

ARCUS 2.1

для интеграции
с кассовыми
решениями

Руководство администратора

/ Версия 2.00

Содержание

Хронология изменений	3
Условные обозначения.....	3
Глоссарий.....	3
Аббревиатуры	5
Введение	6
1_ ARCUS 2.1 CAP	7
1_1 Состав, основные принципы функционирования	7
1_1_1 Использование TLS шифрования в режиме IP через кассу	8
1_2 Режим работы с кассой	8
1_2_1 Настройки режима работы с кассой	8
1_2_2 Файл настроек ARCUS.CFG для Режимы работы с кассой терминального приложения ISO (Openway, SmartVista, Tieto, CTL)	9
1_2_3 Файл настроек ARCUS.CFG для Режимы работы с кассой терминального приложения Unipay (CompassPlus, SPDH).....	11
1_3 Интеграция с ККМ	13
1_3_1 Формат и содержание файла данных output.dat	15
1_3_2 Дополнительные данные платежа (ARCUS2.0).....	16
1_4 Вызов ARCUS 2.1	17
1_5 Файл инициализации Cashreg.ini	19
1_6 Настройки ARCUS 2.1 CAP	24
1_6_1 Файл ops.ini для ISO8583.....	24
1_6_2 Чтение карты для ISO	27
1_6_3 Файл ops.ini для Compass+ и SPDH.....	27
1_6_4 Файл rc_conv.ini	28
1_6_5 Файл rc_res.ini	29
1_7 Вывод сообщений и меню на терминал (Dialog Direct)	29
1_8 Реализация поддержки внешних диалогов.....	30
1_9 Возврат (отмена) без предъявления карты	31
1_10 Передача информации о товарах (Arcus 2.0).....	32
1_11 Печать данных на принтере терминала (Arcus 2.0)	33
1_12 Специальные версии ARCUS2 CAP (Arcus 2.0)	33
1_12_1 ПО ККМ DKLINK с расчетом скидки на терминале	33
1_12_2 ПО ККМ DATAKPAT с расчетом скидки на кассе.....	34
1_12_3 ПО ККМ Set Retail 10.....	35
1_13 Эксплуатация	36
1_13_1 Интеграция ARCUS2.1 CAP с ПО ККМ.....	36
1_13_2 Виртуальная клавиатура (Arcus 2.0)	36
1_13_3 Способы и периодичность контроля правильности работы ПО	37
1_13_4 Аварийные ситуации и способы их устранения	37
1_13_5 Описание фрагментов файла лога	37

2_ Интеграция через RS232 или USB	38
2_1 Схема взаимодействия с ПО ККМ	38
3_ Низкоуровневая интеграция	39
ПРИЛОЖЕНИЕ 1 Ключи для методов set/get	40
ПРИЛОЖЕНИЕ 2 Пример использования библиотеки	41
Прототипы используемых функций	41
Пример использования на языке C++	41
Пример использования на языке C#	44
Пример использования на языке Object Pascal (Delphi):	45
Приложение 3 Интеграция с использованием COM Object	49
Описание PCPOSTConnectorObj	49
Приложение 4 Описание структуры UserAuthintFull	51
Передача дополнительной информации (Arcus 2.0)	53
Приложение 5 Пример использования библиотек ArcCom.dll / libarcus.so.x.x.x.xx	54
Вызов UserAuthintFull	54
Вызов Результирующей структуры	56
Пример libarcus.so.x.x.x.xx	57
Приложение 6 Прототипы функций диалогов ARCUS.h	58
Пример использования	59
Приложение 7 Рекомендуемый перечень операций при интеграции	60
Приложение 8 Функционал DLS (опционально, ARCUS 2.0)	61
Режим легкой интеграции с кассой	61
Режим полной интеграции с кассой	61
Оплата	61
Оплата наличными с начислением бонусов и без	61
Возврат со списанием бонусов и без	61
Открытие смены	61
Инкассация	61
Работа с функционалом DLS с модулем SwichSales	62
Приложение 9 Описание внутренних кодов ответов	63
Внутренние коды ответа для Universal EMV POS (NewWay)	63
Внутренние коды ответа для Universal EMV POS (Unipay)	65
Приложение 10 Платежи в сторону третьих лиц в режиме работы с кассой (TPTR, SPDH, ARCUS 2.0)	66

Хронология изменений

/ Таблица 1. Хронология изменений

Версия	Дата	Автор	Детали
2.00	06/04/2021	Попова И.Л.	Создание новой версии документа на основе предыдущей версии с добавлением информации из документа с описанием ARCUS 2.0.

Условные обозначения

Следующие условные обозначения использовались в документе:

/ Таблица 2. Условные обозначения

Обозначение	Значение
	Примечание
	Обратите особое внимание

Глоссарий

/ Таблица 3. Глоссарий

Термин	Определение
EMV	Стандарт взаимодействия чиповых карт для проведения платежей по карте (составлено из первых букв компаний Europay, MasterCard and VISA).
EMV Приложение	Микропрограмма и блок параметров на карточке, относящейся к международной платежной системе EMV.
ПИН-пад	Электронное устройство, предназначенное для ввода PIN кода.
PIN код	Секретный код, вводимый клиентом (держателем карточки), для подтверждения того, что именно он обладает полномочиями по использованию карточки.
POS-терминал, терминал	Аппаратно-программный комплекс, предназначенный для регистрации факта приобретения товара и упрощения процедуры приобретения товара.
Код действия терминалов (TAC)	Действия ПО терминала при возникновении конкретной ситуации (отказ от операции, проведении операции онлайн или действие по умолчанию).
Коммуникационный скрипт	Минипрограмма (последовательность команд), определяющая сценарий подключения терминала к серверу авторизации или серверу TMS.

Термин	Определение
Клиент/ Держатель	Физическое лицо (в том числе уполномоченный представитель юридического лица), на имя которого в силу заключенного между ним и Эмитентом договора и/или на имя которого по указанию лица, заключившего договор с Эмитентом, выпущена Карта.
Магнитная полоса	Полоса, содержащая идентификационную информацию о карте и ее держателе.
Операция	Расчетная операция, осуществляемая с использованием Карты с обязательной Авторизацией в соответствии с законодательством Российской Федерации, нормативно-правовыми актами Банка России и договором с Эмитентом, проводимая в Торговой точке по требованию Держателя.
Операция возврата	Расчетная операция, совершаемая с использованием Карты при возникновении необходимости в возврате Держателю денежных средств, списанных с его счета в рамках ранее совершенной Операции.
Операция отмены	Процедура отмены Кода Авторизации до формирования Электронного журнала.
Отчет Электронного терминала (Электронный журнал)	Документ или совокупность документов в электронной форме, сформированные за определенный период времени при совершении Операций/Операций с использованием POS-терминала.
Программное обеспечение (ПО)	Программное обеспечение POS-терминала.
Профиль	Один из объектов системы, предназначенный для хранения настроек групп параметров, определяющих алгоритм работы терминала в процессе выполнения операции с картой или административной операцией.
Реквизиты Карты	Номер Карты, срок действия Карты, код безопасности CVV2/CVC2/Batch code/CVV II/CAV2 (для Карт Платежных систем Visa International/MasterCard Worldwide/American Express/Diners Club International/JCB International соответственно).
Сервисный код карты	Код карты, который указывается область ее действия и тип.
Сервисный код коммерсанта (МСС)	Четырехзначный номер, который обозначает профиль деятельности, которым занимается коммерсант.
Скрипт эмитента (сценарий)	Команда или командная строка, передаваемая от эмитента к терминалу с целью выполнения команд(ы) микропроцессорной картой.
Способ верификации владельца карты	Способ верификации того, что человек, предъявляющий карту, имеет права на ее использование (например, сверка подписи или введение PIN кода).
Справочник	Один из объектов системы, предназначенный для хранения основных значений, шаблонов, терминов и алгоритмов, которыми оперирует система.
Транзакция	Группа последовательных операций, которая представляет собой логическую единицу работы с данными. Транзакция может быть выполнена либо целиком и успешно, соблюдая целостность данных и независимо от параллельно идущих других транзакций, либо не выполнена вообще и тогда она не должна произвести никакого эффекта. Транзакции обрабатываются транзакционными системами, в процессе работы которых создается история транзакций.
Хост, коммуникационный сервер	Сервер базы данных платежной системы, который отвечает на запросы POS-терминала точки обслуживания, а также снабжает его необходимой информацией.
Эквайер	Банк или компания, осуществляющая весь спектр операций по взаимодействию с точками обслуживания карточек, которая состоит из терминалов в торгово-сервисной сети и банкоматов.
Эмитент	Организация, выпустившая (эмитировавшая) карточки для развития и финансирования своей деятельности.

Аббревиатуры

/ Таблица 4. Аббревиатуры

Аббревиатура	Расшифровка
AID	Application Identification Number (Идентификационный номер приложения).
AVN	Application Version Number (Номер версии приложения).
BIN	Bank Identification Number (Банковский идентификационный номер).
CAP	Cash register Application.
CVM	Cardholder Verification Method (Способ верификации владельца карты).
DDOL	Dynamic Data Object List (Список элементов данных необходимых для динамической аутентификации).
ICC	Integrated Circuit(s) Card (Карта с интегрированными электронными схемами).
ID	Identifier (идентификатор).
ISO	International Organization for Standardization (Международная организация стандартизации).
MCC	Merchant Category Code (Сервисный код коммерсанта).
MS	Magnetic Stripe (магнитная полоса).
TAC	Terminal Action Code(s) (Код(ы) действия терминалов).
PAN	Primary Account Number (Индивидуальный номер карты).
PIN	Personal Identification Number (Персональный идентификационный номер).
POS	Point of Sale (точка продажи, сервиса).
RID	Registered Application Provider Identifier (Зарегистрированный номер провайдера приложения; например, VISA, MasterCard).
TDOL	Transaction Data Object List (Список данных о транзакции участвующих при вычислении сертификата).
TMM	Terminal Management Mode (модуль управления терминалами).
TMS	Terminal Management System (система управления терминалами).
СУБД	Система управления базами данных.

Введение

Данное руководство, разработано компанией ООО «Инженико» и описывает этапы установки и настройки ARCUS 2.1 – библиотеки, позволяющей проводить интеграцию Терминального платежного приложения Universal EMV POS и кассового программного обеспечения. ARCUS 2.1 CAP представляет собой совокупность программно-аппаратных средств, позволяющих обеспечить прием безналичных платежей на контрольно-кассовых машинах (ККМ) по банковским картам, включая магнитные, чиповые карты (EMV), бесконтактные карты, а также при ручном вводе данных.



Документ содержит описание программного продукта ARCUS 2.1, который включает в себя почти весь функционал устаревающего решения ARCUS 2.0 (функционал, который поддерживается только в решении ARCUS 2.0 отмечен соответствующим примечанием).



Для решений, в которых подключение терминала к ККМ выполняется по интерфейсу Ethernet, а взаимодействие выполняется при помощи файлового обмена используется программный продукт ARCUS NET SERVER версии 2.0 - описание этого продукта в документации на Ingenico Net Server.

Для решений, в которых подключение терминала к ККМ выполняется по интерфейсу Ethernet и для ККМ, реализованных по схеме «тонкий клиент» используется программный продукт ARCUS NET SERVER версии 1.3 (устаревающее).

Широкие возможности конфигурирования обеспечивают гибкость в настройке при наличии разнообразных требований к функциональности.

ПО соответствует основным требованиям и стандартам международных платежных систем.

Пользователи и администраторы, работающие с программным продуктом, должны пройти обязательное обучение в компании ООО «Инженико» или компании-партнере и получить сертификат.

При возникновении вопросов по работе с программным обеспечением, обращайтесь в службу поддержки ООО «Инженико» - support@ingenico.ru.

Допускается незначительные несоответствия данной документации и программного обеспечения, связанные с постоянным развитием программных продуктов.

Не допускается использование текстов и изображений, входящих в данный документ, без согласования с ООО «Инженико».

1_ARCUS 2.1 CAP

ARCUS 2.1 CAP – это библиотека, предназначенная для минимизации необходимых изменений в существующих решениях кассового программного обеспечения (ПО ККМ) при интеграции терминального оборудования Ingenico на платформах Telium2 и TETRA с ККМ, работающих под управлением данного ПО.



С апреля 2021 года функционал более ранней версии приложения ARCUS 2.0 CAP почти полностью поддерживается в ARCUS 2.1 CAP - функционал, доступный только в ARCUS 2.0 отмечен соответствующим комментарием.

Библиотека является связующим звеном между кассовым ПО и терминальным оборудованием со стандартным платежным приложением Universal EMV POS, реализуя обмен данными по протоколу ARCUS2 и обеспечивая диалог кассир-терминал на экране кассы или терминала (Режим работы с кассой).

ARCUS 2.1 CAP отвечает за:

- поддержку протокола обмена с POS-оборудованием через коммуникации ККМ;
- прозрачную передачу данных между POS-оборудованием и хостом банка;
- взаимодействие ПО POS-оборудования с кассиром в процессе транзакции по банковской карте.



Рис. 1. Схема работы ARCUS 2.1 CAP

Программное решение работает на кассовом ПО в следующих операционных системах:

- Windows (XP SP3 и выше);
- Linux (в некоторых случаях может потребоваться перекомпиляция модуля под конкретный тип операционной системы).



Взаимодействие с ККМ на MS DOS поддерживается только в ARCUS 2.0, но не описано в данном документе, т.к. функционал является устаревшим.

1_1 Состав, основные принципы функционирования

Программное обеспечение ARCUS2 это библиотека CAP¹ (Windows/Linux), которая обеспечивает:

- прозрачную передачу данных между платежным приложением, установленным на терминал или ПИН-пад, и процессингом (напрямую по TCP/IP с использованием сетевого интерфейса кассового компьютера (режим «IP через кассу») если не предусмотрено отдельное подключение ПИН-пад в локальную сеть ТСП);
- взаимодействие (диалог) между платежным приложением, установленным на терминал или ПИН-пад, и кассиром;
- интеграцию с АРМ кассира.

Кассовое приложение (библиотека) ARCUS 2.1 CAP является зависимой от поставщиков АРМ кассира частью. Сегодня на рынке кассового ПО существует широкий список поставщиков, отличающихся между собой как платформой (Windows, Linux, UNIX, 1C, etc), используемым сетевым

¹ ARCUS 2 CAP - Cash Register Application

интерфейсом (TCP/IP), так и способностью к адаптации (взаимодействию) с ПО иных производителей. Поэтому модуль CAP – это набор реализаций одной задачи для разных поставщиков АРМ кассы. Наиболее распространенный вариант поставки кассового приложения CAP – Windows или Linux приложение², взаимодействующее с ПИН-падом по RS232 или USB интерфейсу и с процессингом по TCP/IP.

1_1_1 Использование TLS шифрования в режиме IP через кассу

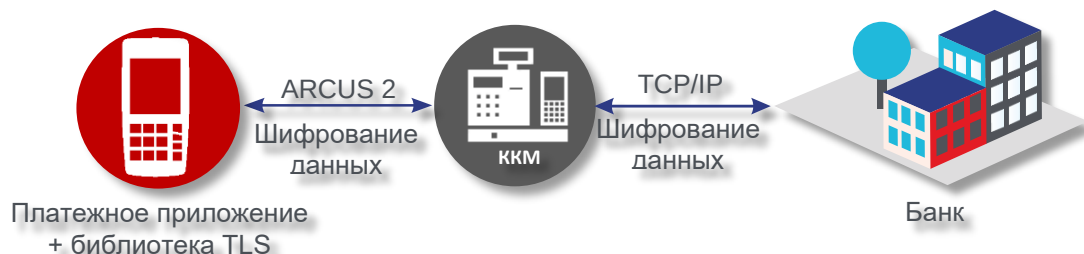


Рис. 2. TLS шифрование в режиме IP через кассу

Логика работы:

1. Данные о транзакции шифруются на терминале при помощи библиотеки TLS, и с использованием протокола ARCUS2 передаются в зашифрованном виде на ККМ.
2. ККМ принимает данные и, используя свои коммуникации, передает их на хост.
3. Хост расшифровывает полученные данные, проводит операцию, формирует ответ в зашифрованном виде и передает его ККМ.
4. ККМ получает ответ и при помощи протокола ARCUS2 передает его терминалу.
5. Терминал получает ответ от ККМ, расшифровывает его при помощи библиотеки TLS, и завершает операцию.

1_2 Режим работы с кассой

Платежный терминал может работать в двух режимах:

- автономный режим – терминал подключен к банковскому хосту через коммуникационный канал (например, Ethernet, GPRS), кассир инициализирует операцию и вводит данные для оплаты на терминале (производится двойной ввод данных), получает подтверждение проведения операции и вводит подтверждение оплаты на кассе;
- режим работы с кассой – терминал физически подключен к кассе (например, через USB, Bluetooth) и ожидает от нее команду и данные, необходимые для проведения операции, выполняет необходимые операции и возвращает на кассу результат проведения транзакции.

Для переключения терминала в режим работы с кассой выполните следующие действия:

1. Для терминалов Telium2 и TETRA при перезагрузке или включении питания, дождитесь появления окна с сообщением «Для настройки работы с кассой нажмите 'F'» и нажмите кнопку 'F'.



При определенных настройках для перехода в режим работы с кассой терминал может запросить пароль.

2. Работа с кассой?

- НЕТ – терминал загружается в автономном режиме;
- ДА – терминал переходит к настройке режима работы с кассой.

1_2_1 Настройки режима работы с кассой

Настройки режима работы с кассой может выполняться двумя способами:

² Вариант для Linux

- при помощи конфигурационного файла ARCUS.CFG, в котором задаются значения всех необходимых параметров;
- вручную при помощи ARCUS меню - при переключении в Режим работы с кассой, заданные вручную параметры сбрасываются в состояние, соответствующее настройкам файла ARCUS.CFG или значениям по-умолчанию, если файл отсутствует.



Описание ARCUS меню - см. Universal EMV POS – руководство администратора.

Файл ARCUS.CFG – текстовый файл в кодировке CP1251, содержит настройки и параметры подключения, загружается на терминал в раздел HOST (процесс загрузки файлов на терминал описан в документе Руководство по установке Universal EMV POS).

Для того, чтобы параметры из файла ARCUS.CFG применились выполните «5» - ARCUS 2 Настройки – Режим работы с кассой – ДА. На экране должно отобразиться сообщение «ПАРАМЕТРЫ КАССЫ ЗАГРУЖЕНЫ».



Параметры настройки и их значения отличаются, в зависимости от протокола.

Для специальных версий программного обеспечения, например для ПО Fuel+, реализованы дополнительные настройки, описание этих настроек доступно в документации на соответствующее программное обеспечение, например Руководство администратора - Топливное Приложение Ingenico Fuel+.

1_2_2 Файл настроек ARCUS.CFG для Режимы работы с кассой терминального приложения ISO (Openway, SmartVista, Tieto, CTL)

/ Таблица 5. Описание файла настроек ARCUS.CFG (ISO)

Параметр	Описание	Возможное значение
#Interface - настройки интерфейса		
PORT	Порт подключения	- COM0 - последовательный порт непосредственно на терминале - USB³ – USB порт непосредственно на терминале - COM_SL – последовательный интерфейс на MagicBox - COMU – USB порт на базе - ETH - Ethernet на терминале - BLUETOOTH - Bluetooth на терминале
PORT-SPEED	Скорость передачи данных	2400/9600/19200/ 115200
DATA-BITS	Количество бит данных	7/ 8
PARTITY	Контроль четности	NONE /EVEN/ODD
STOP-BITS	Стоп биты	1 /2
ARCUSSEVER -IP	IP для подключения к ARCUS Net Server (подробнее см. Ingenico Net Server руководств по эксплуатации)	Значение в формате xxx.xxx.xxx.xxx
ARCUSSEVER -PORT	Номер порта для подключения к ARCUS Net Server (подробнее см. Ingenico Net Server руководств по эксплуатации)	Значение в формате xxxx

³ Значения по умолчанию выделены **жирным** шрифтом

BLUETOOTH-PIN-LENGTH	Длина ПИН кода для подключения по Bluetooth	4-8
USE-TMSID	Использовать TMS ID	YES/NO
USE-ARCUS-ID	Использовать ARCUS ID	YES/NO
ARCUS-ID	Значение ARCUS ID	
#Network - настройки сети		
IP-ON-CASHREG	IP через кассу	YES/NO (для терминалов iUN и ПО TE_02 значение по умолчанию – NO)
STATIC-IP	Сетевые настройки терминала при работе с ARCUS NET Server (подробнее см. Ingenico Net Server руководств по эксплуатации)	YES/NO
TERMINAL-IP		Значение в формате xxx.xxx.xxx.xxx
MASK		Значение в формате xxx.xxx.xxx.xxx
GATEWAY		Значение в формате xxx.xxx.xxx.xxx
 Если значение параметра IP через кассу выставлено «NO», и работа осуществляется НЕ через NET сервер, сетевые настройки терминала задаются в коммуникационном скрипте! В этом меню их задавать НЕ нужно!		
#Timeout - настройка таймаутов		
DIALOG-TIMEOUT	Таймаут отображения диалогов	Значение задается в миллисекундах, значение по умолчанию 12000
DATA-TRANSMIT-TIMEOUT	Таймаут передачи данных	Значение задается в миллисекундах, значение по умолчанию 1000
CARD-READ-TIMEOUT	Таймаут чтения карты	Значение задается в миллисекундах, значение по умолчанию 6000
#Dialogs - настройки диалогов		
DIALOGS-ON-CASHREG	Отображать диалоги на экране кассы	YES/NO (для терминалов ПО SV_05 значение по умолчанию – NO)
#SpecialMask - настройки маскирования номера карты при печати на чеке		
SPECIAL-MASK	Активация доп. маскирования	YES/NO
PAN-START	Количество отображаемых символов в начале номера карты	0-6, но не больше, чем указано в TMS
PAN-END	Количество отображаемых символов в конце номера карты	0-4 но не больше, чем указано в TMS
#PrintSettings - настройки печати		
PRINTER	Выбор принтера для печати чека	CASHREG/INTERNAL/BOTH
PRINTER-ENCODING	Кодировка печати	DEFAULT/CP866/CP1251
RECEIPT-WIDTH	Ширина чека в символах	24-48
FORMATTING	Разрешить спец. Символы	DON'T-USE-ESC/USE-ESC
#AdditionalSettings - дополнительные настройки		
SEND-PAN-SHA1	Передача SHA1 номера карты	YES/NO
CHUNK_SIZE	Размер фрагмента при использовании функционала «Фрагментация данных» (опционально, ARCUS 2.0)	25 - 512
USE-ACTIVE-KEYS	Активация вызова меню кассира по кнопке «3» в режиме работы с кассой, значение по умолчанию - NO), (NW TPTP, ARCUS 2.0)	YES/NO

DEBUG-TRACE	(NW_TPTP_15) Настройка вывода трассы в лог ARCUS2 на DEBUG версиях ПО.	YES/NO
	 Для режима АДМ при подключении терминала по низкоуровневому протоколу ARCUS 2 параметр должен быть выставлен в значение «НЕТ».	

Пример файла

```
#Interface
PORT=COM0
PORT-SPEED=2400
DATA-BITS=8
PARTITY=NONE
STOP-BITS=1
ARCUSSERVER -IP=192.168.0.101
ARCUSSERVER -PORT=9401
BLUETOOTH-PIN-LENGTH=4
USE-ARCUS-ID=YES
USE-TMSID=YES
ARCUS-ID=123456790

#Network
IP-ON-CASHREG=YES
STATIC-IP=YES
TERMINAL-IP=192.168.0.10
MASK=255.255.255.0
GATEWAY=192.168.0.1
#Timeouts
DIALOG-TIMEOUT=12000
DATA-TRANSMIT-TIMEOUT=1000
CARD-READ-TIMEOUT=6000

#Dialogs
DIALOGS-ON-CASHREG=YES

#SpecialMask
SPECIAL-MASK=YES
PAN-START=4
PAN-END=4

#PrintSettings
PRINTER=CASHREG
PRINTER-ENCODING=DEFAULT
RECEIPT-WIDTH=24
FORMATTING=DON'T-USE-ESC

#AdditionalSettings
SEND-PAN-SHA1=YES
```

1_2_3 Файл настроек ARCUS.CFG для Режимы работы с кассой терминального приложения Unipay (CompassPlus, SPDH)

/ Таблица 6. Описание файла настроек ARCUS.CFG UNIPAY

Параметр	Описание	Доступные значения
Настройки интерфейса – раздел #Interface		
PORT	Порт подключения	COM0/USB/COM_SL/COMU/ETH/BLUETOOTH
PORT-SPEED	Скорость передачи данных	2400/9600/ 19200 /115200
ARCUSSERVER -IP	IP для подключения к ARCUS Net Server (подробнее см. Ingenico Net Server руководств по эксплуатации)	Значение в формате xxx.xxx.xxx.xxx

Параметр	Описание	Доступные значения
ARCUSSEVER -PORT	Номер порта для подключения к ARCUS Net Server (подробнее см. Ingenico Net Server руководств по эксплуатации)	Значение в формате xxxx
BLUETOOTH-PIN-LENGTH	Длина ПИН кода для подключения по Bluetooth	4-8
Настройки сети – раздел #Network		
IP-ON-CASHREG	IP через кассу	YES/NO
STATIC-IP	Сетевые настройки терминала при работе с ARCUS NET Server (подробнее см. Ingenico Net Server руководств по эксплуатации)	YES/NO
TERMINAL-IP		
MASK		
GATEWAY		
 Если значение параметра IP через кассу выставлено «NO», и работа осуществляется НЕ через NET сервер, сетевые настройки терминала задаются в коммуникационном скрипте! В этом меню их задавать НЕ нужно!		
Настройки диалогов – раздел #Dialogs		
DIALOGS-ON-CASHREG	Отображать диалоги на экране кассы	YES/NO
Настройки маскирования номера карты при печати на чеке – раздел #SpecialMask		
SPECIAL-MASK	Активация доп. Маскирования	YES/NO
PAN-START	Количество отображаемых символов в начале номера карты	0-6, но не больше, чем указано в TMS
PAN-END	Количество отображаемых символов в конце номера карты	0-4 но не больше, чем указано в TMS
Настройки печати – раздел #PrintSettings		
PRINTER	Выбор принтера для печати чека	CASHREG/INTERNAL
RECEIPT-WIDTH	Ширина чека в символах	24-48
FORMATTING	Разрешить спец. Символы	DON'T-USE-ESC/USE-ESC
Дополнительные настройки – раздел #AdditionalSettings		
AUTH-CODE-FROM-CASHREGISTER	Получение кода авторизации с кассы	YES/NO
PRINT-JOURNAL	Настройки печати журнала операций	YES/NO/MISMATCH

Пример файла

```
#Interface
PORT=COM0
PORT-SPEED=19200
ARCUSSERVER -IP=192.168.0.101
ARCUSSERVER -PORT=9401
BLUETOOTH-PIN-LENGTH=4
```

```
#Network
IP-ON-CASHREG=YES
STATIC-IP=NO
TERMINAL-IP=192.168.0.10
MASK=255.255.255.0
```

```
GATEWAY=192.168.0.1

#Dialogs
DIALOGS-ON-CASHREG=YES

#SpecialMask
SPECIAL-MASK=YES
PAN-START=6
PAN-END=4

#PrintSettings
PRINTER=INTERNAL
RECEIPT-WIDTH=24
FORMATTING= USE-ESC

#AdditionalSettings
AUTH-CODE-FROM-CASHREGISTER=YES
PRINT-JOURNAL= YES
```

1_3 Интеграция с ККМ

1. При проведении безналичного платежа ПО ККМ формирует данные по платежу. Данные могут различаться.

- Обязательные параметры:

- код операции;
- сумма операции в минимальных единицах валюты (если сумма операции не передается, выполнение транзакции прерывается);
- код валюты;

- Необязательные параметры:

- код авторизации [завершение расчета по предавторизации];
- код ссылки RRN [завершение расчета по предавторизации].

В ряде случаев часть параметров может не передаваться с кассы, в таком случае при необходимости на экране терминала отобразится запрос для ручного ввода этих параметров.

2. Передача данных в ARCUS 2.1 может выполняться следующими способами:

- Через структуру UserAuthIntFull:

- a. Собранные кассовым ПО данные по операции записываются в структуру UserAuthIntFull.
- b. Затем ПО кассы вызывает функцию ProcessOWFull и в качестве параметра передает указатель на заполненную структуру UserAuthIntFull. Вызов функции ProcessOWFull является блокирующим. Управление диалогами с клиентом и кассиром передается приложению ПИН-пад.

- Через отдельные методы:

- a. Create – возвращается объект, с которым производятся дальнейшие действия.
- b. Set – устанавливается необходимое значение: сумма, код валюты и т.д.
- c. Run – запускается операция на POS терминале.
- d. Get – получение данных, после завершения работы POS терминала.

3. В ходе выполнения безналичной операции по банковской карте из ПО ПИН-пад может быть затребован ввод дополнительных данных:

- код CVC2/CVV2
- PIN код
- Код авторизации
- RRN

Так же могут быть инициированы другие диалоги, связанные с подтверждением/отказом или выбором параметров операции из списка.



Интерфейс ведения диалогов определяется настройками модуля ARCUS2 и ПО ПИН-пад и может быть назначен на использование клавиатуры и дисплея ККМ или ПИН-пад.

Вся текстовая информация передается в кодировке, зависящей от ОС ККМ (WIN1251 для Windows, DOS866 для DOS и Linux).

4. Возможны следующие варианты диалогов с кассиром и держателем карты:

- Вопросы, на которые необходимо ответить Да/Нет;
 - Выбор из списка значений;
 - Приглашение на ввод текстовой, или числовой информации;
 - А также информационные сообщения и сообщения об ошибках.
5. Возвращаемым значением функции **ProcessOWFull ()** и **Run ()**, является код ответа от сервера (response code). При положительном результате выполнения операции возвращается «000», все остальные значения указывают на то, что операция не выполнена.



Коды ответов зависят от типа используемого процессинга.

ARCUS2 так же использует внутренние коды ответов, например:

- **992=ОПЕРАЦИЯ ПРЕРВАНА КАССИРОМ ИЛИ КЛИЕНТОМ**
 - клиент или кассир отказались от проведения операции в момент ведения диалогов на ПИН-паде;
 - чиповая карта «отказалась» от проведения финансовой операции (напр. ответила кодом AAC);
 - в параметрах вызова модуля ARCUS от кассовой программы получены недопустимые параметры (например, несуществующий в параметрах терминала КОД ВАЛЮТЫ)
- **998=ОШИБКА СВЯЗИ, ПОЗВОНТЕ В БАНК - попытка IP-соединения ПИН -пада с сервером авторизации завершилась:**
 - ошибкой;
 - превышением тайм-аутов;
 - обрывом соединения и пр.
- **999=НЕТ СВЯЗИ С ПИН-ПАДОМ - проблема обмена данными по RS232/USB интерфейсу между ПИН-падом и модулем ARCUS на кассе:**
 - обрыв проводов или неисправность разъемов
 - проблема с питанием ПИН-пада
 - USB ПИН-пад по инициативе кассового ПК перешел в режим энергосбережения (определяется настройками управления питанием в BIOS компьютера, моделью используемого USB кабеля для подключения ПИН-пада к ПК)

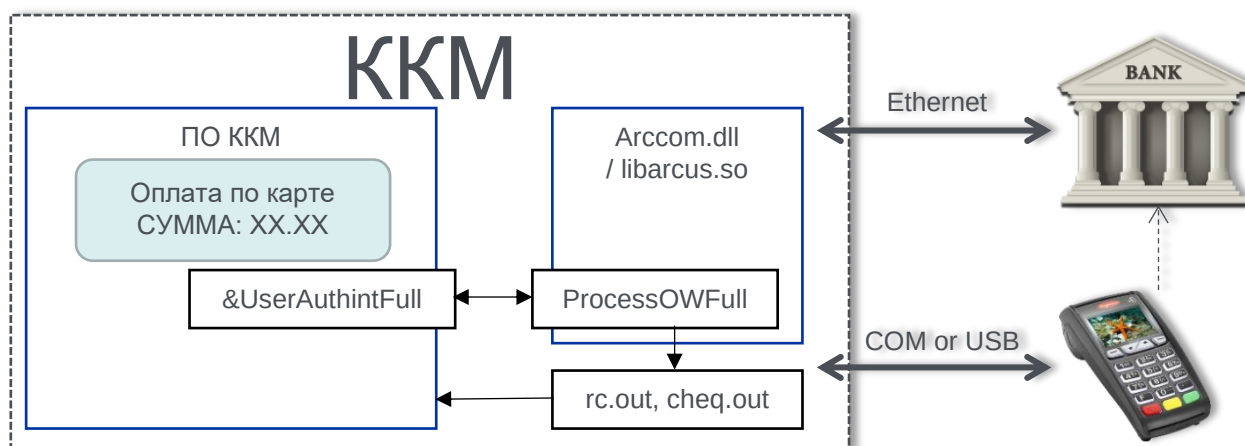


Рис. 3. Схема взаимодействия кассового ПО с модулем ARCUS 2.1 с использованием ArcCom.DLL

Результатом работы модуля ARCUS 2.1 являются формирование двух (трех) текстовых файлов:

- Файл кода ответа (например, RESULT_FILE или rc.out), содержащий трехзначный код ответа от банка. При положительном результате выполнения операции возвращается «000», все остальные значения указывают на то, что операция не выполнена (в операции отказано).
- Файл - чек (например, cheq.out), содержащий данные по операции, для печати на принтере в кодировке, соответствующей ОС. Формат и содержание информации в файле чека банковской операции определяется настройками специалистов Банка при параметризации ПО терминала. Файл чека содержит только печатные символы и не управления форматом печати и ESC- последовательностей.
- Файл данных [опция] (имя файла для ОС Windows и DOS output.dat, для Linux output.out), содержащий строки с данными по операции, в кодировке, соответствующей ОС (файл текстовый). Назначение и формат данных по строкам фиксированный (см. п. 1_3_1 Формат и содержание файла данных output.dat).

Пример

В случае проведения ARCUS2 безналичной транзакции в файле чека будет содержаться банковский чек.

При возникновении программно-аппаратного сбоя в файле кода ответа будет содержаться код ошибки, а в файле чека технический чек с кодом и расшифровкой.

```
=====пример= чека=====
ОШИБКА
Код ошибки:999 (999)
Нет кода ответа
=====конец примера=====

=====пример=====
999
=====конец примера=====
```

```
=====пример= чека=====
ОАО "Банк ВТБ
Северо-Запад"
пр. М. Говорова, д.52
Банк ВТБ Северо-Запад
Терминал R0000009
16/07/10 13:29 Кассир:0

Оплата Покупки
0.02 RUR
Код ответа: 1
Код авторизации 404365 B
Карта 4*****6016
VISA 01/11
CARD TEST
Чек #114 1.1.4
Операцию подтверждаю:
(Подпись клиента)
=====конец примера=====
```

Рис. 4. Банковский чек (Ошибка, успешная оплата)

1_3_1 Формат и содержание файла данных output.dat

Файл output.dat может содержать различные данные о транзакции для различных интеграций.

Пример содержания файла:

000	- код ответа (NNN 000 - 999)
*****2322	- PAN
230331	- срок действия
100100	- сумма (в минимальных единицах валюты)
INGE0800	- Terminal ID (идентификатор терминала в платежной системе)
213355	- код авторизации (для офлайн операций – последние 6 цифр RRN)
VISA	- тип карты
0416	- дата операции
170313	- время операции
A0000000031010	- AID
visa	- метка приложения (APPLICATION LABEL)
0080008000	- TVR
УСПЕШНО	- текстовое сообщение
14	- НОМЕР ЧЕКА
4816D0C8FF699AE1B7638672E880D7943DC95117598971170C1C9D921F9AF09C	- криптограмма данных карты
1234567890	- Transaction ID

В результате транзакции терминал формирует чек, который можно распечатать на внутреннем принтере терминала или на принтере чеков, подключенном к кассе.

Формат чека при печати на встроенном принтере определяется настройками специалистов Банка при параметризации ПО терминала.

В случае если для печати используется принтер, подключенный к кассе, то данные для печати можно получить несколькими способами:

- напечатать содержимое файла чека **cheq.out** и вывести на экран данные из файла ответа **rc.out**;
- кассовое ПО самостоятельно формирует чек безналичной операции на основании данных из структуры **UserAuthIntFull**, из значений свойств, полученных при помощи вызова метода Get или на основании результирующего файла данных **output.dat**.

Обязательный набор данных на кассовом чеке, при проведении операции по банковской карте:

- Код авторизации;
- Валюта операции;
- Место для подписи клиента;
- Наименование, страна и город (адрес) торгового предприятия;
- Код ответа и расшифровка причины отказа;
- Тип карты и дата истечения карты;



Для продукта SV_05 вместо Типа карт выводится Application Label, если он не нулевой.

- Наименование обслуживающего банка;
- EMV данные (при проведении операции по чиповой карте).

В случае использования в ПО ККМ файла чека, выполнение требований международных платежных систем по наличию и содержанию информации на чеке операции по карте осуществляется специалистами Банка при параметризации ПИН-пад.

В случае использования в ПО ККМ данных от модуля ARCUS 2.1 или файла данных выполнение требований международных платежных систем по наличию и содержанию информации на чеке операции по карте должно выполнять ПО ККМ.

Исключение, не требующее печати чека, составляют коды 99x (x = 1...9) — коды технологических сбоев или отказов, сгенерированные ARCUS2.1, которые не являются кодами ответа сервера Банка по безналичной операции.

1_3_2 Дополнительные данные платежа (ARCUS2.0)

В ARCUS2 CAP реализована передача специфичных данных, характерных для топливных карт и карт лояльности. Для активации функционала в файле cashreg.ini задайте параметр OUTPUT_FORMAT_V=1. В результате при отправке данных о платеже будет сформирован результирующий файл, указанный в параметре OUTPUT_FILE в cashreg.ini, в следующем формате:

[параметр]=[значение]

/ Таблица 7. Параметры файла данных

Параметр	Значение
RESPONSE	Код ответа
TEXTMESSAGE	Расшифровка ответа
OPERATION	Тип операции
AMOUNT	Сумма операции
CURRENCY	Валюта операции
PAN	Номер карты лояльности (или криптограмма номера банковской карты)
EXPIRY	Срок окончания действия карты
DATE	Дата проведения операции
TIME	Время проведения операции
AUTHCODE	Код авторизации
TERMINALID	ID терминала;
LT_RQ_DSCNT	Сумма в копейках, которую клиент хочет оплатить\возвратить баллами
LT_BALANCE	Баланс баллов на счете клиента
LT_MINUS	Сумма списанных баллов
LT_PLUS	Сумма начисленных баллов
LT_MAX	Максимальная сумма, которая может быть оплачена по данной корзине с учетом баланса баллов
LT_ADD	Сумма дополнительно, в рамках маркетинговой акции, начисленных баллов
LT_CL_INFO	Маркетинговое сообщение

Параметр	Значение
LT_CNT_CRD	Подключенные в процессе активации карты лояльности
LT_DISCOUNTS	Информация по списанию (или возврату) со счета лояльности по каждой товарной позиции через "\", в копейках в очередности размещения товаров в корзине в запросе
LT_CARD_NUMBER	Номер карты лояльности для кобрендинговых карт
PT_LIM_AMNT	Остаток по сработавшему ограничителю (в валюте транзакции) суммы, по каждой товарной позиции через "\", в формате [Товарный код]:[Остаток]
PT_LIM_UNIT	Остаток по сработавшему ограничителю (в единицах измерения) количества, по каждой товарной позиции через "\", в формате [Товарный код]:[Остаток]

Пример

LT_RQ_DSCNT=12345.

1_4 Вызов ARCUS 2.1

Интеграция с ККМ на базе MS Windows выполняется при помощи библиотек Arccom.dll и libarcus.so (см. раздел Интеграция с ККМ).

Для проверки работы модуль ARCUS 2.1 поставляется вместе с эмулятором кассовой программы **CommandLineTool**. Данная утилита используется для полноценной проверки работы терминала, подключенного к ПК с ОС Windows.

- **/?** или **/help** - вывести список всех доступных ключей;
- **/a[Сумма]** – сумма операции в минимальных единицах валюты без точки;
- **/arcusid** - передача ARCUS_ID при использовании Arcus IP Server;
- **/b[TraceID]** – ID предавторизации (TE_02, OW);
- **/c[Код валюты]** – код валюты операции;
- **/e[Данные карты]** - шифрованные данные о карте, 64 символа;
- **/f[Original Date/Time]** – дата и время оригинальной Предавторизации (TE_02);
- **/h[Original Amount]** – общая авторизованная сумма (TE_02);
- **/i[Идентификатор терминала]** - ID терминала до 9 символов (определяется TMS);



Если в ключе **"/i"** передаются данные длиной в 1 байт, то это номер отдела для ПО UNIPAY

- **/l[код лояльности]** - код системы лояльности (UNIPAY, TPTP_15, NW_TPTP_15 с декабря 2020 в т.ч. для работы с лояльностью Лента);
- **/no_ext_log** - отключение создания файлов output_ex.txt, output_ex.dat (NW_TPTP_15);
- **/o[Код операции ККМ]** – Оплата, Отмена, Возврат и т.д. Цифровое значение соответствует коду операции в файле «ops.ini»;
- **/p** – название платежной системы при оплате QR кодами (Yandex или Alipay);
- **/q[номер чека]** – номер чека;
- **/r[Ссылка]** — RRN, до 12 символов;
- **/t[Трек 2]** – второй трек пластиковой карты;
- **/v[Код авторизации]** - код авторизации, 6-9 символов;
- **/x[TransactioID]** -идентификатор транзакции (TPTP);
- **/z[Сумма наличных]** - сумма наличных для операции Выдача наличных;
- **--dialog-direct** - запуск специальной команды DialogDirect (должен быть настроен в TMS)
- **--origin-oper-type=N** – тип оригинальной операции при отмене вне смены (OW, Fuel+);
- **--payment-data=(payment_data)** - данные платежа в сторону третьих лиц (SPDH Fid "9" Subfid "i");
- **--port=(PORT)** - номер порта для работы ARCUS;
- **--print-file=(имя файла)** - название текстового файла для печати на чеке терминала (без форматирования, NewWay);
- **--repeat=(N)** - повтор последней операции N раз (служебная команда - без выгрузки библиотеки).

Ключи, доступные только в Arcus 2.0:

- **/dlsh** – передача дополнительных данных от кассы (только для продуктов с функционалом DLS, Arcus 2.0);
- **/gb** – проверка работы сканера штрих кодов терминала iSMP (Только для продукта CTL_03, Arcus 2.0);
- **/grN** – вызов печати дубликата чека. N – Номер чека (SV_05, Arcus 2.0);
- **/j[CashReceiptNumber]** - номер кассового чека
- **/pd** – данные для проведения платежа в сторону третьих лиц (TPTP_04);
- **--payment-data** - данные для проведения платежа в сторону третьих лиц (SPDH Fid "9" Subfid "i")
- **--bonus=NNNN** - бонусы для списания в минимальных единицах валюты (1000 = 10 бонусов).
- **--card-type=1** – признак типа карты (1 - банк, 2 - топливо, 4 - лояльность).
- опционально, NewWay)
- **--goods-file=[полный_путь_и_имя_файла]** – путь к файлу со списком товаров (в ARCUS2 CAP доступна функция передачи информации о товаре).
- **--no-icc-removing** выполняет команду с кассы без извлечения карты, если использовалась чиповая карта.



Обязательными параметрами для передачи являются: Код операции, валюта и сумма. Другие параметры могут отсутствовать. В случае необходимости терминал запросит недостающую информацию (например, номер RRN (Retrieval Reference Number) для операции «Отмена»). Кассир вводит данные по приглашению на экране терминала или кассы.



Специфика работы unattended-устройств (iUN250, i9500 и др.) не подразумевает ввод дополнительной информации клиентом, поэтому при работе с такими терминалами необходимо передавать все необходимые параметры, например, «Код валюты» и «Сумму» при операции «Оплата».



CommandLineTool поддерживает использование флага /debug – вывод информации о содержимом полей структуры AuthIntFULL в текстовом виде в файл output_ex.dat, который размещается в соответствии с настройкой аркуса Output & Exchange path.

Дополнительные ключи для продукта SV_05, только Arcus 2.0

- **/tD** - основной номер счета (PAN - Primary account number);
- **/g1**- получить хеш карты (SHA);
- **/g2** - получить хеш карты (SHA2);
- **/g3** - получить PAN;
- **/gb** - получить штрих-код;
- **/gd** - получить конфигурацию диалогов (0 - на ПИН-паде, 1 - на кассе);
- **/gp** - получить зашифрованный ПИН блок;
- **/gs** - получить серийный номер;
- **/gt** - получить track2;
- **/k** - установить ключ шифрования ПИН;
- **/m** - установить мастер ключ шифрования ПИН.

Пример использования:

- Оплата на 123,45 рублей, код валюты 643.

```
CommandLineTool.exe /o1 /c643 /a12345
```



Номер карты и срок действия может быть введен вручную с клавиатуры ПИН-пад. Для этого в момент приглашения на экране ПИН-пад «Вставьте карту» необходимо нажать желтую кнопку «С».

- Передача данных трек 2, например, при чтении магнитной полосы на клавиатуре кассы. Оплата на 123.45 рублей, код валюты 643, трек карты «10000230000000034=06062011021300000641».

```
CommandLineTool.exe /o1 /c643 /a12345 /t;1000023000000034=06062011021300000641?
```

Данные трек 2 предваряются символом «;» и завершаются «?».

В случае эмуляции ручного ввода данные передаются следующим образом:

М[Номер карты]=[срок действия в формате ГГММ]?

```
CommandLineTool.exe /o1 /c643 /a12345 /tm1000023000000034=1106?
```



ARCUS 2.1 исключен из области сертификации Payment Application Data Security Standards (PA DSS), так как при правильной настройке не хранит и не передает данные платежных карт в нешифрованном виде. Использование специальных версий программы CAP с возможностью считывания трека карты на ККМ, требует проведение сертификации ПО ККМ.



При использовании функционала HRS для проверки шифрованных данных о карте, шифрованные данные передаются после ключа «/e».

Пример запуска:

```
CommandLineTool.exe /o1 /a2000
```

```
/e3AB5E25D6091053EAB71AFAE303B5327F3D713043819649F3D20D11F78A73811
```

Получение шифрованных данных настраивается в меню администратора - доп. настройки – РЕЖИМ HRS (использовать шифрование параметров).

Данные шифруются с помощью 3DES CBC. Ключ для шифрования вводится в открытом виде в меню администратора – ПИН-пад - ARCUS КЛЮЧ.

1_5 Файл инициализации Cashreg.ini

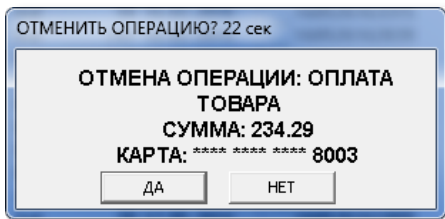
/ Таблица 8. Переменные в Cashreg.ini

Переменная	Описание
Параметры порта	
PORT	Настройки порта имя последовательного порта (системное имя последовательного порта в среде Windows). При работе с ARCUS 2 CAP рекомендуется использовать номера COM-портов с 1 по 9. В случае необходимости номер порта может быть и выше, тогда в cashreg.ini его необходимо задать как PORT=//./COM[Номер_порта_от_10_до_99] или при помощи диспетчера устройств Windows: Параметры порта - Дополнительно - Номер COM порта.
ADDR_IP	IP адрес ArcusIPService /ArcusNETServer /IngenicoNETServer.
PORT_IP	IP порт для подключения к ArcusIPService /ArcusNETServer /IngenicoNETServer.
ARCUS_ID	Идентификатор для ArcusNetServer или IngenicoNETServer, при использовании режима authType="arcusid".
SPEED	Скорость обмена с портом. Доступные значения 2400, 4800, 9600, 19200, 38400, 57600, 115200. При работе по RS232 рекомендуемое значение 9600, при работе по USB рекомендуемое значение 115200.
BYTE	Количество бит данных.
PARITY	Четность (N- нет контроля четности, E – контроль по четности, O – контроль по нечетности).
STOP	Стоп биты.

Переменная	Описание
TIMEOUT	Таймаут на получение одного сообщения от устройства PIN Pad (в миллисекундах). Параметр используется при работе терминала в режиме работы с кассой и «Диалоги на кассе – ДА».
PPAD_IDLE_TO	Таймаут для следующих действий: • изъятие карты из чип-ридера на терминале; • окончание сессии работы с чип-картой; • таймаут нажатия кнопок в окне диалогов.  При инициализации с терминала сессии с кассой параметр "Таймаут диалога" устанавливает максимальное значение времени, в течение которого терминал ожидает ответ от кассы на запущенный диалог (режим диалоги на кассе). После запуска диалога на кассе идет отсчет 120 сек. (таймаут задан в коде и не изменяется), поэтому параметр "Таймаут диалога" на терминале должен быть меньше либо равен этому значению.
Настройки файлов	
OPERATION_INI_FILE	Имя файла настройки конвертации кодов операций. Значение по умолчанию - ops.ini.
RC_RESOLVE_FILE	Имя файла с тестовыми расшифровками кодов ответа. Значение по умолчанию - rc_res.ini.  Для проекта МН: RC_RESOLVE_FILE_LT - переменная для файла с расшифровкой кодов ответа для карт лояльности RC_RESOLVE_FILE_PT - переменная для файла с расшифровкой кодов ответа для топливных карт.
DEFAULT_RC_STRING	Расшифровка неизвестного кода ответа (по умолчанию).
USEORIGINALRC	Если флаг установлен – для расшифровки используется код ответа до конвертации, если нет – используется уже сконвертированное значение (см. RC_CONVERT_FILE и rc_conv.ini).
RC_CONVERT_FILE	Имя файла настройки конвертации кодов ответа. Значение по умолчанию - rc_conv.ini.
CHEQ_FILE	Файл, в который будет выводиться чек. Значение по умолчанию - cheq.out.
RESULT_FILE	Файл, в который будет выводиться код ответа. Значение по умолчанию - rc.out.
CANCEL_CH_FILE	Файл, в который будет выводиться чек при автоматической отмене в случае сбоя. Значение по умолчанию - auto_can.out.
CANCEL_RC_FILE	Файл, в который будет выводиться код ответа при автоматической отмене в случае сбоя. Значение по умолчанию - can_rc.out.
NO_CHECK_LAST_OP	Параметр для отключения функции отмены операции, если она завершилась неудачно в Аркус сессии (см. CANCEL_RC_FILE)
CANCEL_LAST_OP_AUTO	всегда отменяет последнюю неудачную операцию в Аркус сессии (см. CANCEL_RC_FILE).
DATETIME_FORMAT	Формат вывода даты/время в файл лога, может принимать значение: • 1 - YYYYMMDD HHMMSS.SSS; • 2 - только секунды; • по умолчанию - YYYY/MM/DD HH:MM:SS.
OUTPUT_FILE	Название файла, в котором передаются дополнительные данные на кассу, характерных для топливных карт и карт лояльности в формате

Переменная	Описание
	[параметр]=[значение]. Для активации функционала в файле cashreg.ini задайте параметр OUTPUT_FORMAT_V=1.  С августа 2019 года этот параметр так же используется для функционала передачи данных платежа для с поддержкой файла шаблона (OUTPUT_FILE_TEMPLATE)
OUTPUT_FILE_TEMPLATE	Файл с шаблоном, в котором содержится список и последовательность выходных данных для функционала передачи данных платежа Значение по умолчанию - output.tpl.  Если параметр OUTPUT_FILE_TEMPLATE не задан, или файл шаблона отсутствует, данные платежа передаются на кассу при помощи файла, указанного в настройке OUTPUT_FILE, без применения шаблона в следующем порядке: RESULTCODE PAN TERMINALID AUTHCODE APPLABLE MESSAGE AMOUNT RRN SYMBOL_ACTION LOYALTY_CODE TRANSACTIONID
Настройки логирования	
USEAPPLOG	Активация функционала вывода краткой отладочной информации в файлы логов операций - файлы сохраняются в папку «Logs» или заданную в параметре LOG_FOLDER_NAME. Для автоматического удаления файлов логирования задайте значения параметров NDAYS, NFILES и NSIZE.  Лог-файлы хранятся в папке до тех пор, пока значение хотя бы одного из параметров NDAYS, NFILES и NSIZE не превысят ограничения. Проверка параметров проводится при любом успешном вызове библиотеки. Расположение папки с лог-файлами задается в настройках «working directory» и может отличаться от стандартного пути установки «C:\ARCUS2».
NDAYS	Время хранения лог-файла в днях. Значение по умолчанию – 130. Максимальное значение 1825 (5 лет).
NFILES	Максимальное количество лог-файлов в папке ARCUS2/logs (удаление файлов выполняется после проверки параметра NDAYS). Значение по умолчанию - 0 (без ограничений).
NSIZE	Максимальный совокупный объем лог-файлов в папке ARCUS2/logs в МБ (удаление файлов выполняется после проверки параметра NDAYS и NSIZE). Значение по умолчанию - 0 (без ограничений).
LOG_FOLDER_NAME	Название директории с логами, по умолчанию <Logs>.  Лог-файлы хранятся в папке до тех пор, пока значение хотя бы одного из параметров NDAYS, NFILES и NSIZE не превысят ограничения. Проверка параметров проводится при любом успешном вызове библиотеки. Расположение папки с лог-файлами задается в настройках «working directory» и может отличаться от стандартного пути установки «C:\Arcus2».

Переменная	Описание
LOG_FOLDER_MODE (L)	Битовая маска установки мод. доступа к папке в восьмиричной системе, по умолчанию "770". (только для Arcus2.1 Linux).
DISABLE_NAK	Отключение вывода в лог файл команд NAK.
DEBUG	В лог файл будут выводиться все "распознанные" теги, в виде <название поля>: <строка данных> [<бинарное представление>].
NOTAGSLOG	Отключение логирования тегов из команд Send/RecvOWTags.
NOFILELOG	Отключение логирования результатов файловых операций.
Настройки безопасности	
NO_PA_DSS	Если переменная NO_PA_DSS определена, ARCUS2 обрабатывает ТРЕК2 карты, переданный из кассовой программы, иначе данные ТРЕК2 игнорируются, и карта читается только на ПИН-паде.
NOTIMESYNC	Деактивация функционала синхронизации времени с ККМ.
USEPPADTRACE	Активация функционала вывода подробной трассировки отладочной информации в файлы для Debug версий ПО.
USEJL	Активация функционала использование нескольких юридических лиц (виртуальных терминалов) на терминале.
	 <u>Логика работы</u> В одной папке с библиотекой «ArcCom.dll» создайте файл «jl.txt», в файле укажите значение TID виртуального терминала (для проектов ISO) или порядковый номер виртуального терминала (для проектов Compass+/SPDH) данного юр. лица. Выберите один из следующих вариантов логики работы: - создайте разные модули ArcCom.dll размещенные в разных папках вместе с разными файлами jl.txt. Внешнее ПО кассы будет вызывать разные модули; - внешнее ПО кассы должно переписывать содержимое jl.txt перед вызовом библиотеки.
USEJL	 С декабря 2020 года для UniPay_SPDH в режиме работы с кассой изменена логика обработки TerminalID виртуального терминала в файле jl.txt: - если TerminalId имеет размер 1 байт, то его значение рассматривается как бинарный индекс виртуального терминала и располагается в теге 1F73; - если значение не цифровое, значение тега пустое (нулевой длины); - если значение имеет длину от 2 до 8 байт, то оно передается в теге 9F1C. Если функционал USELJ отключен, идентификатор виртуального терминала определяется по значению, переданному в ключе /i при вызове операции.
Настройки кодировки (для файлов *.out)	
	Для поддержки возможности менять кодировку установите библиотеку «iconv». Для определения доступных кодировок выполните команду iconv –list. Должны быть заданы все три параметра!
PPCHARSET	Кодировка передачи сообщений в терминал (ТОЛЬКО CP1251).
OPCHARSET	кодировка сообщений на экране кассы (CP1251, UTF-8, cp866, KOI8-r);
PRINTERCHARSET	Кодировка сообщений при печати в файл cheq.out и тд.(ТОЛЬКО cp866).
Настройки подтверждения отмены последней операции (в режиме диалоги на кассе)	

Переменная	Описание
DEF_OPDET_RESPONSE	Переменная содержит настройку отображения на кассе диалога подтверждения проведения ОТМЕНЫ ПОСЛЕДНЕЙ ОПЕРАЦИИ: если переменная не определена, то на экране кассы будет отображаться окно с возможностью выбора ответа – Да/Нет; если переменной присвоено значение «YES» или «NO», диалог отображаться не будет, а в качестве ответа будет автоматически применяться значение, указанное в переменной. 
Дополнительные настройки	
USE_PRINTER	Переменная содержит имя принтера Windows, если его необходимо использовать для печати банковского чека. Значение по умолчанию NONE – принтер не используется.
INFO_DIALOG	(Только для работы с ПО Инфоточек) Активация функционала отображения диалогов payofINFO. Для активации функционала в Доп. настройках TMS должен быть установлен флаг «Инфо-диалоги».
USE_DLG_DIRECT	Активация функционала Dialog Direct – возможность вывода сообщений и меню на экран терминала с кассы, считывание ответов пользователя (анализ нажатия на кнопки) и отправка ответа на кассу. В TMS должны быть установлены соответствующие значения параметров;
USE_COMPORT_WARNING	информационное сообщение, если невозможно открыть COM порт (только для SV_05).
GET_SERIAL_OPID	Команда для определения серийного номера терминала при запросе с кассы (только для SV_05, Arcus 2.0)
USE_UNIPAY	Индекс тэга для номера чека для работы с ПО Universal EMV POS (Unipay)
AUTH_CODE_FULL	Признак для определения типа кода авторизации (TPTP_09): 1 - (значение по умолчанию) код авторизации дополняется буквой через пробел (если она содержится в теге \x1F\x2D); 0 - код авторизации не дополняется.
NODIALOGS	Включение/выключение диалогов на кассе.
PREPAYMENT	Переменная активирует особый режим работы с кассой для работы с приложением Fuel+ (подробнее см. руководство Администратора Fuel+, Arcus 2.0)
NO_HRS	Отключает использование HRS (для интеграции со структурой без данного поля)
AUTODETECT_OPS	(NW_TPTP_15) Активация функционала выбора операции по специальным критериям.  Для корректной работы функционала в параметре OPS_CONV_INI_FILE задайте файл соответствия TID и файлов ops.ini, в параметре ID_CONV_INI_FILE - файл соответствия дополнительных и реальных TID).
RRN_TRANSACTION_ID	Активация функционал использования TransactionID в качестве RRN для Модуль AC1. Arcus 2.0 12321
DISABLE_NAK	Отключение в Аркус2 передачу команды NAK при истечение таймута ожидания команды от пин-пада . (рекомендуется использовать эту настройку при промышленной эксплуатации Аркус2 и на стендах приемочного тестирования ИКР)

1_6 Настройки ARCUS 2.1 CAP

1_6_1 Файл ops.ini для ISO8583

Файл соответствия кода операции, передаваемого с ККМ и кода операции ПИН-пад (пример для протоколов на базе ISO8583 - OpenWay, SmartVista, Tieto).

/ Настройки операции:

Op code	Op class	Op id	Op Name
1 =	1,	1,	ОПЛАТА

- **Op code** - код операции для кассы
- **Op class** - класс операции
- **Op id** - ID операции

/ Таблица 9. Список операций для интеграции с ПО Universal EMV POS (NewWay)

Класс	ID	Название
ОСНОВНЫЕ ФИНАНСОВЫЕ ОПЕРАЦИИ		
1	1	ОПЛАТА ПОКУПКИ / ОПЛАТА ТОВАРА / ОПЛАТА
1	2	ОПЛАТА ПОКУПКИ DUAL
1	3	ПРЕДАВТОРИЗАЦИЯ
1	4	ЗАВЕРШЕНИЕ ПРЕДАВТОРИЗАЦИИ / ЗАВЕРШЕНИЕ РАСЧЕТА
1	5	УНИВЕРСАЛЬНАЯ ОТМЕНА
1	6	ОТМЕНА ADVICE (TIETO)
1	8	МИНИ-ВЫПИСКА
1	10	УНИВЕРСАЛЬНАЯ ОТМЕНА ADVICE
1	11	ВОЗВРАТ / ВОЗВРАТ ТОВАРА
1	15	CREDIT VOUCHER
БАНК, ПУНКТ ВЫДАЧИ НАЛИЧНЫХ		
1	7	ЗАПРОС БАЛАНСА (в т.ч. для АДМ SV_05)
1	12	ВЫДАЧА НАЛИЧНЫХ (в т.ч. для АДМ)
1	13	ВЫДАЧА НАЛИЧНЫХ DUAL
1	14	ВНЕСЕНИЕ НАЛИЧНЫХ (в т.ч. для АДМ и работы с депозитными машинами SV_05)
1	47	BALANCE ACC
ПЕРЕВОДЫ, ПЛАТЕЖИ		
1	16	ПЛАТЕЖ В СТОРОНУ ТРЕТЬИХ ЛИЦ (UTILITY_PAYMENT)
1	17	ПЛАТЕЖ BILLING_PAYMENT_SMS
1	18	ПЛАТЕЖ BILLING_PAYMENT_DUAL
1	40	ПЕРЕВОД Card2Card
1	41	ПЕРЕВОД Cash2Card
1	109	Оплата Alipay (TPTP_NW)
1	110	Возврат Alipay (TPTP_NW).
ФУНКЦИОНАЛ DLS		
1	84	ОПЛАТА НАЛ+БОН

Класс	ID	Название
1	85	ОПЛАТА БОН
1	86	ВОЗВРАТ НАЛ+БОН
1	87	ВОЗВРАТ БОН
1	88	БАЛАНС БОН
1	89	IDLE ACTION (запуск выгрузки офлайн операций для DLS SwS, подробнее см. Руководство Администратора Universal EMV POS, Arcus 2.0)
ФУНКЦИОНАЛ FUEL+ (Fuel+ OW_03, Arcus 2.0)		
1	108	ИНФО ПО ТОВАРАМ И ЛИМИТАМ (Fuel+ OW_03, Arcus 2.0)
9	41	ОБНОВЛЕНИЕ СПИСКА ТОВАРОВ (Fuel+ OW_03, Arcus 2.0)
ТЕХНИЧЕСКИЕ ОПЕРАЦИИ		
2	3	АВТООТМЕНА (ТЕХНИЧЕСКАЯ ОТМЕНА)
2	4	ЧТЕНИЕ МАГНИТНОЙ КАРТЫ
2	6	ТЕСТ СВЯЗИ С ХОСТОМ проверяется только возможность соединения с ним, без авторизации на хосте, отправки ему команды и печати чека
2	18	ТЕСТ СВЯЗИ EXCHANGE (ЗАПРОС - ОТВЕТ) тестирует связь с хостом (авторизуется, отправляет эхо запрос), после чего печатает чек (SV).
2	21	ИНФО ТЕРМИНАЛА
ОТЧЕТЫ / ИТОГИ		
2	0	ПОЛНЫЙ ЖУРНАЛ
2	1	СВЕРКА ИТОГОВ
2	10	КРАТКИЙ ОТЧЕТ
2	19	ОТПРАВКА ОТЛОЖЕННОЙ ОТМЕНЫ / СООБЩЕНИЯ (SV)
2	20	 Поддержка сканера штрих-кода терминала iSMP. Только для продукта с кодом CTL_03. Подробнее см. Universal EMV POS, руководство администратора.
ПЕЧАТЬ ДУБЛИКАТА ЧЕКА		
2	14	ДУБЛИКАТ ЧЕКА ПО НОМЕРУ (при работе с ПО PayDroid)
2	26	ПЕЧАТЬ ЧЕКА ПО НОМЕРУ, ВВОД КАССИРА (при работе с ПО PayDroid)
2	27	ПЕЧАТЬ ДУБЛИКАТА ПОСЛЕДНЕГО ЧЕКА
2	38	ПЕЧАТЬ ДУБЛИКАТА ЧЕКА TRANSACTION_ID
РАБОТА С МЕНЮ		
2	2	МЕНЮ КАССИРА
2	9	ПАРОЛЬ СТАРШЕГО КАССИРА
ПО С ФУНКЦИОНАЛОМ КОНТРОЛЯ КАЧЕСТВА		
2	28	КОНТРОЛЬ КАЧЕСТВА НАЛИЧНЫЕ
2	29	КОНТРОЛЬ КАЧЕСТВА ВОЗВРАТ
КОМАНДЫ ДЛЯ РАБОТЫ В NEWWAY С ПРОЦЕССИНГАМИ SPDH / COMPASS+		
2	0	ПОЛНЫЙ ЖУРНАЛ ТРАНЗАКЦИЙ
2	1	СВЕРКА ИТОГОВ (ПАКЕТ)

Класс	ID	Название
2	22	СВЕРКА ИТОГОВ (ДЕНЬ)  По умолчанию закрывает день на всех виртуальных терминалах. При определенных настройках конфигурации платежного приложения можно выполнить закрытие дня для конкретного виртуального терминала.
2	23	ЖУРНАЛ (СМЕНА)
2	24	ЖУРНАЛ (ДЕНЬ)
2	25	СВЕРКА ИТОГОВ (СМЕНА)
2	39	КРАТКИЙ ОТЧЕТ (ПАКЕТ)
2	40	КРАТКИЙ ОТЧЕТ (СМЕНА)
2	41	КРАТКИЙ ОТЧЕТ(ДЕНЬ)
2	42	ПОЛНЫЙ ОТЧЕТ (ПАКЕТ)
АДМИНИСТРАТИВНЫЕ ОПЕРАЦИИ		
3	1	СИНХРОНИЗАЦИЯ ВРЕМЕНИ
3	2	СИНХРОНИЗАЦИЯ ВРЕМЕНИ
3	4	СЕССИЯ TMS
3	5	МЕНЮ АДМИНИСТРАТОРА
3	8	СЕССИЯ TMS ПО DEFAULT СКРИПТУ
3	9	АКТИВАЦИЯ ТЕРМИНАЛА
ОПЕРАЦИИ С НАЛИЧНЫМИ (в TMS должен быть добавлен тип карт «Наличные»)		
4	89	ОПЛАТА НАЛ
4	90	ВОЗВРАТ НАЛ
ОПЕРАЦИИ ДЛЯ СПЕЦИАЛЬНЫХ ВЕРСИЙ ПО		
1	96	АКТИВАЦИЯ КАРТ ЛОЯЛЬНОСТИ
1	97	НАЧИСЛЕНИЕ БАЛЛОВ НА КАРТУ ЛОЯЛЬНОСТИ
4	86	ВОЗВРАТ НАЧИСЛЕННЫХ БОНУСОВ БЕЗ КАРТЫ (DLS)
5	4	ЗАВЕРШЕНИЕ АВТОРИЗАЦИИ БЕЗ КАРТЫ (DLS)
9	2	ЧТЕНИЕ КАРТЫ MS (SV)
9	3	ЧТЕНИЕ КАРТЫ MS (ANY TRACK)
9	6	ПРОВЕРКА ПОДКЛЮЧЕНИЯ ПИН-ПАДА / PING
9	7	ЧТЕНИЕ КАРТЫ (CHIP MS)
9	8	ЧТЕНИЕ КАРТЫ (HOLD CHIP)  (OW_ 02) При выполнении команды считывается номер карты и код лояльности, после чего код лояльности передается на кассу. Если была считана токенизированная карта, на которой нет записанного кода лояльности, он определяется по бинам и терминал запрашивает код лояльности с хоста (Должна быть выполнена соответствующая настройка на TMS).  (OW_ cless) Если карта вставлена чипом, то терминал не просит её забрать по окончании операции. Если включен режим HRS и обрабатывается чиповая карта, то на кассу не передаются зашифрованные данные (карты).
9	33	ВВОД ДАННЫХ С КЛАВИАТУРЫ ПИН-ПАДА
9	34	ПИН-ПАД КАК УСТРОЙСТВО ВВОДА ПАРОЛЯ (SV_05)

Класс	ID	Название
9	35	ЧТЕНИЕ КАРТЫ ЛОЯЛЬНОСТИ В РЕЖИМЕ РАБОТЫ С КАССОЙ (РИВ-ГОШ) (SV_05)
9	37	Включение режима работы с принтером терминала (SV_05)
9	38	Запуск режима прокси сервера (SV_05)
9	40	MIFARE – чтение UID и передача его на кассу (карты Mifare Plus X 2k)
9	41	ПЕРЕДАЧА СТАТУСА УСТРОЙСТВ ADM НА ХОСТ (NW_TPTP_15)
9	42	ПЕРЕДАЧА ЖУРНАЛА СОБЫТИЙ ADM НА ХОСТ (NW_TPTP_15)
9	43	ЗАПРОС КОДА ЛОЯЛЬНОСТИ (NW_TPTP_15)
9	45	ЧТЕНИЕ КАРТЫ (CHIP MS) - активация команды через TMS (SV_05 с апреля 2021г.)
9	98	Запрос серийного номера терминала для АДМ (SV_05)



ПО Universal EMV POS поддерживает функционал Динамического меню – в Режиме Работы с кассой для пунктов меню используется класс команд «8», в качестве Op id необходимо использовать значение, указанное в TMS – Терминальные группы – Динамическое меню – Касса (может принимать значения от 1 до 20).

1_6_2 Чтение карты для ISO

В терминальном ПО определены несколько команд для чтения карт:

- 2,4 – чтение магнитной карты (track1 и 2)
- 9,2 – аналогично 2,4. Добавлено для совместимости со старой SmartVista
- 9,3 – аналогично 2,4, но снижены требования к формату треков
- 9,7 – чтение магнитной или чиповой карты

Терминал возвращает значение номера карты клиента в маскированном виде. В большинстве случаев возвращаются первых 6 и/или последние 4 цифры, однако возможны особенности в зависимости от банка.

Если при интеграции необходимо работать с полным номером карты (или SHA1) необходимо использовать специализированную сборку ПО, которая не маскирует данные – предоставляется по запросу. В этом случае кассовой компании необходимо иметь сертификат Payment Application Data Security Standards (PA DSS).

В случае необходимости работы с картой рекомендовано использовать HRS версию платежного софта и ArcCom.dll, которые шифруют данные карты по 3DES алгоритму (поддерживается только в Windows).

1_6_3 Файл ops.ini для Compass+ и SPDH

Класс	ID	Название
ОСНОВНЫЕ ФИНАНСОВЫЕ ОПЕРАЦИИ		
0	128	ОПЛАТА ПОКУПКИ (SPDH)
0	130	ВОЗВРАТ ТОВАРА (SPDH)
0	133	ПРЕДАВТОРИЗАЦИЯ
0	134	ЗАВЕРШЕНИЕ ПРЕДАВТОРИЗАЦИИ
0	224	ОТМЕНА ПОСЛЕДНЕЙ (ТЕХНИЧЕСКАЯ ОТМЕНА, АННУЛИРОВАНИЕ ЧЕКА)
0	225	ОТМЕНА ПО ЧЕКУ (УНИВЕРСАЛЬНАЯ)
СВЕРКА, ОТЧЕТЫ		
0	160	ЗАКРЫТИЕ ПАКЕТА
0	161	ЗАКРЫТИЕ СМЕНЫ
0	162	ЗАКРЫТИЕ ДНЯ
0	179	ЛОКАЛЬНЫЕ ИТОГИ

Класс	ID	Название
0	180	КРАТКИЙ ОТЧЕТ
0	181	ПЕЧАТЬ ЖУРНАЛА
0	186	ПЕЧАТЬ ЖУРНАЛА ТРАНЗАКЦИЙ
0	187	ПОЛНЫЙ ЖУРНАЛ
БАНК - ПУНКТ ВЫДАЧИ НАЛИЧНЫХ		
0	129	БЕЗНАЛИЧНЫЙ ПЕРЕВОД
0	131	ПРИЕМ НАЛИЧНЫХ
0	132	ВЫДАЧА НАЛИЧНЫХ
0	136	БАЛАНС
0	154	ПРОВЕРКА КАРТЫ
0	231	СМЕНА ПИН-КОДА
КОРРЕКЦИИ		
0	144	КОРРЕКЦИЯ ПОКУПКИ
0	147	КОРРЕКЦИЯ ПОПОЛНЕНИЯ НАЛИЧНЫМИ
0	148	КОРРЕКЦИЯ ВЫДАЧИ
ПЛАТЕЖИ ПЕРЕВОДЫ		
0	138	ПЛАТЕЖ В СТОРОНУ ТРЕТЬИХ ЛИЦ (ТРТР_04)
0	142	ПЕРЕВОД С КАРТЫ НА КАРТУ
0	143	ПОПОЛНЕНИЕ КАРТЫ НАЛИЧНЫМИ
СЕРВИСНЫЕ		
0	176	КОПИЯ ПОСЛЕДНЕГО ЧЕКА
0	177	КОПИЯ ЧЕКА ПО НОМЕРУ
0	182	ДЕТАЛИ EMV
СПЕЦИАЛЬНЫЕ ВЕРСИИ ПО		
0	220	ОТМЕНА ПО СУММЕ, сумма - обязательный параметр (SPDH_03)  Данный вид отмены доступен только в режиме работы с кассой, при включенном параметре на TMS "требовать карту для отмены"
12	1	ПРОВЕРКА ПОДКЛЮЧЕНИЯ ПИН-ПАДА (PING) С ЗАПРОСОМ СЕРИЙНОГО НОМЕРА
АДМИНИСТРАТИВНЫЕ ОПЕРАЦИИ И МЕНЮ КАССИРОВ		
0	166	ТЕСТ СВЯЗИ С ХОСТОМ
0	167	ЗАГРУЗКА ПАРАМЕТРОВ С ХОСТА (SPDH)
0	50	МЕНЮ КАССИРА
0	53	АДМИНИСТРАТИВНОЕ МЕНЮ
0	204	СЕССИЯ TMS

1_6_4 Файл rc_conv.ini

В файле rc_conv.ini (фрагмент) заполняется таблица преобразования кода ответа устройства ПИН-пад в код ответа необходимый для ККМ.

К примеру, если необходимо чтобы код ответ **001** записывался как **005**, а код ответа **123** как **843**, то для этого в файл **rc_conv.ini** необходимо добавить строки в следующем формате:

[код ответа ПИН-пад]=[код ответа ККМ].

Например:

001=005
123=843

Добавление этих строк означает что, в случае если ПИН-пад присылает код ответа «001», то код ответа будет заменен на «005», если же код ответа «123», то в файл ответа будет записано «843».

В случае если в файле `rc_conv.ini` данный код ответа отсутствует, то будет использовано значение, полученное с ПИН-пад.



Некорректные данные в файле `rc_conv.ini` могут привести к неверной обработке ККМ кодов ответов терминала и последующим финансовым последствиям. Разработчики и администраторы ПО ККМ должны предпринять меры по защите файла `rc_conv.ini` от несанкционированных изменений

1_6_5 Файл `rc_res.ini`

В файле `Rc_res.ini` (фрагмент) задаются коды ответа процессинга.

В файле заполняется таблица соответствий кодов ответа и информационных сообщений, выводимых на экран. Строки файла имеют следующий формат:

[код ответа]=[Текстовое сообщение]

000= Одобрено
001= Одобрено
002= Одобрено
003= Одобрено
004= Одобрено
005= Одобрено
006= Одобрено
007= Одобрено
008= Одобрено
009= Одобрено
050= Отказ
051= Карта просрочена
052= Лимит PIN исчерпан
053= Чужая адм. карта
054= Нет блока секретности
055= Неверная операция



Подробная информация по содержимому файлов инициализации находится в имеющемся у Вас ТЗ на разработку или внедрению проекта.



При работе с ОС Linux регулируется флагом `USEORIGINALRC` в файле `cashreg.ini` (см. 1.5 Файл `Cashreg.ini`).

1_7 Вывод сообщений и меню на терминал (Dialog Direct)

Для интеграций с ПО ККМ на Windows с использованием `ArcCom.dll` есть возможность выводить произвольные сообщения и меню на терминале. Для активации функционала укажите в файле `cashreg.ini` параметр `USE_DLG_DIRECT=1`.

Возвращаемые значения: 0 : OK
-1: NOK

-3: ERROR_NO_ANSWER
-9: ERROR_NO_SESSION

Начало сессии DialogDirect

int dlgSessionStart(void);

Завершение сессии DialogDirect

int dlgSessionEnd(void);

Отображение строки

int dlgWriteLine(int dest, int rowNum, int cellNum, char *line);

- dest – выполнить команду на терминале(1), либо на ПИН-паде(0)
- rowNum – номер строки
- cellNum – номер столбца
- line – указатель на строку

Чтение строки

int dlgReadLine(int dest, int rowNum, int cellNum, int isMask, int timeout, int length, char *buffer);

- dest – выполнить команду на терминале(1), либо на пин-паде(0)
- rowNum – номер строки
- cellNum – номер столбца
- line – указатель на строку
- isMask – включить/выключить маскирование строки ввода
- timeout – таймаут
- length - максимальная длина строки ввода
- buffer – указатель на память, куда будет записан результат ввода

Чтение кнопки

int dlgReadKey(int dest, int timeout, char *keyCode);

- dest – выполнить команду на терминале(1), либо на пин-паде(0)
- timeout – таймаут
- buffer – указатель на память, куда будет записан результат нажатия кнопки

Очистка экрана

int dlgClrScreen(int dest);

Отображение меню

int dlgDisplayMenu(int dest, const char *caption, const char *const items[], size_t items_size, int *selection);

- dest – выполнить команду на терминале(1)
- caption – указатель на строку с заголовком меню
- items – массив указателей на указатели с названиями пункта меню
- items_size - количество указателей в аргументе items
- selection – указатель на память куда будет записан номер пункта выбранного меню

1_8 Реализация поддержки внешних диалогов

В ARCUS 2.1 доступен функционал отображения диалогов на кассе.

Диалоги настраиваются в структуре, описанной в файле arcus.h (поставляется вместе с библиотекой):

```
typedef struct
{
    size_t size;
    void * userdata;
    cbMenu cbfMenu;
    cbYesNoDialog cbfYesNoDialog;
    cbwarningDialog cbfwarningDialog;
    cbInfo cbfInfo;
```

```

        cbStatus      cbfStatus;
        cbStringEdit  cbfStringEdit;
    }
DialogsCBs;
typedef struct
{
    size_t      size;
    cbI344Window cbfI344OpenWindow;
    cbI344Window cbfI344ClearWindow;
    cbI344DisplayLine cbfI344DisplayLine;
    cbI344ClearLine  cbfI344ClearLine;
    cbI344KeyReq     cbfI344KeyReq;
} DialogsI344CBs;

```



userdata не используется, всегда NULLPTR.

Установка указателей на функции (пример):

```

DialogsCBs callbacks;

...
    if (use_console_dialogs != 0)
    {
        callbacks.userdata = nullptr;
        callbacks.size = sizeof(callbacks);
        callbacks.cbfMenu = callback_menu;
        callbacks.cbfYesNoDialog = callback_yesno;
        callbacks.cbfWarningDialog = callback_warning;
        callbacks.cbfInfo = callback_info;
        callbacks.cbfStatus = callback_status;
        callbacks.cbfStringEdit = callback_stringedit;
ITPosSet(itpos, "DialogsCBs", std::to_string((long) &callbacks).c_str(), -1);
    }
...

```

Проверка работы для Windows версии:

Для проверки работы в ArcCom.dll необходимо использовать ключ /console - Use callback dialog functions, например:

```
CommandLineTool /o1 /a101 /console
```

Проверка работы для Linux версии:

```
./CommandLineTool /o1 /a101 /c643 /console
```

1_9 Возврат (отмена) без предъявления карты

Функционал реализует возможность проведения операций «Отмена», «Возврат» и «Завершения предавторизации» по коду авторизации без карты клиента в режиме работы с кассой.

Так как в журнале операций хранится только номер карты и срок действия, операции «Возврат» и «Завершения предавторизации» по коду авторизации выполняются только в режиме ручного ввода (без данных треков и данных EMV).

Для операций «Отмена», «Возврат» и «Завершения предавторизации» в режиме работы с кассой терминал обрабатывает полученные с кассы код авторизации, сумму и 4 последних цифры номера карты⁴.

Для операции «Отмена» поиск выполняется по трем параметрам (код авторизации, сумма, 4 последних цифры номера карты). Поиск производится по текущему бизнес-дню.

Для операций «Возврат» и «Завершения предавторизации» поиск выполняется только по коду авторизации и 4 последним цифрам номера карты, а полученная с кассы сумма используется при проведении новой операции. Поиск производится по всему журналу транзакций в памяти терминала, включая операции по бизнес-дням, по которым уже было выполнено закрытие дня.

⁴ Терминал должен быть настроен для передачи на кассу как минимум 4 последних цифр номера карты в открытом виде

Если операция в журнале найдена, то терминал выполняет операцию без необходимости ввода карты клиента.

Если операция не найдена, то терминал продолжает работу по стандартному сценарию: запросить карту клиента (для операций «Возврат» и «Завершение расчета») или номер чека (для операции «Отмена»).

Если один или несколько параметров (код авторизации, сумма, 4 последних цифры номера карты) не были получены терминалом, то поиск транзакции выполняется без них (например, по коду авторизации и номеру карты или по коду авторизации и сумме).

Для операций «Возврат» и «Завершение предавторизации» терминал использует сумму, полученную с кассы. Операция «Отмена» производится только на полную сумму.

Для реализации функционала разработана специальная версия DLL.

1_10 Передача информации о товарах (Arcus 2.0)

Для использования и передачи списка товаров в ARCUS.dll (Windows) или arcus.so (Linux) добавляется следующий вызов (см. arcus.h - поставляется вместе с библиотекой):

```
#ifndef _MSC_VER
    #pragma pack(push, 1)
#endif
/**
 * @struct GoodsItem_t
 *
 * This is a structure for good item
 */
typedef struct_packed {
    uint32_t version;                // версия структуры, должна быть 1
    char product_name[16];           // название продукта, кодировка указана в параметре
                                     // ORCHARSET, файла cashreg.ini
    size_t product_name_len;         // итоговая длина буфера product_name в байтах
    char commodity_code[14];         // код товара, формат:ascii alphanumeric
    size_t commodity_code_len;       // итоговая длина буфера commodity_code в байтах
    uint8_t dispenser_number;        // номер диспенсера, от 1 до 99
    char measure;                    // единица измерения, принимает значения
    'L','U','O','W',' '              //
    uint8_t exponenta;               // количество знаков после запятой параметра
                                     // «количество», например: 2 - 0.00, 3 - 0.000
    uint32_t quantity;               // количество, в минимальных единицах измерения,
                                     // например (десять с половиной литров,
                                     // экспонента = 3): 10500
    uint64_t price;                  // цена единицы товара в минимальных единицах валюты
    uint64_t amount;                 // общая сумма по позиции в минимальных единицах
    валюты
} GoodsItem_t;

* @struct BasketOfGoods_t
*
* This is a structure for goods of basket
*/
typedef struct_packed {
    uint32_t version;                // версия структуры, должна быть 1
    uint16_t curcode;                // код валюты(число) в iso4217,
                                     // например 643-Российский рубль
    uint64_t total_amount;           // общая стоимость корзины,
                                     // например (10 руб. 50 коп.): 1050
    char sale_number[40];            // номер чека продажи, формат:ascii alphanumeric
    size_t sale_number_len;          // общая длина буфера sale_number в байтах
    uint16_t items_number;           // количество позиций в корзине
    GoodsItem_t *items;              // указатель на массив GoodsItem_t
} BasketOfGoods_t;
#ifdef _MSC_VER
#pragma pack(pop)
#endif
/**
 * \brief describes Callback-function for getting basket of goods.
 * \return - NULL or pointer to the basket of goods
 */
typedef BasketOfGoods_t *(__cdecl *cbGetBasketOfGoods)(void);
/**
 * \brief Sets callback-function for basket of goods.
 *
 * \param [in] func - pointer to callback-function.
 */
```

```
DLL_EXPORT void __cdecl SetBasketOfGoodsCB(cbGetBasketOfGoods func);
```

Функция из библиотеки SetBasketOfGoodsCB позволяет передать указатель на функцию, по которой будет получена товарная информация.

Заполнить товарную корзину, а также передать указатель на функцию необходимо до вызова функции ProcessOwFull (см. пример в приложении 4).

1_11 Печать данных на принтере терминала (Arcus 2.0)

В ARCUS2 реализован функционал вывода данных для печати на принтере терминала. Для этого используется функция API ArcCom.dll:

```
int PrintBuffer(char *buffer, unsigned int length)
```

, где

- buffer- текстовый буфер в кодировке cp1251.
- length- длина буфера.



**Форматирование при печати не предусмотрено, выравнивание текста по левому краю.
Допускается использовать символ перевода строки.**

Возвращаемое значение: 0 - успешно, иначе ошибка.

1_12 Специальные версии ARCUS2 CAP (Arcus 2.0)

Существуют реализации ARCUS 2 CAP для кассового ПО различных производителей, поддерживающие индивидуальную логику работы.

Реализация может осуществляться путем внесения изменений в библиотеку ARCUS2, добавлением дополнительных файлов настройки и пр. В некоторых случаях необходимо использовать модифицированные версии платежного приложения Universal EMV POS с поддержкой дополнительных функций.

1_12_1 ПО ККМ DKLINK с расчетом скидки на терминале

Данная модификация предназначена для продукта с кодом SPDH_03.

При оплате банковской картой с BIN, принадлежащим определенному диапазону, начисляется скидка. Если BIN карты не попадает в диапазон, оплата проводится обычным способом.

Дополнительные настройки оплаты ARCUS2 производятся в файле arcus.cfg.

- Скидки при оплате Аркус-2: 0 – выключены (простая оплата – по умолчанию), 1 – включены (оплата со скидкой банка).
- Код валюты: по умолчанию – 643 (рубли).
- Признак магазина, в котором установлено оборудование: 0 – супермаркет (по умолчанию, 1 – гипермаркет).

В чеке может быть только одна оплата ARCUS2 со скидками банка независимо от состояния параметра «Разрешить смешанную оплату», т.е. если в чеке есть наличная оплата, то нельзя добавить оплату Аркус-2 и наоборот, если в чеке есть оплата Аркус-2, то нельзя добавить другой тип оплаты. Чек должен быть полностью оплачен типом оплаты ARCUS2.

При оплате со скидкой банка помимо обычных параметров кассовый модуль (KM) дополнительно передает два параметра:

- /dmin[сумма скидки в копейках] – минимальная сумма скидки по всему чеку.
- /dmax[сумма скидки в копейках] – при работе в гипермаркете максимальная сумма скидки;
/dmax[сумма товаров на которые распространяется скидка] – при работе в супермаркете сумма товаров, на которые распространяется скидка.
- /a - сумма чека в копейках без скидок.

/ Логика расчета скидки

Расчет суммы авторизации:

Для гипермаркета

Сумма, переданная в параметре /a, умножается на размер скидки по банковской карте, после чего определяется допустимость предоставления получившейся скидки исходя из параметров /dmin и /dmax. Если рассчитанный дисконт получился менее /dmin, то предоставляется скидка в размере /dmin, если рассчитанный дисконт получился больше /dmax, то предоставляется дисконт в размере /dmax, во всех других случаях предоставляется дисконт, определенный терминалом.

Для супермаркета

Из суммы, переданной в параметре /a вычитается /dmax, умноженная на размер скидки по карте Банка. Если рассчитанный дисконт получился менее /dmin, то предоставляется скидка в размере /dmin, если рассчитанный дисконт получился больше /dmax, то предоставляется дисконт в размере /dmax, во всех других случаях предоставляется дисконт, определенный терминалом.

Пример 1:

В чеке обычный товар на который возможна скидка в диапазоне от 0 до 8%, клиент владеет дисконтной картой 5%:

```
/a13100 /dmin655 /dmax1048
```

Пример 2:

В чеке с единственным товаром стоимостью 131.00 руб., на который установлено ограничение по скидке 8% (т.е. максимально допустимая скидка на данный товар – 8%), магазин предоставляет скидку 5%:

```
/a 13100 /dmin655 /dmax1048  
/dmin655 = (131.00 * 5%) * 100  
/dmax1048= (131.00 * 8%) * 100
```

Пример 3:

В чеке только товар по акции, т.е. ограничение по скидке – 0%, не допускается скидка на товар:

```
/a13100 /dmin0 /dmax0
```

- Кассовое ПО, после получения суммы авторизации, выполняет обратный расчет и регистрирует процент выполненной скидки в журналах работы кассы.

1_12_2 ПО ККМ ДАТАКРАТ с расчетом скидки на кассе

Данная модификация предназначена для продукта с кодом OW_cless.

/ Логика расчета скидки

1. При вызове Безналичной оплаты на ККМ, кассовое ПО передает на модуль Universal EMV POS/ARCUS2 набор данных с указанием типа операции, кода валюты и суммы.
2. Кассир считывает карту на терминале, который в ходе обработки данных, если необходимо, запрашивает дополнительные действия, например, ввод ПИН.
3. Терминал формирует запрос итоговой суммы операции, в котором передает на ККМ тип карты и первые 6 цифр PAN карты без спецсимволов.
4. ККМ, получив запрос, взаимодействует с дисконтным модулем кассовой системы с целью расчета итоговой суммы операции. Полученную сумму кассир подтверждает на кассе.
5. Если сумма подтверждена, то она передается терминалу, в противном случае операция сбрасывается и на терминал посылается отмена проведения транзакции.
6. После получения новой суммы, терминал формирует транзакционный пакет и отправляет запрос на банковский хост.
7. Полученный от банка ответ, обрабатывается терминалом и на ККМ передаются результаты авторизации, а также образ чека.

В случае если ККМ не получила ответ от терминала в течение заданного промежутка времени, а также если нет возможности распечатать чек о совершенной операции, кассовое ПО должно инициировать выполнение операции «Технический реверсал». Операция «Технический реверсал» предназначена для отмены последней операции, отправленной на выполнение на терминал.

/ Изменения в интерфейсе ARCUS2:

Для передачи кассе типа карты, первых шести цифр номера карты и новой суммы терминалу, в интерфейс библиотеки «arcscm.dll» добавляется следующий вызов:

```
/*!  
* \brief describes callback-function for calculation of net amount.  
* \param [in] bin - pointer to buffer which contains first 6 digits of PAN (ASCIIIZ)  
* \param [in] ct_name - pointer to buffer which contains the name of card type (ASCIIIZ)  
* \param [out] newamount- pointer to buffer for storing new amount (must be formatted as  
ASCIIIZ)  
* \param [in] newamount_size - size of buffer for storing new amount  
* \return - 1 if transaction must be continued  
* \return - 0 if transaction must be terminated  
* */  
typedef int (__cdecl *cbUpdateAmount2)(const char *pan, const char *ct_name, char  
*newamount, size_t newamount_size);  
  
/*!  
* \brief sets callback-function for calculation of new amount.  
*  
* \param [in] func - pointer to callback-function.  
* */  
extern "C" __declspec(dllexport) void __cdecl SetAmountUpdateCB2(cbUpdateAmount2 func);
```

Функция позволяет передать указатель на функцию, которая непосредственно взаимодействует с дисконтным модулем кассы, и рассчитывает новую сумму операции.

Для контроля вызова функции SetAmountUpdateCB добавлены следующие возвращаемые значения:

```
/*!  
* \brief sets callback-function for calculation of new amount.  
*  
* \param [in] func - pointer to callback-function.  
* \return 0 - callback was setted  
* \return 1 - callback was NOT setted  
* */  
extern "C" __declspec(dllexport) int __cdecl  
SetAmountUpdateCB(cbUpdateAmount func);
```

В остальном интерфейс библиотеки не меняется.

Вызов функции «SetAmountUpdateCB2» необходимо выполнять каждый раз перед вызовом функции «ProcessOWFull». В процессе выполнения транзакции, после получения первых шести цифр номера карты, модуль Universal EMV POS/ARCUS2 вызовет функцию, указатель на которую был передан в качестве параметра функции «SetAmountUpdateCB2». При этом 6 цифр номера карты будут переданы в параметре «pan». Функция должна вернуть код ответа 1, и новую сумму в параметре «newamount» для продолжения операции, или код ответа 0 для прерывания операции.



Функционал расчета скидки и итоговой суммы кассовым приложениям не применим к операциям по бесконтактным картам.

1_12_3 ПО ККМ Set Retail 10

Для работы с ПО ККМ Set Retail 10 реализован функционал передачи данных платежа на кассу при помощи файла и предварительно заданного шаблона передаваемых параметров. Для активации функционала в файле настроек cashreg.ini необходимо задать два параметра:

- OUTPUT_FILE=[Имя_Файла] - название исходящего файла с параметрами платежа (значение по умолчанию - chek.out);
- OUTPUT_FILE_TEMPLATE=[Имя_Файла] - название файла шаблона с заданными параметрами и их последовательностью (значение по умолчанию - output.tpl).



Если параметр OUTPUT_FILE_TEMPLATE не задан, или файл шаблона отсутствует, данные платежа передаются на кассу при помощи файла, указанного в настройке OUTPUT_FILE, без применения шаблона - в таком порядке, как задано в примере файла шаблона «по умолчанию».

Файл шаблона output.tmp1 «по умолчанию»

RESULTCODE
PAN
TERMINALID
AUTHCODE
APPLABLE
MESSAGE
AMOUNT
RECEIPT
RRN
TRANSACTIONID



Если значение Application Label нулевое вместо него выводится Тип карты.

1_13 Эксплуатация

1_13_1 Интеграция ARCUS2.1 CAP с ПО ККМ

ARCUS 2.1 CAP в результате доработок кассового ПО становится дополнительным модулем ПО ККМ, расширяющим возможности функционирования ПО ККМ в плане выполнения расчетов по банковским картам при проведении операций оплаты товаров и услуг в торговых и сервисных предприятиях и осуществления операций со счетами клиентов в отделениях Банков.

1_13_2 Виртуальная клавиатура (Arcus 2.0)

Виртуальная клавиатура доступна для касс, работающих на ОС Windows, с сенсорным экраном для ПО Unipay. Клавиатура не отключаемая и для её работы необходима специальная версия dialogs.dll.



Рис. 1. Виртуальная клавиатура

При помощи виртуальной клавиатуры возможно осуществлять ввод следующих данных:

- цифровой ввод – редактирование;
- ввод/редактирование IP- адресов;
- ввод/редактирование HEX.

1_13_3 Способы и периодичность контроля правильности работы ПО

Контроль за правильностью работы ARCUS 2.1 CAP осуществляется в рамках комплексного технического обслуживания эквайринга Банка в ТСП, техническими специалистами служб эквайринга банковских карт со стороны процессинга и фирмы производителя ARCUS 2.1 CAP на основании статистической информации по сбоям, которая поступает из процессинга Банка и от IT-специалистов ТСП.

В случае получения уведомления от производителя кассового ПО о выходе обновлений существующей версии кассового ПО или выпуске принципиально новой версии кассового ПО, осуществляются мероприятия по контролю совместимости настроек модуля ARCUS 2.1 CAP и выполнению тестовых испытаний на стенде кассовой компании или в ТСП.

В случае выхода обновлений существующей версии ARCUS 2.1 CAP или выпуске принципиально новой версии ARCUS 2.1 CAP, осуществляются мероприятия по контролю совместимости интерфейса и настроек модуля ARCUS 2.1 CAP и выполнению тестовых испытаний на стенде кассовой компании или в ТСП.

1_13_4 Аварийные ситуации и способы их устранения

При возникновении аварийных ситуаций (затирая или удаления критических файлов из состава ARCUS 2.1 CAP, краха ПО или ОС) техническими специалистами ТСП и службы тех. обслуживания эквайринга, осуществляются мероприятия по восстановлению и настройке модуля ARCUS 2.1 CAP и выполнению пусковых испытаний в ТСП.

При возникновении аварийных ситуаций (сбоев при выполнении финансовых и административных операций по банковским картам) техническими специалистами ТСП и службы тех. обслуживания эквайринга, осуществляются мероприятия по анализу причин сбоев на основании данных из файлов логирования ARCUS 2.1 CAP, составлению отчетов по выявленным причинам и передача информации в службы поддержки программных продуктов и департамент разработки ПО ARCUS 2.1 CAP для проведения работ по устранению ошибок ПО ARCUS 2.1 CAP и выпуску обновления, после чего происходит выполнение отладочных, тестовых и пусковых испытаний в службах тестирования производителя ARCUS 2.1 CAP и в ТСП.

1_13_5 Описание фрагментов файла лога

- Версия сборки Arccom.dll

Version: 2.1.0.1
Build date:
Tue Dec 23 17:56:01 2014

- USE_JL ON\OFF - (Arcus 2.0) соответствует включенному или выключенному параметру USEJL в cashreg.ini. Параметр сигнализирует об использовании нескольких юридических лиц на терминале. Если параметр включен – в лог передается значение, указанное в usejl.txt (порядковый номер виртуального терминала или TerminalID):

USEJL ON, value=00010145

2_Интеграция через RS232 или USB

В зависимости от модели используемого терминала его подключение к ККМ может осуществляться либо через последовательный порт RS232, либо к USB порту.

В случае, если сетевое подключение к банковскому процессингу предполагается осуществлять через оборудование ККМ, то на ней должен быть установлен сетевой интерфейс, через который обеспечивается доступ в сеть. Подключение к процессингу возможно также без участия ККМ, а непосредственно от терминала, в таком случае необходимо наличие канала связи, к которому будет подключен терминал.

На ККМ должна быть установлена операционная система Windows 98/2000/XP, кассовое ПО, а также пакет ПО ARCUS 2.1.

При подключении терминала через порт USB на ККМ должен быть установлен драйвер виртуального COM-порта.

В подключаемый терминал должно быть загружено программное обеспечение, версия которого зависит от типа банковского процессинга, а также конфигурационные файлы.

2_1 Схема взаимодействия с ПО ККМ

При проведении безналичного платежа, кассовое ПО должно сформировать данные по платежу:

- Код операции;
- Код валюты;
- Сумма операции в сотых долях.

Данные заполняются одним из следующих способов:

- через метод «Set», затем ПО кассы вызывает библиотечную функцию «Run», в качестве параметра передав ей номер операции (вызов функции «Run» является блокирующим, управление диалогом с кассиром передается приложению ПИН-пад);
- через структуру UserAuthIntFull, затем ПО кассы должно вызвать библиотечную функцию ProcessOWFull, в качестве параметра передав ей указатель на заполненную структуру UserAuthIntFull. Вызов функции ProcessOWFull является блокирующим, управление диалогом с кассиром передается приложению ПИН-пад.

В ходе выполнения операции могут быть затребованы ввод кода CVC2/CVV2, ввод ПИН-кода. Вся текстовая информация передается в кодировке WIN1251.

3_Низкоуровневая интеграция

ПО ККМ может взаимодействовать с ПИН-пад без использования библиотеки ARCUS 2.1. В этом случае в ПО ККМ должна быть реализована полная поддержка протокола ARCUS 2.1.

ПИН-пад должен быть настроен на работу с ККМ в соответствии с документом «ARCUS 2.1 для интеграции с кассовыми решениями - Руководство по установке, п. 3.2. Переключение терминала в режим работы с кассой». Устройство (ПИН-пад), после включения питания и запуска ПО находится в состоянии ожидания команды начала проведения операции по настроенному каналу связи (RS232, USB, TCP ...).

После получения команды ПИН-пад переходит в активный режим и начинает обработку запрошенной операции. В процессе выполнения устройство отправляет на ККМ команды открытия тех или иных ресурсов, чтения и записи, печати и отображения различных диалогов. Присланная команда должна быть обработана на ККМ, после чего должен быть выработан ответ на команду и отправлен на ПИН-пад.

В конце каждой операции ПИН-пад отправляет команду завершения операции, после ответа на которую, операция считается завершенной и устройство ККМ должно закрыть все открытые в процессе ресурсы. Операция считается завершенной.

Результат выполнения определяется по значению кода ответа присланного специальной командой с ПИН-пад-а. Отсутствие в процессе выполнения операции подобной команды считается отрицательным результатом выполнения (операция завершилась с неизвестной ошибкой). Так же в процессе проведения может быть прислано несколько таких команд, в этом случае в качестве значения кода ответа используется последнее присланное ПИН-пад-ом.

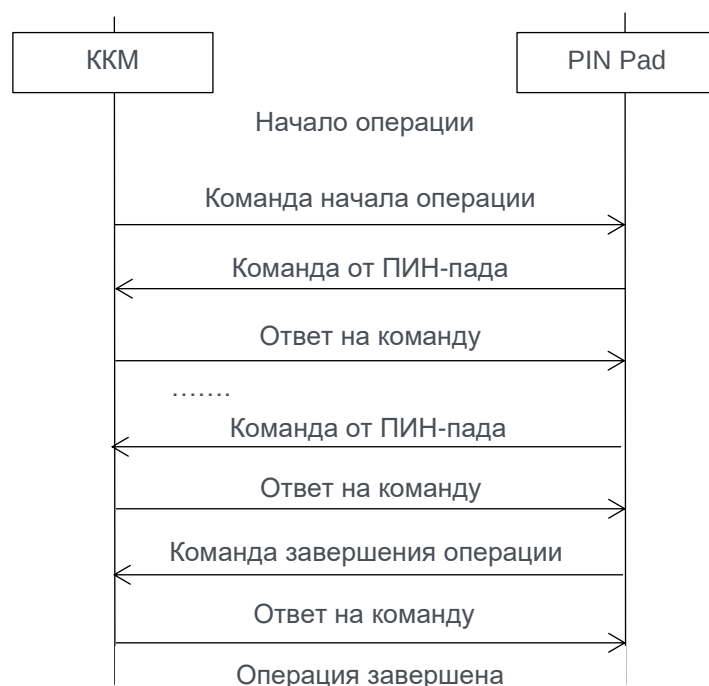


Рис. 2. Общая схема взаимодействия

Описание низкоуровневого протокола взаимодействия предоставляется по запросу в службу технической поддержки Ingenico.

ПРИЛОЖЕНИЕ 1

КЛЮЧИ ДЛЯ МЕТОДОВ SET/GET

Взаимодействие происходит через методы Set/Get(IN/OUT), в которые передается следующий набор ключей:



Ключи регистрозависимые.

Ключ	Назначение	Направление	Максимальная длина данных (рекомендованная ISO8583)
currency	Код валюты ISO	IN	3
amount	Сумма операции без разделителей (в минимальных единицах валюты)	IN	12
result_amount	Сумма после операции (в минимальных единицах валюты)	OUT	12
original_amount	Оригинальная сумма операции без разделителей (в минимальных единицах валюты)	IN	12
rfu	Трек2 карты (Track2 ISO)	IN	60
pan	Маскированный номер карты(PAN)	OUT	20
expiry	Expiry Date (YYMM)	OUT	4
aid	EMV AID	OUT	80
card_type	Тип карты	OUT	80
application_label	EMV ApplicationLabel	OUT	80
date	Дата (процессинг) (DDMM)	OUT	4
time	Время (процессинг) (hhmmss)	OUT	6
terminal_id	Идентификатор терминала TID (процессинг)	IN, OUT	8
terminal_serial_number	Серийный номер терминала	OUT	24
ARCUS_ID	Идентификатор Аркус2 и АРМ кассира	IN	12
response_code	Код ответа- результат операции	OUT	3
auth_code	Код авторизации	IN, OUT	8
rrn	RRN (номер ссылки)	IN, OUT	12
tvr	EMV TVR	OUT	80
received_text_message	Текстовое	OUT	80
text_message	Расшифровка кода ответа	OUT	80
cardholder_name	Имя держателя карты	OUT	27
enc_data / result_enc_data	Шифрованные данные карты для поддержки функционала HRS	IN / OUT	64
trace_id	ID-преавторизации	IN, OUT	60
original_date_time	Дата-время оригинальной операции (YYMMDDhhmmss)	IN, OUT	12
receipt_number	Номер чека	IN, OUT	4
transaction_id	Transaction ID	IN / OUT	18
loyalty_code	код лояльности (TPTP_15)	IN / OUT	4

ПРИЛОЖЕНИЕ 2

ПРИМЕР ИСПОЛЬЗОВАНИЯ

БИБЛИОТЕКИ

Прототипы используемых функций

```
extern "C" ARCCOM_API void* __cdecl CreateITPos();
extern "C" ARCCOM_API void __cdecl DeleteITPos(void *obj);
extern "C" ARCCOM_API int __cdecl ITPosSet(void *obj, const char *key, const char *value, int size = -1);
extern "C" ARCCOM_API int __cdecl ITPosGet(void *obj, const char *key, char *value, int size = -1);
extern "C" ARCCOM_API int __cdecl ITPosRun(void *obj, int cmd);
extern "C" ARCCOM_API int __cdecl ITPosRunCmd(void *obj, const char *cmd_str, const char *value, int size = -1);
extern "C" ARCCOM_API void __cdecl ITPosClear(void *obj);
extern "C" ARCCOM_API void __cdecl Stop(void *obj);
```

- obj – объект взаимодействия с POS терминалом;
- key – название параметра, передаётся в виде ASCII строки (с терминальным символом);
- value – содержимое параметра (IN/OUT), передаётся в виде массива символов;
- size – длина данных, передаваемых в параметре, если -1, то длина вычисляется как для ASCII строки (только для нуль терминированных строк);
- cmd – код операции, определенной в ops.ini;
- cmd_str. – вызов специализированных операций.

Описание функций:

Функция	Описание
CreateITPos	Создание объектов взаимодействия с POS терминалом
DeleteITPos	Удаление объектов взаимодействия с POS терминалом
ITPosSet	Установка параметров для операции
ITPosGet	Получение параметров после выполнения операции
ITPosRun	Выполнение операции
ITPosRunCmd	Выполнение специализированной операции (например, TEST_PING)
ITPosClear	Очистка параметров перед выполнением новой операции
Stop	Отмена ожидания карты (асинхронный метод)

Пример использования на языке C++



Под ОС Linux необходимо добавить дефайн `__LINUX__`

```
//-----
#include "stdafx.h"
#include <string>
#include <conio.h>

#ifdef __LINUX__
    #include <stdexcept>
    #include <dlfcn.h>
#else
    #include <windows.h>
#endif
//-----
#ifdef __LINUX__
    #ifndef __cdecl
        #define __cdecl          __attribute__((cdecl))
    #endif
#endif
```

```

#define HINSTANCE void*
#define LoadLibrary(x) dlopen((x), RTLD_LAZY)
#define FreeLibrary(x) dlclose((x))
#define GetProcAddress(x, y) dlsym((x), (y))
#define GetLastError std::to_string(errno).c_str
#endif

typedef void *(__cdecl *ARCUS_CREATE)();
typedef void (__cdecl *ARCUS_DELETE)(void *);
typedef int(__cdecl *ARCUS_SET)(void *, const char *, const char *, int);
typedef int(__cdecl *ARCUS_GET)(void *, const char *, char *, int);
typedef int(__cdecl *ARCUS_RUN)(void *, int);
typedef int(__cdecl *ARCUS_RUN_CMD)(void *, const char *, const char *, int);
typedef int(__cdecl *ARCUS_CLEAR)(void *);

#ifdef __LINUX__
    const char dll_name[] = "./libarccom.so";
#else
    const char dll_name[] = "Arccom.dll";
#endif
//-----
int main(int argc, char *argv[])
{
    void *pos_obj = NULL;
    ARCUS_DELETE ArcusDelete = NULL;

    try
    {
        /// load extern DLL with require functions
        HINSTANCE dll = LoadLibrary(dll_name);

        if (dll == NULL)
        {
            /// exit if DLL file not loaded
            throw std::runtime_error("DLL not Load");
        }

        /// create all DLL functions
        ArcusDelete = (ARCUS_DELETE) GetProcAddress(dll, "DeleteITPos");

        ARCUS_CREATE ArcusCreate = (ARCUS_CREATE) GetProcAddress(dll, "CreateITPos");
        ARCUS_SET ArcusSet = (ARCUS_SET) GetProcAddress(dll, "ITPosSet");
        ARCUS_GET ArcusGet = (ARCUS_GET) GetProcAddress(dll, "ITPosGet");
        ARCUS_RUN ArcusRun = (ARCUS_RUN) GetProcAddress(dll, "ITPosRun");
        ARCUS_RUN_CMD ArcusRunCmd = (ARCUS_RUN_CMD) GetProcAddress(dll, "ITPosRunCmd");
        ARCUS_CLEAR ArcusClear = (ARCUS_CLEAR) GetProcAddress(dll, "ITPosClear");

        /// check DLL functions
        if (ArcusCreate == NULL) throw std::runtime_error("function ArcusCreate not loaded");
        if (ArcusDelete == NULL) throw std::runtime_error("function ArcusDelete not loaded");
        if (ArcusSet == NULL) throw std::runtime_error("function ArcusSet not loaded");
        if (ArcusGet == NULL) throw std::runtime_error("function ArcusGet not loaded");
        if (ArcusRun == NULL) throw std::runtime_error("function ArcusRun not loaded");
        if (ArcusRunCmd == NULL) throw std::runtime_error("function ArcusRunCmd not loaded");
        if (ArcusClear == NULL) throw std::runtime_error("function ArcusClear not loaded");

        /// initialize object for POS
        pos_obj = ArcusCreate();

        if (pos_obj == NULL)
        {
            /// exit if object not created
            throw std::runtime_error("create object fail");
        }

        /// repeat operation
        while (1)
        {
            if ((argc > 1) && (std::string(argv[1]) == "ping"))
            {
                printf("PING\n");
                /// run PING POS operation without create log files
                ArcusRunCmd(pos_obj, "TEST_PING", "1", -1);
            }
            else
            {
                /// set amount and currency for purchase operation
                if (ArcusSet(pos_obj, "amount", "2000", -1) != 0) throw
            }
        }
    }
}

```

```

std::runtime_error("set amount fail");
    if (ArcusSet(pos_obj, "currency", "643", -1) != 0)        throw
std::runtime_error("set currency fail");

- ERROR

    /// start purchase (1), result is response code of this operation: 0 - OK, else

    int result = ArcusRun(pos_obj, 1);
    printf("Command result: %d\n", result);

    if (result != 0)
    {
        /// exit if operation return not 0
        throw std::runtime_error("operation fail");
    }

    /// if send NULL that function return length of this parameter
    int size = ArcusGet(pos_obj, "rrn", NULL, -1);

    if (size <= 0)
    {
        /// exit if object cann't return szie of RRN buffer
        throw std::runtime_error("get RRN fail");
    }

    /// create buffer for parameter more than 1 byte for terminate character
    char *rrn = new char[size + 1];

    /// for example: try to get RRN of operation
    if (ArcusGet(pos_obj, "rrn", rrn, size + 1) < 0)
    {
        /// exit if object cann;t return RRN
        delete[] rrn;
        throw std::runtime_error("get RRN fail");
    }

    printf("RRN: %s\n", rrn);

    /// free buffer
    delete[] rrn;

    /// clear all transaction data before start new operation
    ArcusClear(pos_obj);
}

printf("Press any key for repeat (ESC - exit) ...\n");
int c = getch();

if (c == 0x1B)
{
    /// exit after press ESC
    break;
}
}
catch (std::exception &e)
{
    printf("ERROR: %s\n", e.what());
}
catch (...)
{
    printf("ERROR: unhandled exception\n");
}

if ((ArcusDelete != NULL) && (pos_obj != NULL))
{
    ArcusDelete(pos_obj);
}

return 0;
}
//-----

```

Пример использования на языке C#

```
using System;
using System.Text;
using System.Runtime.InteropServices;
using System.IO;

namespace TestArcus
{
    class ITPos
    {
        [DllImport("Arccom.dll")]
        static public extern IntPtr CreateITPos();

        [DllImport("Arccom.dll")]
        static public extern IntPtr DeleteITPos(IntPtr pClassNameObject);

        [DllImport("Arccom.dll")]
        static public extern int ITPosSet(IntPtr pClassNameObject, byte[] key, byte[] value, int len);

        [DllImport("Arccom.dll")]
        static public extern int ITPosGet(IntPtr pClassNameObject, byte[] key, byte[] value, int len);

        [DllImport("Arccom.dll")]
        static public extern int ITPosRun(IntPtr pClassNameObject, int cmd);

        // in
        public int OperationCode = -1;
        public string Amount = "";
        public string CurrencyCode = "";
        public string ArcusID = "";

        // out
        public string ResponseCodeHost = "";
        public string ReferenceNumber = "";
        public string Slip = "";
        public int Run()
        {
            // create object
            IntPtr itpos = CreateITPos();

            if (itpos == null)
            {
                // if object not created
                return -1;
            }

            byte[] amt = Encoding.ASCII.GetBytes(Amount);
            byte[] cur = Encoding.ASCII.GetBytes(CurrencyCode);
            byte[] id = Encoding.ASCII.GetBytes(ArcusID);

            // set require parameters
            ITPosSet(itpos, Encoding.ASCII.GetBytes("amount"), amt, amt.Length);
            ITPosSet(itpos, Encoding.ASCII.GetBytes("currency"), cur, cur.Length);
            ITPosSet(itpos, Encoding.ASCII.GetBytes("ARCUS_ID"), id, id.Length);

            if (ReferenceNumber.Length > 0)
            {
                byte[] rrn_in = Encoding.ASCII.GetBytes(ReferenceNumber);
                ITPosSet(itpos, Encoding.ASCII.GetBytes("rrn"), rrn_in, rrn_in.Length);
            }

            byte[] empty = { 0 };
            ITPosSet(itpos, Encoding.ASCII.GetBytes("NO_CHECK_LAST_OP"), empty, 0);

            if (!File.Exists(@"debug"))
            {
                // disable all temporary files
                ITPosSet(itpos, Encoding.ASCII.GetBytes("TRACE_FILE"), empty, 0);
                ITPosSet(itpos, Encoding.ASCII.GetBytes("OUTPUT_FILE"), empty, 0);
                ITPosSet(itpos, Encoding.ASCII.GetBytes("RESULT_FILE"), empty, 0);
                ITPosSet(itpos, Encoding.ASCII.GetBytes("CHEQ_FILE"), empty, 0);
            }

            // start operation
            int res = ITPosRun(itpos, OperationCode);
        }
    }
}
```

```

        byte[] rrn = new byte[20];
        byte[] rc = new byte[20];
        byte[] slip = new byte[4096];

        // get result fields
        ITPosGet(itpos, Encoding.ASCII.GetBytes("rrn"), rrn, rrn.Length);
        ITPosGet(itpos, Encoding.ASCII.GetBytes("response_code"), rc, rc.Length);
        ITPosGet(itpos, Encoding.ASCII.GetBytes("slip"), slip, slip.Length);

        ResponseCodeHost = Encoding.ASCII.GetString(rc);
        ReferenceNumber = Encoding.ASCII.GetString(rrn);
        Slip = Encoding.ASCII.GetString(slip);

        // delete object
        DeleteITPos(itpos);

        return res;
    }
}

```

Пример использования на языке Object Pascal (Delphi):

```

unit test;

{ $mode objfpc } { $H + }

interface

    uses
        Classes, SysUtils, FileUtil, Forms, Controls, Graphics, Dialogs, StdCtrls, dynlibs;

type

    { TArcusForm }

    TArcusForm = class(TForm)
        AmountEdit : TEdit;
        AdminButton: TButton;
        MessageEdit: TEdit;
        MessageLabel: TLabel;
        BalanceButton: TButton;
        ResultEdit: TEdit;
        ResultLabel: TLabel;
        PurchaseButton: TButton;
        CurrencyEdit: TEdit;
        AmountLabel: TLabel;
        CurrenyLabel: TLabel;
        procedure AdminButtonClick(Sender: TObject);
        procedure BalanceButtonClick(Sender: TObject);
        procedure FormCreate(Sender: TObject);
        procedure PurchaseButtonClick(Sender: TObject);
    private
        { private declarations }
        procedure StartArcus(cmd : integer);
    public
        { public declarations }
    end;

var
    ArcusForm : TArcusForm;

implementation

{ $R *.lfm }

{ TArcusForm }

```

```

type
  CharArray = array[0..1024] of char;
  // функция создания объекта взаимодействия с терминалом
  // результат: указатель на объект для работы с ПОС терминалом, если "nil" - ошибка при
создании объекта
  TCreateITPos = function() : Pointer; cdecl;
  // функция удаления объекта (после данной функции, использовать указатель нельзя,
необходимо снова вызывать CreateITPos)
  // входные данные
  // Ptr - указатель на объект
  TDeleteITPos = procedure(var Ptr : Pointer); cdecl;
  // установка параметра
  // входные данные
  // Ptr - указатель на объект
  // Key - название параметра, передаётся в виде ASCII строки с завершающим терминальным
символом
  // Value - значение параметра, буфер данных, размером Size, если Size=-1, то длина
вычисляется по терминальному символу (для ASCII строк)
  // Size - размер передаваемых данных, -1 - размер вычисляется автоматически
  // результат: если 0 - параметр успешно записан, отличное от 0 - ошибка
  TITPosSet = function(Ptr : Pointer; Key: PChar; Value: PChar; Size: Integer) : Integer;
cdecl;
  // получение параметра
  // входные данные
  // Ptr - указатель на объект
  // Key - название параметра, передаётся в виде ASCII строки с завершающим терминальным
символом
  // выходные данные
  // Value - значение параметра, буфер данных
  // Size - размер зарезервированного буфера
  // результат: если 0 - параметр успешно записан, отличное от 0 - ошибка (наприме маленький
размер буфера)
  TITPosGet = function(Ptr : Pointer; Key: PChar; var Value : CharArray; Size: Integer) :
Integer; cdecl;
  // запуск операции на терминале
  // входные данные
  // Ptr - указатель на объект
  // Cmd - номер команды, указанный в файле "ops.ini"
  TITPosRun = function(Ptr : Pointer; Cmd: Integer) : Integer; cdecl;

var
  // глобальный объект загруженной библиотеки
  lHandle : TLibHandle;
  // функция создания объекта взаимодействия с терминалом
  CreateITPos: TCreateITPos;
  // функция удаления объекта (после данной функции, использовать указатель нельзя,
необходимо снова вызывать CreateITPos)
  DeleteITPos: TDeleteITPos;
  // установка параметра
  ITPosSet: TITPosSet;
  // получение параметра
  ITPosGet: TITPosGet;
  // запуск операции на терминале
  ITPosRun: TITPosRun;

  // преобразываем буфер в строку (возможно не лучшее решение, в паскале не силён)
  // есть проблема с преобразованием буфера с русскими буквами (win1251), получается строка
из вопросов, видимо проблема в кодировках
function ArrayToString(const a : array of char) : string;
begin
  if Length(a) > 0 then
    SetString(Result, PChar(@a[0]), Length(a))
  else
    Result := '';
end;

```

```

// начало работы, обнуляем все указатели
procedure TArcusForm.FormCreate(Sender: TObject);
begin
    lHandle : = NILHandle;
    CreateITPos: = nil;
    DeleteITPos: = nil;
    ITPosSet: = nil;
    ITPosGet: = nil;
    ITPosRun: = nil;
end;

// процедура взаимодействия с терминалом
procedure TArcusForm.StartArcus(cmd : integer);
var
    Res : Integer;
    Buffer: CharArray;
    PosPtr: Pointer;
begin
    // если запускаем первый раз, то инициализируем библиотеку
    if lHandle = NILHandle then
    begin
        lHandle : = LoadLibrary('Arccom.dll');

        if lHandle <> NILHandle then
        begin
            // инициализируем функции взаимодействия
            Pointer(CreateITPos) := GetProcAddress(lHandle, 'CreateITPos');
            Pointer(DeleteITPos) := GetProcAddress(lHandle, 'DeleteITPos');
            Pointer(ITPosSet) := GetProcAddress(lHandle, 'ITPosSet');
            Pointer(ITPosGet) := GetProcAddress(lHandle, 'ITPosGet');
            Pointer(ITPosRun) := GetProcAddress(lHandle, 'ITPosRun');
        end;
    end;

    // если какая то из функций не проинициализирована, выходим (лучше выходить с кодом
    // ошибки)
    if @CreateITPos = nil then exit;
    if @DeleteITPos = nil then exit;
    if @ITPosSet = nil then exit;
    if @ITPosGet = nil then exit;
    if @ITPosRun = nil then exit;

    // создаём объект управления ПОС терминалом, данный объект передаём во всех остальных
    // функциях
    PosPtr: = CreateITPos();

    // если объект не создан, завершаем (лучше выходить с кодом ошибки)
    if PosPtr = nil then exit;

    // используем try на всякий случай, хотя сама библиотека не генерирует исключений
    try
        // указываем сумму операции
        ITPosSet(PosPtr, 'amount', PChar(AmountEdit.Text), -1);
        // указываем валюту
        ITPosSet(PosPtr, 'currency', PChar(CurrencyEdit.Text), -1);
        // запускаем операцию под номером "cmd"
        Res: = ITPosRun(PosPtr, cmd);

        // в переменной Res получаем результат работы ПОС терминала: 0 - успех, в остальных
        // случаях - ошибка
        ResultEdit.Text : = IntToStr(Res);

        // чистим массив для получения дополнительной информации о прошедшей операции
        FillChar(Buffer[0], sizeof(Buffer), #0);

        // запрашиваем RRN операции

```

```

ITPosGet(PosPtr, 'rrn', Buffer, sizeof(Buffer));

// другие переменные, которые можно запросить/установить:
// auth_code
// card_type
// date
// time
// aid
// trace_id
// response_code
// result_amount
// received_text_message
// terminal_id
// original_amount
// pan          - маскированный номер карты
// expiry       - срок действия
// rfu          - трек2 карты (только на установку)

MessageEdit.Text := ArrayToString(Buffer);
except
    showmessage('непредвиденная ошибка');
end;

// удаляем объект работы
DeleteITPos(PosPtr);
end;

// запускаем операцию ОПЛАТА
procedure TArcusForm.PurchaseButtonClick(Sender: TObject);
begin
    StartArcus(1);
end;

// запускаем операцию БАЛАНС (номер операции завит от настроек ops.ini)
procedure TArcusForm.BalanceButtonClick(Sender: TObject);
begin
    StartArcus(62);
end;

// запускаем меню администратора
procedure TArcusForm.AdminButtonClick(Sender: TObject);
begin
    StartArcus(99);
end;

end.

```

ПРИЛОЖЕНИЕ 3

ИНТЕГРАЦИЯ С ИСПОЛЬЗОВАНИЕМ COM OBJECT

Для работы с Connector рекомендуется следующая последовательность действий:

1. Создать Request экземпляр объекта *ISAPacketObj* для пересылки данных Server для проведения транзакции.
2. Создать Response экземпляр объекта *ISAPacketObj* для получения результатов транзакции от Server.
3. Создать экземпляр объекта *Connector Obj* для организации каналов обмена данными с Server.
4. Подготовить данные для проведения транзакции, заполнив соответствующие поля в *Request*.
5. Послать запрос на проведение транзакции вызовом метода *Exchange*.
6. Повторить предыдущий пункт столько раз, сколько транзакций требуется провести.

Описание PCPOSTConnectorObj

Метод Exchange

1. Обменяться информацией с Server, т.е. отправить серверу запрос на проведение транзакции и получить от него ответ.
2. Вызвать Connector. Обменяться данными (*Request*, *Response*, *Timeout*, *Result*)

- *Request* – Данные для проведения транзакции;
- *Response* – Результат проведения транзакции;
- *Timeout* – Таймаут на получение результатов проведения транзакции; (не используется)
- *Result* – значение, возвращаемое методом *Exchange*.

Свойство	№ поля	Описание поля
Amount	0	Сумма операции, выраженная в минимальных единицах валюты.
CurrencyCode	4	Код валюты операции.
DateTimeHost	6	Оригинальная дата и время совершения операции YYYYMMDDHHMMSS на хосте.
CardEntryMode	8	Способ ввода карты (возможные варианты: 1 – ручной, 2 – считыватель магнитной полосы, 3 – микропроцессорная карта вводится через микропроцессорный считыватель ПИН-пад).
PAN	10	Номер карты (заполняется совместно с полем 11, В этом случае.
CardExpiryDate	11	Срок действия карты YYMM (заполняется совместно с полем 10, в этом случае поле 12 не заполняется).
TRACK2	12	Данные дорожки 2 (в этом случае поле 10 и 11 не заполняются).
AuthorizationCode	13	Код авторизации (может возвращаться хостом).
ReferenceNumber	14	Номер ссылки.
ResponseCodeHost	15	Код ответа.
TextResponse	19	Дополнительные данные ответа.
DateTimeCRM	21	Дата и время проведения операции на терминале ГГГГММДДччммсс.
TrxID	23	Идентификатор транзакции в коммуникационном сервере.
OperationCode	25	Код операции.
TrxIDCRM	26	Уникальный номер транзакции.
CRMID	27	Идентификатор терминала на входе.

Свойство	№ поля	Описание поля
SLIP		Образ чека передается в объект response метода Exchange класса PCPOSTConnectorObj. Образ передается в кодировке указанной в параметре PRINTERCHARSET в cashreg.ini. Образ чека также сохраняется в файл. В случае, если с ПИН-пада не приходили команды на печать, свойство имеет значение <i>null</i>.  Файл из СОМ-объекта формируется ТОЛЬКО при печати на кассе!
CommodityCode		Код товара (тэг "\x1F\x1F").
PaymentDetails		Детали транзакции (req тэг "\x1F\x4E", rsp тэг "\x1F\x26").
ProviderCode		Код провайдера (тэг "\x1F\x4F").
ProcessingFlag		PIN флаг (тэг "\x1F\x61").
CommandResult		Код ответа RESPONSE_CODE.
CVV2		CVV2 (тэг "\x1F\x5D").
TerminalOutID		Идентификатор терминала (тэг "\x9F\x1C").
CashbackAmount		cashback - сумма операции, в минимальных денежных единицах (тэг "\x9F\x03").
RFU		Тоже самое поле, что и TRACK2. Сделано для поддержки не PA_DSS интерфейсов. (тэг "\x57").
BIN		Первые 6 цифр PAN.
Last4Digits		Последние 4 цифры PAN.
Hash		Хэш (SHA1) от PAN.
EncData		Шифрованные данные карты (тэг "\xA1\xA1").
Signature		Запуск механизма захвата подписи на терминале.
SignaturePath		Установить путь для сохранения файлов подписей.
TerminalIDX		Индекс для выбора терминала (USEJL).
TransactionID		Идентификатор транзакции в коммуникационном сервере (TPTP).

Для использования данных объектов необходимо создать в проекте ссылку на СОМ объект.

ПРИЛОЖЕНИЕ 4

ОПИСАНИЕ СТРУКТУРЫ USERAUTHINTFULL

```

typedef struct
{
int handle;
int abg_id;
int operType; // [in] Код операции (кассовый)
char rfu[InRFU];
char pan[InPan]; // [out] PAN
char expiry[ InExpiry ]; // [out] Expiry Date ГГММ
char pay_acc[InPan]; // не используется
char additional_payment_data[80]; // не используется
char amount[ InAmount ]; // [in] Сумма в копейках
char original_amount[ InAmount ]; // [in] Оригинальная сумма в копейках
char currency[ InCurrency ]; // [in] Код валюты
char terminalID[InTerminalID ]; // [out][in] ID терминала
char rrn[InRetrievalReference ]; // [in][out] Ссылка (только для операций для которых
// нужно, в остальных случаях - пуста)

char authCode[ InAuthIdentResponse ]; // [out][in] Код авторизации
char responseCode[InResponseCode ]; // [out] Код ответа
char cardType[InCardType]; // [out] Название типа карты
char date[InDate]; // [out] Дата транзакции
char time[InTime]; // [out] Время транзакции
char payment_data[InPayData]; // не используется
char data_to_print[InPayData]; // не используется
char home_operator[InPayData]; // не используется
char received_text_message[InReceivedTextMsg]; // не используется
char text_message[InReceivedTextMsg]; // [out] Расшифровка
char AID[InAID]; // [out] EMV AID
char ApplicationLabel[InApplicationLabel]; // [out] EMV ApplicationLabel
char TVR[InTVR]; // [out] EMV TVR
int system_res; // не используется
unsigned short receiptNumber; // номер чека
unsigned short Invoice; // номер платежа
int operatopn_type; // тип операции
char card_number[20]; // номер карты
char merchant[80]; // Идентификатор организации
char date_buffer[100]; //
unsigned short debit_number; // количество дебетовых операций
char debit_total[7]; // итог дебет
char debit_clear_total[7]; // итог дебет без учета комиссий и скидок
unsigned short return_number; // Количество возвратов
char return_total[7]; //
char return_clear_total[7]; //
unsigned long butch_num; // номер смены / пакета
char TrAmount[13]; //
char enc_data[64]; // [in][out] шифрованные данные карты (PAN) (для ПО
// с функцией поддержки HRS)

char cardholder_name[InCardholderName]; // имя владельца карты
char max_discount[InAmount]; // максимальная сумма скидки (в минимальных
// единицах валюты)

char min_discount[InAmount]; // минимальная сумма скидки (в минимальных
// единицах валюты)

char commission_amount[ InAmount+1 ]; // сумма комиссии (в минимальных единицах валюты)
char PaymentsData[102]; // для unipay (каждое поле должно заканчиваться
// «NULL»)
PaymentsData [0:2] – Идентификатор платежа;

```

char BIN[7];	PaymentsData [3:35] – Дополнительные данные 1;
char Hash[41];	PaymentsData [36:68] - Дополнительные данные 2;
char Last4Digits[7];	PaymentsData [69:101] - Дополнительные данные 3;
char isDiscountedGoods;	//BIN карты, поле должно заканчиваться «NULL»
char Balance[InAmount+1];	//Хэш номера карты (SHA1)
char TraceID[InTraceID];	//Последние 4 цифры номера карты
char OriginalDateTime[InDateTime];	//Признак дисконтных товаров в чеке
unsigned short CashReceiptNumber;	//Баланс карты
char OutPutTransactionData[256];	//Идентификатор предавторизации
	//Дата и время оригинальной операции
	//Номер кассового чека
	//Дополнительные данные транзакции (сообщение от хоста)
char TransactionID[19];	//Идентификатор транзакции
char ComissionOffline[InAmount];	//Офлайн комиссия
char ComissionOfflineCurrency[InCurrency];	//Валюта офлайн комиссии
char ComissionAcquirer[InAmount];	//Комиссия эквайера
char ComissionAcquirerCurrency[InCurrency];	//Валюта комиссии эквайера
char ComissionIssuer[InAmount];	//Комиссия эмитента
char ComissionIssuerCurrency[InCurrency];	//Валюта комиссии эмитента
char AuthorizedAmount[InAmount];	//Авторизованная сумма
char AuthorizedAmountCurrency[InCurrency];	//Валюта авторизованной суммы
char HostResponseCode[InResponseCode];	//код ответа от хоста
char HostDialectName[32];	//имя диалекта, по которому проводилась операция
unsigned char CardTypeLimitation;	// битовая маска. Используется для указания терминалу по какому типу карт проводить операцию (по банковской карте, топливной или карте лояльности):
	0 – автоопределение
	1 – банковская карта
	2 - топливная карта
	4 – карта лояльности
char RequestedDiscount[InAmount];	//сумма в минимальных единицах валюты, которую пользователь хочет оплатить баллами
	//режим ввода карты
unsigned char PosEntryMode;	
char NoICCREmoving;	
char Envelope9F70[LLC_ENVELOPELEN];	
char EqualDeal[LLC_EQUALDEALLEN];	
char HostDesc[LLC_HOSTDESCLEN];	
int EqualDealLen;	
int HostDescLen;	
union {	
struct TLV_data_t as_tlv;	//структурированная разметка тега, заполнять здесь
unsigned char storage[34];	// хранилище для тега (длина 2, макс.содерж. 32)
} Tag9F6E;	
int OriginalOperType;	//[in] Код оригинальной операции при отмене вне смены (только для продукта OW/Fuel+)
/* add boundary padding here if needed */	
}UserAuthIntFull;	
Enum {	
InProcCode = 7,	
InRFU = 60,	
InPan = 20,	
InExpiry = 5,	
InAmount = 13,	
InCurrency = 4,	
InTerminalID = 9,	
InSpdhTerminalID=17,	
InRetrievalReference = 13,	
InAuthIdentResponse = 9,	

```

InResponseCode = 4,
InCardType = 80,
InDate = 7,
InTime = 7,
InBatchNum = 8,
InRFUCredit = 50,
InPinblock = 17,
InPayData = 50,
InPayId = 3,
InMtid = 5,
InReceivedTextMsg=80,
InAID=80,
InApplicationLabel=80,
InTVR=80,
InCardholderName = 27,
InTraceID = 61,
InDateTime = 13
};

#define LLC_ENVELOPELEN 192
#define LLC_EQUALDEALLEN 48
#define LLC_HOSTDESCLEN 750

```

Передача дополнительной информации (Arcus 2.0)



В ARCUS 2 реализована возможность передачи дополнительных данных через результирующую структуру, которая будет заполнена после обработки (вызова ProcessOwFull)

```

typedef struct_packed {
    char lt_card_number[lnPan]; //номер карты лояльности
    uint8_t lt_card_flag;      //признак карты лояльности (кобрендинговая карта)
    uint8_t debts_flag;        //признак карты лояльности (дебетовая карта)
    char lt_balance[lnAmount]; //доступный баланс по карте лояльности в минимальных
                               //единицах валюты (LT_BALANCE)
    char lt_total[lnAmount];   //общий баланс по карте лояльности в минимальных
                               //единицах валюты (LT_ADD)
    char lt_minus[lnAmount];   //количество удержанных баллов в минимальных
                               //единицах валюты (LT_MINUS)
    char lt_plus[lnAmount];    //количество начисленных баллов в минимальных
                               //единицах валюты (LT_PLUS)
    char lt_max[lnAmount];     //максимальное количество бонусов, которые можно
                               //потратить (LT_MAX)
    char *client_info;         //в случае успешной операции, содержит маркетинговую
                               //информацию для клиента, в случае отказа – причину
                               //отказа (CL_INFO)
    size_t client_info_len;    //длина буфера данных для вывода информации для
                               //клиента (до 512)
    char *conected_cards;      //список присоединенных карт, для операции активации
                               //карты, разделенных '\\'
    size_t conected_cards_len; //длина буфера с данными о подключенных картах (до
                               //99), разделитель '\\'
} ProcessingResult_t;

```

ПРИЛОЖЕНИЕ 5

ПРИМЕР ИСПОЛЬЗОВАНИЯ БИБЛИОТЕК ARCCOM.DLL / LIBARCUS.SO.X.X.X.XX

Вызов UserAuthintFull



Перед использованием структуры ее необходимо обнулить функцией `memset()`.

```
#include <windows.h>
#include <iostream>
#include <string>
enum
{
    InProcCode = 7,
    InRFU = 60,
    InPan = 20,
    InExpiry = 5,
    InAmount = 13,
    InCurrency = 4,
    InTerminalID = 9,
    InSpdhTerminalID = 17,
    InRetrievalReference = 13,
    InAuthIdentResponse = 9,
    InResponseCode = 4,
    InCardType = 80,
    InDate = 7,
    InTime = 7,
    InBatchNum = 8,
    InRFUCredit = 50,
    InPinblock = 17,
    InPayData = 50,
    InPayId = 3,
    InMtid = 5,
    InReceivedTextMsg = 80,
    InAID = 80,
    InApplicationLabel = 80,
    InTVR = 80,
    InCardholderName = 27,
    InTraceID = 61,
    InDateTime = 13
};
```

```
#ifdef _MSC_VER
#pragma pack(push, 1)
#endif
```

```
typedef struct
{
    int handle;
    int abg_id;
    int operType;
    char rfu[InRFU];
    char pan[InPan];
    char expiry[InExpiry];
```

```

char pay_acc[InPan];
char additional_payment_data[80];
char amount[InAmount];
char original_amount[InAmount];
char currency[InCurrency];
char terminalID[InTerminalID];
char rrn[InRetrievalReference];
char authCode[InAuthIdentResponse];
char responseCode[InResponseCode];
char cardType[InCardType];
char date[InDate];
char time[InTime];
char payment_data[InPayData];
char data_to_print[InPayData];
char home_operator[InPayData];
char received_text_message[InReceivedTextMsg];
char text_message[InReceivedTextMsg];
char AID[InAID];
char ApplicationLabel[InApplicationLabel];
char TVR[InTVR];
int system_res;
unsigned short receiptNumber;
unsigned short Invoice;
int operatopn_type;
char card_number[20];
char merchant[80];
char date_buffer[100];
unsigned short debit_number;
char debit_total[7];
char debit_clear_total[7];
unsigned short return_number;
char return_total[7];
char return_clear_total[7];
unsigned long butch_num;
char TrAmount[13];
char enc_data[64];
char cardholder_name[InCardholderName];
char max_discount[InAmount];
char min_discount[InAmount];
char commission_amount[InAmount + 1];
char PaymentsData[102];

// данные для отправки на хост
// данные для печати на чеке
// данные для печати на чеке

//в unipay (каждое поле должно быть нул-
//терминировано)
PaymentsData [0:2] – идентификатор платежа;
PaymentsData [3:35] – Поле платежа, значение 1;
PaymentsData [36:68] - Поле платежа, значение 2;
PaymentsData [69:101] - Поле платежа, значение 3;

char BIN[7];
char Hash[41];
char Last4Digits[7];
char isDiscountedGoods;
char Balance[InAmount + 1];
char TraceID[InTraceID];
char OriginalDateTime[InDateTime];
unsigned short CashRecieptNumber;
char OutPutTransactionData[256];
char TransactionID[19];
char ComissionOffline[InAmount];
char ComissionOfflineCurrency[InCurrency];
char ComissionAcquirer[InAmount];
char ComissionAcquirerCurrency[InCurrency];
char ComissionIssuer[InAmount];

```

```

char ComissionIssuerCurrency[InCurrency];
char AuthorizedAmount[InAmount];
char AuthorizedAmountCurrency[InCurrency];
char HostResponseCode[InResponseCode];
char HostDialectName[32];
unsigned char CardTypeLimitation;
char RequestedDiscount[ InAmount ];

unsigned char PosEntryMode;
char NoICCREmoving;
char Envelope9F70[LLC_ENVELOPELEN];
char EqualDeal[LLC_EQUALDEALLEN];
char HostDesc[LLC_HOSTDESCLEN];
int  EqualDealLen;
int  HostDescLen;
union {
    struct TLV_data_t as_tlv;
    unsigned char storage[34];
} Tag9F6E;
int OriginalOperType;

unsigned int LoyaltyCode;
char CashbackAmount[InAmount + 1];
char BankID[20];
unsigned int applicationType;

/* add boundary padding here if needed */
}
UserAuthIntFull;
#ifdef _MSC_VER
#pragma pack(pop)
#endif
typedef int(__cdecl *_PCPOSFUNCFULL)(UserAuthIntFull *auth_st_full, int size);
int main(void) {
    HMODULE hLib =
    LoadLibrary("D:\\work\\workspaces\\arcus_workspace\\arccom\\ArcCom___Win32_Debug_MinDependency\\
    ArcCom.dll");
    _PCPOSFUNCFULL ProcessOwFull = (_PCPOSFUNCFULL)GetProcAddress(hLib, "ProcessOwFull");
    UserAuthIntFull auth;
    memset(&auth, 0, sizeof(UserAuthIntFull));
    auth.operType = 1;
    strncpy_s(auth.amount, sizeof(auth.amount), "1234", 4);
    int res = ProcessOwFull(&auth, sizeof(UserAuthIntFull));
    FreeLibrary(hLib);
    return 0;
}

```

//признак процессинга, см. CARD_TYPE_*
 //сумма в минимальных единицах валюты, которую
 пользователь хочет оплатить баллами
 //режим ввода карты
 //содержимое тега 9F70 карточки (проект LLC)
 //содержимое тега EqualDeal (проект LLC)
 //содержимое тега HostDesc (проект LLC)
 //длина тега EqualDeal (проект LLC)
 //длина тега HostDesc (проект LLC)
 //[in] Код оригинальной операции при отмене вне
 смены (только для продукта OW/Fuel+)
 //код лояльности для ПО NW_TPTP_15
 //сумма выдачи наличных для PWCB
 //идентификатор банка для TITP_ENG
 //[in] Идентификатор терминального приложения
 (только для продукта OW/Fuel+)

Вызов Результирующей структуры

```

/**
 * \brief Sets structure for getting processing results.
 *
 * \param [in] func - pointer to ProcessingResult_t structure.
 */
DLL_EXPORT void __cdecl SetProcessingResultStruct(ProcessingResult_t * result, size_t
struct_length);

```

Пример libarcus.so.x.x.x.xx

```
#include <stdio.h>
#include <dlfcn.h>
#include <string.h>
#include "arcus.h"
BasketOfGoods_t cur_basket;
GoodsItem_t items[2];
int PrepareGoodsBasket(){
    cur_basket.version = 1;
    cur_basket.curcode = 643;
    cur_basket.total_amount = 14513; //145 py6. 13 коп.
    snprintf(cur_basket.sale_number, sizeof(cur_basket.sale_number), "123456789012");
    cur_basket.sale_number_len = strlen(cur_basket.sale_number);
    cur_basket.items_number = 2;
    cur_basket.items = items;
    items[0].version = 1;
    snprintf(items[0].product_name, sizeof(items[0].product_name), "Fanta");
    items[0].product_name_len = strlen(items[0].product_name);
    snprintf(items[0].commodity_code, sizeof(items[0].commodity_code), "00100062");
    items[0].commodity_code_len = strlen(items[0].commodity_code);
    items[0].dispenser_number = 0;
    items[0].measure = 'U';
    items[0].exponenta = 0;
    items[0].quantity = 1;
    items[0].price = 7513;
    items[0].amount = 7513;
    items[1].version = 1;
    snprintf(items[1].product_name, sizeof(items[1].product_name), "AI-92");
    items[1].product_name_len = strlen(items[1].product_name);
    snprintf(items[1].commodity_code, sizeof(items[1].commodity_code), "00100074");
    items[1].commodity_code_len = strlen(items[1].commodity_code);
    items[1].dispenser_number = 3;
    items[1].measure = 'U';
    items[1].exponenta = 2;
    items[1].quantity = 200;
    items[1].price = 3500;
    items[1].amount = 7000;
    return 0;
}

BasketOfGoods_t * GetGoodsBasket(void){
    return &cur_basket;
}

int MessageToStdout(const char *message, void *userdata){
    printf("%s", message);
    return 0;
}

int main( int argc, char ** argv){
    UserAuthIntFull auth;
    DialogsCBs dialogs;
    memset(&dialogs, 0, sizeof(DialogsCBs));
    memset(&auth, 0, sizeof(UserAuthIntFull));
    //the basket must to be prepared before calling ProcessOWFull
    PrepareGoodsBasket();
    //set call-back for geting basket of goods
    SetBasketOfGoodsCB( GetGoodsBasket);
    dialogs.size = sizeof(DialogsCBs);
    dialogs.cbInfo = MessageToStdout;
    dialogs.cbStatus = MessageToStdout;
    SetDialogsCBs(&dialogs);
    auth.operType = 1;
    snprintf(auth.amount, lnAmount -1, "14513");
    snprintf(auth.currency, lnCurrency - 1, "643");
    auth.CardTypeLimitation |= CARD_TYPE_PETROL; //запрос оплаты по топливной карте
    char RequestedDiscount[ lnAmount ]; //сумма в минимальных единицах валюты,
    которую пользователь оплатит баллами
    //all ready to beginig paymant
    ProcessOWFull(&auth, sizeof(UserAuthIntFull));
    return 0;
}
```

ПРИЛОЖЕНИЕ 6

ПРОТОТИПЫ ФУНКЦИЙ

ДИАЛОГОВ ARCUS.H

```
/**
 * \brief Menu dialog          - описание меню;
 * \param caption [in]        - заголовок меню;
 * \param items [in]          - указатель на пункт меню;
 * \param number [in]         - номер пункта меню;
 * \param timeout [in]        - время ожидания выбора (миллисекунды);
 * \param userdata [in]       - указатель на данные пользователя (см. DialogsCBs);
 * \функция возвращает номер выбранного пункта меню или «-1» в случае отказа.
 */
typedef int (__cdecl *cbMenu)(const char *caption, const char **items, int number,
uint32_t timeout, void *userdata);
/**
 * \brief Yes/No dialog        - диалог да/нет;
 * \param caption [in]        - заголовок диалога;
 * \param message [in]        - сообщение;
 * \param timeout [in]        - время ожидания выбора (миллисекунды);
 * \param userdata [in]       - указатель на данные пользователя (см. DialogsCBs);
 * \функция возвращает значение 1 - да, 0 - нет.
 */
typedef int (__cdecl *cbYesNoDialog)(const char *caption, const char *message, uint32_t
timeout, void *userdata);
/**
 * \brief Warnig dialog        - важная информация;
 * \param message [in]        - сообщение;
 * \param timeout [in]        - время ожидания выбора (миллисекунды);
 * \param userdata [in]       - указатель на данные пользователя (см. DialogsCBs);
 * \возвращает 0 или -1 по истечении таймаута.
 */
typedef int (__cdecl *cbWarningDialog)(const char *message, uint32_t timeout, void
*userdata);
/**
 * \brief Information message to the client - краткая информация для клиентов;
 * \param message [in]        - сообщение;
 * \param userdata [in]       - указатель на данные пользователя (см. DialogsCBs);
 * \функция возвращает 0 - если успешно, -1 в случае ошибки.
 */
typedef int (__cdecl *cbInfo)(const char *message);
/**
 * \brief Terminal status message - статус терминала;
 * \param message [in]        - сообщение;
 * \param userdata [in]       - указатель на данные пользователя (см. DialogsCBs);
 * \функция возвращает 0 - если успешно, -1 в случае ошибки.
 */
typedef int (__cdecl *cbStatus)(const char *message, void *userdata);
/**
 * \brief Edit string dialog    - диалог для изменения введенных данных;
 * \param caption [in]        - заголовок диалога;
 * \param data [in]           - данные для изменения
 * \param buffer [out]        - исходящий буфер;
 * \param buffersize [in]     - размер исходящего буфера;
 * \param mask [in]           - маска вводимых данных;
 * \param mode [in]           - режим ввода;
 * \param timeout [in]        - время ожидания выбора (миллисекунды);
 * \param userdata [in]       - указатель на данные пользователя (см. DialogsCBs);
 * \функция возвращает 0 - если успешно, -1 в случае ошибки.
 */
typedef int (__cdecl *cbStringEdit)(const char *caption, const char *data, char *buffer,
size_t buffersize, const char *mask, const char *mode, uint32_t timeout, void *userdata);
typedef int (__cdecl *cbI344Window)(void);
typedef int (__cdecl *cbI344DisplayLine)(const char *text, unsigned int line, unsigned int
column, unsigned int inv_type);
typedef int (__cdecl *cbI344ClearLine)(unsigned int line);
typedef int (__cdecl *cbI344KeyReq)(unsigned int to);
```

Пример использования

```
DialogsCBS callbacks;
DialogsI344CBS i344callbacks;

if (use_console_dialogs != 0)
{
    callbacks.userdata = nullptr;
    callbacks.size = sizeof(callbacks);
    callbacks.cbMenuItem = callback_menu;
    callbacks.cbYesNoDialog = callback_yesno;
    callbacks.cbWarningDialog = callback_warning;
    callbacks.cbInfo = callback_info;
    callbacks.cbStatus = callback_status;
    callbacks.cbStringEdit = callback_stringedit;

    i344callbacks.size = sizeof(i344callbacks);
    i344callbacks.cbI344OpenWindow = callback_i344openwindow;
    i344callbacks.cbI344ClearWindow = callback_i344clearwindow;
    i344callbacks.cbI344DisplayLine = callback_i344displayline;
    i344callbacks.cbI344ClearLine = callback_i344clearline;
    i344callbacks.cbI344KeyReq = callback_i344keyrequest;

    ITPosSet(itpos, "DialogsCBS", std::to_string((long)&callbacks).c_str(), -1);
    ITPosSet(itpos, "DialogsI344CBS", std::to_string((long)&i344callbacks).c_str(), -1);
}
```

ПРИЛОЖЕНИЕ 7

РЕКОМЕНДУЕМЫЙ ПЕРЕЧЕНЬ ОПЕРАЦИЙ ПРИ ИНТЕГРАЦИИ

Интерфейс кассира кассового ПО должен поддерживать вызов модуля ARCUS 2.1 для выполнения следующих финансовых и административных операций:

- **ТОРГОВЛЯ – (БАНК- ВЫДАЧА НАЛИЧНЫХ)**

- 1 ОПЛАТА ТОВАРА (ВЫДАЧА НАЛИЧНЫХ)
- 2 ВОЗВРАТ ТОВАРА (КРЕДИТ, ПОПОЛНЕНИЕ СЧЕТА)
- 3 ОТМЕНА ПОСЛЕДНЕЙ ПОКУПКИ (ОТМЕНА ВЫДАЧИ / КРЕДИТА)
- 4 ЗАКРЫТИЕ_ДНЯ (СВЕРКА ИТОГОВ СМЕНЫ, ПАКЕТА, ДНЯ — Z отчет)
- 5 ПЕЧАТЬ ЖУРНАЛА ОПЕРАЦИЙ
- 6 МЕНЮ_КАССИРА (МЕНЮ АДМИНИСТРАТОРА)

- **ОТЕЛИ, АЗС, РЕСТОРАНЫ**

- 7 ПРЕАВТОРИЗАЦИЯ (резервирование суммы) [опция]
- 8 ЗАВЕРШЕНИЕ РАСЧЕТА ПО ПРЕАВТОРИЗАЦИИ [опция]

- **ПРИ НАЛИЧИИ ТЕХ.ВОЗМОЖНОСТИ АРМ ТОРГОВЛИ**

- 9 ОТМЕНА ОПЕРАЦИИ ПО НОМЕРУ ЧЕКА [опция]

В большинстве случаев в ПО терминала определены 3 операции:

- ВОЗВРАТ ТОВАР - служит для возврата денежных средств на счёт клиента. В большинстве случаев выполняется если начальная операция оплаты была в другой смене (на другом терминале). Для проведения операции обязательно запрашивается карта клиента. Другие данные могут отличаться (номер чека, RRN, сумма) и настраиваются при конфигурации терминала. «Возврат» является кредитовой операцией с перечислением средств на счёт клиента, поэтому обрабатывается банком дольше (от 3х до 40 дней).
- ОТМЕНА ПОСЛЕДНЕЙ - это технологическая отмена, при которой не запрашиваются данные о проведенной транзакции (номер чека, номер RRN, PAN карты и д.р.). Команда вызывается кассовым ПО в случае, когда необходимо отменить последнюю успешную оплату (строго говоря, можно отменять и другие успешные транзакции, но необходимо уточнять такую возможность для каждого конкретного процессинга). В случае если транзакцию отменить нельзя, терминал выдаст сообщение "Невозможно отменить"
- УНИВЕРСАЛЬНАЯ ОТМЕНА - служит для отмены с обязательным указанием данных о начальной транзакции и предъявлением карты клиента. Запрашиваемые данные могут быть различны, и настраиваются при конфигурации терминала. В большинстве случаев Отмену можно проводить только если начальная операция оплаты была в этой же смене. В этом случае, чаще всего, для поиска начальной транзакции в журнале терминала используется RRN (хотя возможно использовать и номер чека). Операция «Отмена» обрабатывается на хосте достаточно быстро и средства поступают обратно на счёт клиента.



В связи с тем, что из структуры UserAuthIntFull можно получить RRN, желательно его и использовать для операций «Универсальная отмена» и «Возврат».



Использование каждой из них необходимо обговаривать непосредственно с банком.

ПРИЛОЖЕНИЕ 8

ФУНКЦИОНАЛ DLS (ОПЦИОНАЛЬНО, ARCUS 2.0)

Данный функционал доступен для процессинга OpenWay.

Функционал реализует применение системы лояльности и фискализации DLS – поддержка протокола АПДК версии 8.1.2.

Для подключения функционала необходимо обновить ArcCom.dll и CommandLineTool.exe.

Поддерживается 2 режима интеграции с кассой: полная и легкая интеграции.

Режим легкой интеграции с кассой

Обеспечивает работу системы лояльности без модернизации ПО кассы.

Основан на использовании специализированных кодов валют (для операций Оплата / Возврат).

- 999 - будет выполнена операция Оплата за бонусы / Возврат за бонусы;
- 998 - будет выполнена операция Оплата за наличные с начислением бонусов / Возврат за наличные со списанием бонусов.

Режим полной интеграции с кассой

Оплата

Стандартный запуск операции.

В режиме работы с кассой опционально поддерживается передача на хост DLS списка товаров.

Для этого необходимо заполнить структуру UserAuthIntFull и передать ее функции

DisProcessOw(UserAuthIntFull *auth_st, char * data, unsigned int data_size);

где data - указатель на список товаров.

Оплата наличными с начислением бонусов и без

Аналогично операции оплата. В режиме работы с кассой поддерживает режим Silent, когда не запрашивается на предъявление карта и не печатается чек.

Этот режим включается записью '1' в поле PaymentsData[102] структуры UserAuthIntFull.

Возврат со списанием бонусов и без

Операция возврат подразумевает передачу на терминал:

- список товаров (аналогично операции оплата);
- rrn (поле rrn структуры UserAuthIntFull).

При отсутствии rrn будет выведен диалог для ввода rrn. Если терминал не смог найти по rrn дату проведения операции, будет выведен диалог для ввода даты.

Открытие смены

Подразумевает передачу на терминал:

- код кассира (поле rrn структуры UserAuthIntFull);
- (опционально) фио кассира (поле PaymentsData[102] структуры UserAuthIntFull).

При отсутствии кода кассира будет выведен диалог для ввода кода.

Инкассация

Подразумевает передачу на терминал:

- внесенная сумма (поле amount структуры UserAuthIntFull);
- изъятая сумма (поле original amount структуры UserAuthIntFull).

При отсутствии данных будет выведен диалог для ввода.

Работа с функционалом DLS с модулем SwitchSales

С Января 2021 года в Universal EMV POS NW_TPTP_15 реализован функционал работы с лояльностью DLS через плагин SwitchSales.

В режиме работы с кассой, для запуска операции необходимо передавать следующий набор полей в структуре UserAuthIntFull:

```
unsigned int swsActionType  
unsigned int swsRespCode;  
char swsRespData[100 + 1];  
char swsAcqId[50 + 1];  
char swsCardHash[100 + 1];
```

При вызове SwsProcessOw(UserAuthIntFull *auth_st_full, char *additional_data, unsigned int additional_data_size) можно передавать поле ADATA.

Обратная совместимость обеспечена поддержкой вызова функции ProcessOw_dls(UserAuthInt *auth_st, char *data, unsigned int data_size).

ПРИЛОЖЕНИЕ 9

ОПИСАНИЕ ВНУТРЕННИХ КОДОВ ОТВЕТОВ



Раздел содержит описание внутренних кодов ответа, которые платежное приложение передает на кассу.
При интеграции с некоторыми специальными версиями ПО Universal EMV POS коды ответа подменяются на те, которые правильно интерпретируются ПО.

Внутренние коды ответа для Universal EMV POS (NewWay)

Код ответа	Расшифровка
000	успешная операция
086	ошибка при вводе пина
088	ошибка при расчете MAC
096	ошибка в конфигурации, для ПО не MULTIHOST – ошибка сверки итогов
099	ошибка при определении карты
100	выполнение транзакции прервано
200	ошибка при сверке итогов
201	отказ ввода номера чека
202	оригинальная транзакция не найдена
203	отказ ввода суммы при подготовке операции
204	отказ ввода оригинальной суммы при подготовке операции
205	отказ ввода RNN (уникальной идентифицирующей информации)
206	отказ подтверждения операции типа отмены (SmartVista)
207	отказ ввода ID предавторизации
208	зарезервировано для SV_05
209	ошибка построения списка (меню) доступных валют / отказ выбора валюты
210	ошибка поиска номера чека / для ПО не MULTIHOST – отказ ввода кода авторизации
211	ошибка поиска транзакции в логе
212	таймаут ввода карты
213	отказ ввода карты кассира
214	отказ ручного ввода PAN карты
215	отказ подтверждения типа карты
216	отказ выбора операции
217	ошибка сбора данных оригинальной операции для операции отмена/возврат (например, введены не все данные, в журнале нет оригинальной операции и пр.) или оригинальная операция выполнялась по другой карте
218	операция уже отменена
219	отказ ввода кода авторизации
221	зарезервировано для SV_05
222	нет PAN, для ПО не MULTIHOST – подмена вместо 211
223	ошибка связи с кассой, для ПО не MULTIHOST подмена 222
224	зарезервировано для SV_05 – подмена вместо 201
225	зарезервировано для SV_05 – подмена вместо 202
226	зарезервировано для SV_05 – подмена вместо 203
227	зарезервировано для SV_05 – подмена вместо 204

Код ответа	Расшифровка
228	зарезервировано для SV_05 – подмена вместо 205
229	зарезервировано для SV_05 – подмена вместо 206
230	отказ ввода кода авторизации
231	зарезервировано для SV_05 – подмена вместо 207
232	зарезервировано для SV_05 – подмена вместо 209
233	карта не прочитана ни в одном интерфейсе (ошибка чтения MS)
234	ошибка при чтении чипа карты
235	сумма отмены превышает сумму исходной операции (TE_03) сумма операции завершение предавторизации превышает сумму предавторизации, более чем на разрешенное значение
236	отменяемая операция не является последней
237	частичная отмена не разрешена
240	TMS ID не определен для сессии TMS (SV_05)
241	Batch не закрыт для сессии TMS (SV_05)
242	нет возможности установить скрипт сессии TMS (SV_05)
243	ошибка при выполнении TMS сессии (SV_05)
250	карта была вытащена из ридера до окончания операций с ней
301	отказ выбора виртуального терминала
302	невозможно отменить операцию
303	нужно закрыть смену
304	валюта не доступна
305	карта не обслуживается
320	отказ от ввода подписи
321	неверная сумма
322	ошибка ID-виртуального терминала
323	не доступен чип ридер - (актуально для I9500 и IUN)
324	бесконтактный ридер не обнаружен или неисправен
325	пин-пад деактивирован
330	отказ от ввода
331	неверный параметр запроса
332	отказ от ввода номера телефона
333	ошибка передачи номера телефона
334	ошибка обновления корзины товаров (ответ на операцию 1.108)
335	слишком много товаров в списке (ответ на операцию 1.108)
336	получены новые лимиты при обновлении списка товаров (ответ на операцию 9.41)
356	iUN обнаружил проблему, но источник проблемы неизвестен
401	ошибка чтения карты кассира
402	транспорт не поддерживается
403	в конфигурации отсутствуют точки доступа для доступа к хосту
404	ошибка ответа хоста
405	нет отложенных операций
410	ошибка загрузки рабочего ключа
411	таймаут обнаружения карты
412	ошибка расшифровки HRS криптограммы
500	закончилось место для хранения слипов, для ПО не MULTIHOST – подмена 988
600	транзакция отклонена хостом

Код ответа	Расшифровка
900	ошибка при установке соединения – текст, для ПО не MULTIHOST – подмена 998
915	код ошибки для проектов спец. карт
941	ошибка генерации списка операций
970	временный код ответа для проекта ELGAVA_TRANSPORT
971	код проекта ELGAVA_TRANSPORT
984	рестарт терминала по повторной команде с кассы
985	CLESS MIFARE
987	таймаут чтения MS карты
988	ошибка чтения MS карты (определения интерфейса карты)
989	ошибка формата MS карты
990	отказ проведения операции
991	ошибка при проверке срока действия карты
992	команда для кассы OPDET вернула 0 или ошибку
993	отказ ввода PIN, для ПО не MULTIHOST – подмена 305
994	ошибка при работе с модулем шифрования
995	ошибка при вводе ключа с кассы
996	не верный формат команды с кассы
997	проверка деталей платежа - подпись
998	проверка деталей платежа
999	успешно подготовлены диапазоны чеков

Внутренние коды ответа для Universal EMV POS (Unipay)

Код ответа	Расшифровка
000	успешная операция
050	выполнение транзакции прервано
250	карта была вытащена из ридера до окончания операций с ней
304	валюта не доступна
984	предыдущая команда не закончена
991	ошибка выбора кассира
992	ошибка чтения EMV карты или невозможно обработать команду от кассы
998	ошибка при установке соединения
Специальные коды ответа (Карты «Лента» TRTP_15 с версии 2.0.0.32)	
099	специальный отказной код ответа при котором касса будет пересчитывать суммы / начисления бонусов для операции
005	ошибка второго этапа транзакции - предъявлена другая карта либо карта не попадает в лояльность, переданную с кассы

ПРИЛОЖЕНИЕ 10

ПЛАТЕЖИ В СТОРОНУ ТРЕТЬИХ ЛИЦ В РЕЖИМЕ РАБОТЫ С КАССОЙ (TRTP, SPDH, ARCUS 2.0)

Для настройки платежей в сторону третьих лиц в режиме работы с кассой:

1. добавьте в файл OPS.INI строчку:
 - для приложения NewWay: 27=0,16,ПЛАТЕЖ
 - для приложения Unipay: 27=0,138,ПЛАТЕЖ
2. задайте настройки для платежей в сторону третьих лиц в TMS:
 - для приложения NewWay TRTP подробнее см. Руководство Администратора TMS NewWay - Приложение Платежи и переводы - Общие настройки платежей и Utility Payment - Для процессинга TRTP.
 - для приложения Unipay подробнее см. Руководство Администратора TMS Unipay, пункт 5.18.1 Платежи в сторону третьих лиц.

Пример вызова операции с кассы:

```
CommandLineTool.exe /o30 /a10000 /c643 --payment-data=12345678901234;050214;000000025000
```

где в ключе --payment-data передаются данные в формате протокола автоизации:

- 12345678901234 - дополнительные данные о платеже (опционально);
- 050214 - дата в формате ДД/ММ/ГГ;
- 000000025000 - Сумма операции.