,
  RPRM_Command_BSIDocCheckXML                       = 0x00000059,
  RPRM_Command_Options_Get_QuickMrzProcessing       = 0x0000005A,
  RPRM_Command_Options_Set_QuickMrzProcessing       = 0x0000005B,
  RPRM_Command_Device_SetVideoDetectionDivider= 0x0000005C,
  RPRM_Command_Device_GetVideoDetectionDivider= 0x0000005D,
  RPRM_Command_Device_SetRequiredOcrFields          = 0x0000005E,
  RPRM_Command_Device_GetRequiredOcrFields          = 0x0000005F,
  RPRM_Command_Options_Get_BatteryNumber            = 0x00000060,
  RPRM_Command_Options_Get_QuickBoardingPassProcessing
                                               = 0x00000061,
  RPRM_Command_Options_Set_QuickBoardingPassProcessing
                                               = 0x00000062,
  RPRM_Command_Options_Set_WaitForReadingComplete   = 0x00000063,
  RPRM_Command_ReadingComplete                      = 0x00000064,
  RPRM_Command_Options_Get_LexAnalysisDepth         = 0x00000065,
  RPRM_Command_Options_Set_LexAnalysisDepth         = 0x00000066,
  RPRM_Command_Options_Get_LexDateFormat            = 0x00000067,
  RPRM_Command_Options_Set_LexDateFormat            = 0x00000068,
  RPRM_Command_Device_Get_GetJpegImages             = 0x00000069,
  RPRM_Command_Device_Set_GetJpegImages             = 0x0000006A,
  RPRM_Command_BSIDocCheckXMLv2                     = 0x0000006B,
  RPRM_Command_Device_Get_TrustDPI                  = 0x0000006C,
  RPRM_Command_Device_Set_TrustDPI                  = 0x0000006D,
  RPRM_Command_Options_Get_LexParams                = 0x0000006E,
  RPRM_Command_Options_Set_LexParams                = 0x0000006F,
  RPRM_Command_Options_Get_StopOnBadInputImage      = 0x00000070,
  RPRM_Command_Options_Set_StopOnBadInputImage      = 0x00000071,
  RPRM_Command_Set_ProcessParametersJson            = 0x00000072,
  RPRM_Command_Options_Set_VideodetectionLowSensibility
                                               = 0x00000073,
  RPRM_Command_Options_Set_TrustVideodetectionResult
                                               = 0x00000074,
  RPRM_Command_Device_Get_LED                       = 0x00000075,
  RPRM_Command_Get_DatabaseInfo                     = 0x00000076,

  };
```

Подробное описание каждой команды приводится далее.

5.5. Система команд SDK

Основной функцией библиотеки, посредством которой пользовательское приложение может инициировать все необходимые действия для работы со считывателями документов, является функция **_ExecuteCommand()**. В качестве параметров она принимает командный триплет: *код команды* (параметр **command**), *входной параметр команды* (параметр **params**) и *указатель на контейнер-приемник* (параметр **result**) для возвращаемых результатов.

Описание каждой отдельной команды приводится ниже по следующей схеме:
Код команды
Входной параметр
Выходной параметр
Назначение:
<Краткое описание>

5.5.1. RPRM_Command_Device_Count

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: определение общего количества считывателей документов, подключенных к ПК в текущий момент времени

5.5.2. RPRM_Command_Device_Features

Входной параметр – `long`
Выходной параметр – `TRegulaDeviceProperties **`
Назначение: получение информации о характеристиках считывателя документов

Индекс считывателя документов в общем списке указывается во входном параметре команды.

Память под **TRegulaDeviceProperties** выделяется главной управляющей библиотекой и не требует освобождения в пользовательском приложении.

5.5.3. RPRM_Command_Device_RefreshList

Входной параметр – не используется
Выходной параметр – не используется
Назначение: обновление общего списка подключенных к ПК считывателей документов

Не используется.

Для построения списка подключенных устройств необходимо выполнить переинициализацию главной управляющей библиотеки последовательностью вызовов функций **`_Free()`** и **`_Initialize()`**.

5.5.4. RPRM_Command_Device_ActiveIndex

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: определение индекса текущего активного считывателя документов

5.5.5. RPRM_Command_Device_Connect

Входной параметр – `long`
Выходной параметр – не используется
Назначение: активизация конкретного считывателя из общего списка

Индекс считывателя документов в общем списке указывается во входном параметре команды.

В случае указания вместо индекса устройства значения –1 возможны два варианта продолжения работы:

- если в списке подключенных к ПК устройств находится только один считыватель документов, он будет подключен по умолчанию;
- если в списке подключенных к ПК устройств находятся несколько считывателей документов, то будет подключен первый из них.

5.5.6. RPRM_Command_Device_Disconnect

Входной параметр – не используется
Выходной параметр – не используется
Назначение: отключение текущего активного считывателя документов

5.5.7. RPRM_Command_Device_Light_ScanList_Clear

Входной параметр – не используется
Выходной параметр – не используется
Назначение: очистка списка схем освещения для сканирования

5.5.8. RPRM_Command_Device_Light_ScanList_Default

Входной параметр – не используется
Выходной параметр – не используется

Назначение: заполнение списка схем освещения для сканирования значениями по умолчанию

5.5.9. RPRM_Command_Device_Light_ScanList_Count

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: определение общего количества элементов в списке схем освещения для сканирования

5.5.10. RPRM_Command_Device_Light_ScanList_Item

Входной параметр – `long`
Выходной параметр – `long *`
Назначение: чтение значения конкретного элемента из списка схем освещения для сканирования

Индекс запрашиваемого элемента *списка* указывается во входном параметре команды.

5.5.11. RPRM_Command_Device_Light_ScanList_AddTo

Входной параметр – `long`
Выходной параметр – не используется
Назначение: занесение нового элемента в список схем освещения для сканирования

Во входном параметре указывается идентификатор схемы освещения (возможна и комбинация идентификаторов). Принимаются только значения, которые присутствовали в *списке схем освещения для сканирования*, заполненном значениями по умолчанию.

5.5.12. RPRM_Command_Device_Light_TurnOn

Входной параметр – `long`
Выходной параметр – не используется
Назначение: включение схемы освещения

Во входном параметре указывается идентификатор схемы освещения (возможна и комбинация идентификаторов), которую необходимо активизировать. Принимаются любые корректные комбинации идентификаторов.

Эта команда может использоваться в целях тестирования и никакого влияния на проведение цикла сканирования и обработки данных не оказывает.

5.5.13. RPRM_Command_Device_LED

Входной параметр – `TIndicationLED *`
Выходной параметр – не используется
Назначение: задание логики поведения индикаторных светодиодов считывателя документов

5.5.14. RPRM_Command_Device_Set_ParamLowLight

Входной параметр – `long`
Выходной параметр – не используется
Назначение: установка значения экспозиции видеокамеры при получении изображений для схемы УФ-освещения

5.5.15. RPRM_Command_Device_ PlaySound

Входной параметр – не используется
Выходной параметр – не используется
Назначение: управление звуковым сигналом считывателя документов

5.5.16. RPRM_Command_Device_Get_ParamLowLight

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: чтение текущего значения экспозиции видеокамеры при получении изображений для схемы УФ-освещения

5.5.17. RPRM_Command_Device_Calibration

Входной параметр – не используется
Выходной параметр – не используется
Назначение: проведение калибровки активного считывателя документов

5.5.18. RPRM_Command_Process

Входной параметр – `long`
Выходной параметр – не используется
Назначение: проведение цикла сканирования и обработки данных для активного считывателя документов

Во входном параметре команды должна содержаться комбинация значений из перечисления **`eRPRM_GetImage_Modes`**, которая определяет набор функций получения и обработки изображений для данного цикла и, соответственно, набор запрашиваемых результатов.

5.5.19. RPRM_Command_ProcessImagesList

Входной параметр – `TResultContainerList*`
Выходной параметр – `long`
Назначение: проведение цикла обработки данных для списка изображений

Список изображений, предназначенных для обработки, передается во входном параметре **`params`**. Пользовательское приложение несет ответственность за выделение памяти под этот список и его элементы.

Во втором параметре команды **`result`**, который обычно используется для получения результатов, для этой команды передается комбинация значений из перечисления **`eRPRM_GetImage_Modes`**, определяющая набор функций обработки изображений для данного цикла и, соответственно, набор запрашиваемых результатов.

5.5.20. RPRM_Command_Options_GraphicFormat_Count

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: определение общего количества графических форматов записи изображений

5.5.21. RPRM_Command_Options_GraphicFormat_Select

Входной параметр – `long`
Выходной параметр – не используется
Назначение: выбор графического формата записи изображений

Во входном параметре указывается индекс графического формата в общем списке.

5.5.22. RPRM_Command_Options_GraphicFormat_Name

Входной параметр – `long`
Выходной параметр – `char **`
Назначение: получение символьного имени одного из доступных графических форматов записи изображений

Во входном параметре указывается индекс графического формата в общем списке.

Память для хранения передаваемой символьной строки выделяется главной управляющей библиотекой и не требует освобождения в пользовательском приложении.

Возвращаемая символьная строка содержит расширение файла запрашиваемого формата (например, «.BMP»).

5.5.23. RPRM_Command_Options_GraphicFormat_ActiveIndex

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получение индекса текущего графического формата записи изображений

5.5.24. RPRM_Command_Options_GetSDKCapabilities

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: определение набора функциональных возможностей главной управляющей библиотеки при работе с текущим активным считывателем документов

Возвращаемое в выходном параметре значение – комбинация значений из перечисления **`eRPRM_Capabilities`**.

5.5.25. RPRM_Command_Options_GetSDKAuthCapabilities

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: определение набора MCL проверок подлинности при работе с текущим активным считывателем документов

5.5.26. RPRM_Command_Options_Set_MRZTestQualityParams

Входной параметр – `TCommandsMRZTestQuality*`
Выходной параметр – не используется
Назначение: установка параметров контроля качества заполнения МСЗ документа

5.5.27. RPRM_Command_Options_Get_MRZTestQualityParams

Входной параметр – не используется
Выходной параметр – `TCommandsMRZTestQuality**`
Назначение: получение текущих параметров контроля качества заполнения МСЗ документа

Память для хранения заполняемой структуры **`TCommandsMRZTestQuality`** выделяется главной управляющей библиотекой и не требует освобождения в пользовательском приложении.

5.5.28. RPRM_Command_Options_Get_CurrentDocumentType

Входной параметр – не используется
Выходной параметр – `char **`
Назначение: получение текущего значения символьного идентификатора типа документа, выбираемого по умолчанию для проведения последующих операций распознавания

5.5.29. RPRM_Command_Options_Set_CurrentDocumentType

Входной параметр – `char *`
Выходной параметр – не используется
Назначение: установка символьного идентификатора типа документа, выбираемого по умолчанию для проведения последующих операций распознавания

Память для хранения передаваемой символьной строки выделяется главной управляющей библиотекой и не требует освобождения в пользовательском приложении.

5.5.30. RPRM_Command_Options_Set_CustomDocTypeMode

Входной параметр – `long`
Выходной параметр – не используется
Назначение: активизация режима пользовательского определения типа документа

5.5.31. RPRM_Command_Options_Get_CustomDocTypeMode

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получение текущего значения активности режима пользовательского определения типа документа

5.5.32. RPRM_Command_Get_DocumentsInfoList

Входной параметр – не используется
Выходной параметр – не используется
Назначение: получение полного списка документов, хранящихся в подключенной базе данных документов

Результат типа **`RPRM_ResultType_DocumentsInfoList`** передается в пользовательское приложение через вызов callback-функции **`ResultReceivingFunc`** и доступен через вызов функции **`_CheckResult()`**.

5.5.33. RPRM_Command_OCRLexicalAnalyze

Входной параметр – `TDocVisualExtendedInfo *`
Выходной параметр – не используется
Назначение: проведение лексического анализа текстовых данных, полученных операциями чтения МСЗ, полей заполнения документа, штрихкодов и данных из памяти RFID-микросхемы документа

Структура `TDocVisualExtendedInfo`, передаваемая во входном параметре команды, должна содержать данные из памяти RFID-микросхемы и может быть получена при совместной работе с «*SDK для считывателей бесконтактных идентификационных микросхем*».

При отсутствии данных чтения RFID-микросхемы во входном параметре допустима передача значения `0` (`NULL`).

5.5.34. RPRM_Command_Device_IsCalibrated

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: определение факта проведения калибровки для активного устройства

Возвращаемое значение (`TRUE` или `FALSE`) показывает, был ли откалиброван активный считыватель документов или данную процедуру необходимо провести заново.

5.5.35. RPRM_Command_Device_Set_WorkingVideoMode

Входной параметр – `long`
Выходной параметр – не используется
Назначение: переключение между доступными режимами работы сенсора видеокамеры

В качестве входного параметра используется одно из значений перечисления **`eRPRM_VideoModes`** (определяется на основании характеристик подключенного считывателя документов **`TRegulaDeviceProperties`**).

5.5.36. RPRM_Command_Device_Get_WorkingVideoMode

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получение текущего значения режима работы сенсора видеокамеры

5.5.37. RPRM_Command_Options_Set_CheckResultHeight

Входной параметр – `long`
Выходной параметр – не используется
Назначение: установка требуемой высоты (в точках) изображений, получаемых с помощью функций `CheckResultFromList()` и `CheckResult()`

При получении изображений в качестве результатов проведения цикла сканирования с помощью функций **`CheckResultFromList()`** и **`CheckResult()`** их размер (высота) будет приведен в соответствие с установленным данной командой значением. Для получения изображений в оригинальном размере необходимо установить данный параметр в `0`.
Этот параметр игнорируется при вызове **`_CheckResult()`** в следующих случаях:

- получение `RPRM_ResultType_RawImage` без параметра `output` установленного в `ofrFormat_XML` и/или `ofrFormat_FileBuffer`;
- получение `RPRM_ResultType_EOSImage`.

5.5.38. RPRM_Command_Options_Set_AuthenticityCheckMode

Входной параметр – `long`
Выходной параметр – не используется
Назначение: задание набора функций проверки подлинности документа (с заданным `RPRM_GetImage_Modes_Authenticity` при выполнении команды `RPRM_Command_Process`)
Во входном параметре задается комбинация значений из перечисления **`eRPRM_Authenticity`**.

5.5.39. RPRM_Command_Options_Get_AuthenticityCheckMode

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получение текущего набора функций проверки подлинности документа

5.5.40. RPRM_Command_Options_Get_BatteryStatus

Входной параметр – `long`
Выходной параметр – `long *`
Назначение: получение текущего статуса заряда аккумулятора с выбранным индексом мобильных версий считывателя документов. При отсутствии в считывателе опции RPRM_DeviceAdditionalFeature_Accumulator возвращает код ошибки RPRM_ResultStatus_NotAvailable

Возвращаемое значение (выходной параметр):

- `0–100` – заряд аккумулятора, %;
- `0xFF` – нет батареи;
- `0xFE` – идет заряд;

- `0xF0` – заряд 100 %.

5.5.41. RPRM_Command_Options_BuildExtLog

Входной параметр – `long`
Выходной параметр – не используется
Назначение: включение/выключение ведения протокола работы

Файлы протокола работы SDK с подробной информацией о производимых действиях создаются в директориях: `\Users\[User Name]\AppData\Local\Regula\Logs`.

5.5.42. RPRM_Command_Device_SetFrequencyDivider

Входной параметр – `long`
Выходной параметр – не используется
Назначение: установка делителя частоты видеочипа

Для стабильной работы под операционной системой Windows 7 или на малопроизводительном компьютере последовательно устанавливайте делитель от `1` до `5` до тех пор, пока не добьетесь стабильной работы.

5.5.43. RPRM_Command_Device_Get_DriverVersion

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получение версии драйвера

5.5.44. RPRM_Command_Device_APM_Mode

Входной параметр – `long`
Выходной параметр – не используется
Назначение: установка режима работы считывателя документов

Для уменьшения электромагнитного излучения или увеличения длительности работы от аккумуляторов мобильных версий считывателя документов установите с помощью этой команды `APM = 1`.

5.5.45. RPRM_Command_Device_UseVideoDetection

Входной параметр – `long`
Выходной параметр – не используется
Назначение: зарезервировано для внутренних нужд

5.5.46. RPRM_Command_ExpertAnalyze

Входной параметр – `TDocVisualExtendedInfo *`

Выходной параметр – не используется
Назначение: зарезервировано для внутренних нужд

5.5.47. RPRM_Command_ClearResults

Входной параметр – не используется
Выходной параметр – не используется
Назначение: очистка выделенной памяти под предыдущие результаты обработки документа

5.5.48. RPRM_Command_Options_GraphicFormat_SetCompressionRatio

Входной параметр – `long`
Выходной параметр – не используется
Назначение: установка коэффициента сжатия для сохранения в графических форматах, использующих алгоритмы сжатия с потерей информации (например, JPEG). Значение параметра от `0` до `10`

5.5.49. RPRM_Command_Options_GraphicFormat_GetCompressionRatio

Входной параметр – не используется
Выходной параметр – `long*`
Назначение: получение текущего коэффициента сжатия для сохранения в графических форматах, использующих алгоритмы сжатия с потерей информации (например, JPEG)

5.5.50. RPRM_Command_Process_Cancel

Входной параметр – не используется
Выходной параметр – не используется
Назначение: отмена процесса сканирования или калибровки

5.5.51. RPRM_Command_ExcludeCapabilities

Входной параметр – `long`
Выходной параметр – не используется
Назначение: задание набора `eRPRM_Capabilities`, исключенного из дальнейшей работы. Выполняется до подключения устройства

5.5.52. RPRM_Command_ExcludeAuthCapabilities

Входной параметр – `long`
Выходной параметр – не используется

Назначение: задание набора `eRPRM_Authenticity`, исключенного из дальнейшей работы. Выполняется до подключения устройства

5.5.53. RPRM_Command_MakeSingleShot

Входной параметр – `long`
Выходной параметр – `long` (используется как входной)
Назначение: съемка одного кадра со светом `eRPRM_Lights` во входном параметре и режимом съемки `eRPRM_GetImage_Modes` (`long`) в выходном параметре

5.5.54. RPRM_Command_Device_GetFrequencyDivider

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получение значения делителя частоты видеочипа

5.5.55. RPRM_Command_ComplexAuthenticityCheck

Входной параметр – не используется
Выходной параметр – не используется
Назначение: проведение анализа результатов проверки подлинности всех страниц текущего многостраничного документа

5.5.56. RPRM_Command_Options_Set_GlareCompensation

Входной параметр – `long`
Выходной параметр – не используется
Назначение: включение (входной параметр `!= 0`) или отключение (входной параметр `== 0`) режима компенсации бликов на белом и ИК-изображениях

5.5.57. RPRM_Command_Options_Set_ExtendProcessingModes

Входной параметр – `long`
Выходной параметр – не используется
Назначение: включение (входной параметр `!= 0`) или отключение (входной параметр `== 0`) режима автоматического включения режимов обработки при их необходимости

5.5.58. RPRM_Command_Options_Get_AppendVisa

Не используется.

5.5.59. RPRM_Command_Options_Set_AppendVisa

Не используется.

5.5.60. RPRM_Command_Options_Set_MultiPageProcessingMode

Входной параметр – `long`
Выходной параметр – не используется
Назначение: включение (входной параметр `!= 0`) или отключение (входной параметр `== 0`) режима многостраничной обработки (по умолчанию включен)

5.5.61. RPRM_Command_Device_Get_Calibration_FrequencyDivider

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получение значения делителя частоты видеочипа, при котором был откалиброван прибор

5.5.62. RPRM_Command_PortraitGraphicalAnalyze

Входной параметр – `TResultContainerList *`
Выходной параметр – не используется
Назначение: проведение сравнения портретов, полученных при сканировании документа, а также из RFID (передается во входном параметре)

5.5.63. RPRM_Command_Options_Set_SmartUV

Входной параметр – `long`
Выходной параметр – не используется
Назначение: включение (входной параметр `!= 0`) или отключение (входной параметр `== 0`) режима улучшения качества получаемых изображений в УФ

5.5.64. RPRM_Command_Options_Set_RotateResultImages

Входной параметр – `long`
Выходной параметр – не используется
Назначение: включение (входной параметр `!= 0`) или отключение (входной параметр `== 0`) режима поворота результирующих изображений в соответствии с ориентацией портрета

5.5.65. RPRM_Command_BSIDocCheckXML

Входной параметр – `TResultContainerList *`
Выходной параметр – `char **`
Назначение: создание результата в формате XML по стандарту BSI TR-03135 v1.

5.5.66. RPRM_Command_Options_Get_BatteryNumber

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получение количества установленных батарей.

5.5.67. RPRM_Command_Options_Get_QuickBoardingPassProcessing

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получение состояние режима быстрой обработки посадочных талонов.

5.5.68. RPRM_Command_Options_Set_QuickBoardingPassProcessing

Входной параметр – `long`
Выходной параметр – не используется
Назначение: включение (входной параметр `!= 0`) или отключение (входной параметр `== 0`) режима быстрой обработки посадочных талонов.

5.5.69. RPRM_Command_Options_Set_QuickMrzProcessing

Входной параметр – `long`
Выходной параметр – не используется
Назначение: включение (входной параметр `!= 0`) или отключение (входной параметр `== 0`) режима быстрой обработки МСЗ.

5.5.70. RPRM_Command_Options_Get_QuickMrzProcessing

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получение состояния режима быстрой обработки МСЗ.

5.5.71. RPRM_Command_Device_SetVideoDetectionDivider

Входной параметр – `long`
Выходной параметр – не используется
Назначение: установка делителя размера кадра видеодетекции (Входной параметр = 2 – размер / 2, и т.д.).

5.5.72. RPRM_Command_Device_GetVideoDetectionDivider

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получение значения делителя размера кадра видеодетекции.

5.5.73. RPRM_Command_Device_SetRequiredOcrFields

Входной параметр – `TDwordArray *`
Выходной параметр – не используется
Назначение: установка требуемых полей OCR.

5.5.74. RPRM_Command_Device_GetRequiredOcrFields

Входной параметр – не используется
Выходной параметр – `TDwordArray **`
Назначение: получение установленного параметра требуемых полей OCR.

5.5.75. RPRM_Command_Options_Set_WaitForReadingComplete

Входной параметр – `long`
Выходной параметр – не используется
Назначение: установить (Входной параметр != 0) или отключить (Входной параметр == 0) значение параметра WaitForReadingComplete. Используется в считывателях 72x3 совместно с командой `RPRM_Command_ReadingComplete`.

5.5.76. RPRM_Command_ReadingComplete

Входной параметр – `long`
Выходной параметр – не используется
Назначение: Сообщает, что чтение RFID закончено и можно выбрасывать карточку. Используется в считывателях 72x3 совместно с командой `RPRM_Command_Options_Set_WaitForReadingComplete`.

5.5.77. RPRM_Command_Options_Get_LexAnalysisDepth

Не используется.

5.5.78. RPRM_Command_Options_Set_LexAnalysisDepth

Не используется.

5.5.79. RPRM_Command_Options_Get_LexDateFormat

Не используется.

5.5.80. RPRM_Command_Options_Set_LexDateFormat

Не используется.

5.5.81. RPRM_Command_Device_Get_GetJpegImages

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получить значение параметра `GetJpegImages`.

5.5.82. RPRM_Command_Device_Set_GetJpegImages

Входной параметр – `long`
Выходной параметр – не используется
Назначение: установить значение параметра `GetJpegImages`.

5.5.83. RPRM_Command_BSIDocCheckXMLv2

Входной параметр – `TResultContainerList *`
Выходной параметр – `char **`
Назначение: получить результат XML в соответствии с форматом BSI Technical Guideline TR-03135-2.

5.5.84. RPRM_Command_Options_Get_TrustDPI

Входной параметр – не используется
Выходной параметр – `long *`
Назначение: получить значение параметра `TrustDPI`.

5.5.85. RPRM_Command_Options_Set_TrustDPI

Входной параметр – `long`

Выходной параметр – не используется
Назначение: установить значение параметра `TrustDPI`.

5.5.86. RPRM_Command_Options_Get_LexParams

Входной параметр – не используется
Выходной параметр – `char **`
Назначение: получить параметры лексического анализа в формате JSON.

5.5.87. RPRM_Command_Options_Set_LexParams

Входной параметр – `char *`
Выходной параметр – не используется
Назначение: установить параметры лексического анализа в формате JSON.

5.5.88. RPRM_Command_Options_Get_StopOnBadInputImage

Входной параметр – не используется
Выходной параметр – `long *`, по умолчанию `FALSE`
Назначение: получить значение параметра `StopOnBadInputImage`.

5.5.89. RPRM_Command_Options_Set_StopOnBadInputImage

Входной параметр – `long`
Выходной параметр – не используется
Назначение: установить значение параметра `StopOnBadInputImage`.

5.5.90. RPRM_Command_Set_ProcessParametersJson

Входной параметр – `char *`
Выходной параметр – не используется
Назначение: установить параметры в формате JSON.

5.5.91. RPRM_Command_Options_Set_VideodetectionLowSensibility

Входной параметр – `long`
Выходной параметр – не используется
Designation: установить значение параметра `VideodetectionLowSensibility`.

5.5.92. RPRM_Command_Options_Set_TrustVideodetectionResult

Входной параметр – `long`
Выходной параметр – не используется

Назначение: установить значение параметра `TrustVideodetectionResult`.

5.5.93. RPRM_Command_Device_Get_LED

Входной параметр – не используется
Выходной параметр – `TIndicationLED *`
Назначение: получение логики поведения индикаторных светодиодов считывателя документов.

5.5.94. RPRM_Command_Get_DatabaseInfo

Входной параметр – не используется
Выходной параметр – `char **`
Назначение: получение информации о базе данных в формате строки JSON.