Программный модуль операционной системы смарт-карт, реализующий методы хранения и доступа к данным, сохраняемым в долговременной памяти

Руководство оператора

Версия: 1.0 от 07.03.2023

Количество листов: на 19 листах

Москва, 2023

Общество с ограниченной ответственностью «СКОН ТЕХНОЛОГИИ»
Юридический адрес: 115230, город Москва, Электролитный проезд, д. 1 к. 3, помещ./этаж х/3 ком. 9, 13
ИНН 9726018244, КПП 772601001, р/с 40702810000000256170 в АО "Райффайзенбанк", г. Москва,
к/с 30101810200000000700, БИК 044525700, ОГРН 1227700461790
+7 925 011 30 07 | www.sconetech.ru

АННОТАЦИЯ

В данном программном документе приведено руководство оператора по настройке и проверке программного модуля операционной системы смарт-карт, реализующего методы хранения и доступа к данным, сохраняемым в долговременной памяти.

СОДЕРЖАНИЕ

1. ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

Таблица 1. Термины и определения

APDU	Application Protocol Data Unit – тип управляющих команд, используемых для интегральных схем. ГОСТ Р ИСО/МЭК 7816-4-2013 «Карты идентификационные. Карты на интегральных схемах. Часть 4. Организация, защита и команды для обмена»
API	API (программный интерфейс приложения, интерфейс прикладного программирования) (англ. application programming interface, API [эй-пи-ай]) – описание способов (набор классов, процедур, функций, структур или констант), которыми одна программа может взаимодействовать с другой программой.
NVM	Non-volatile memory (NVM) or non-volatile storage – это тип памяти которая может сохранить информацию даже после отключения питания
МК	Микроконтро́ллер (англ. Micro Controller Unit, MCU) – микросхема, предназначенная для управления электронными устройствами, банковскими картами
ОС	Операционная система смарт-карт (англ. operating system smart cards, OS smart cards) – комплекс взаимосвязанных программ, предназначенных для управления ресурсами МК и организации взаимодействия с пользователем по средствам интерфейсов ввода\вывода.
ПО	Программное обеспечение
Платежное приложение (ПП)	Прикладное программного обеспечение, сертифицируемое ПС
Платежная система (ПС)	Платежные системы осуществляют перевод финансовых средств (денег, чеков, ценных бумаг, сертификатов, условных платёжных единиц) в электронном или реальном виде. Платежная система являет собой совокупность определенных процедур, правил и технической инфраструктуры для передачи стоимости одним субъектом экономики другому

2. НАЗНАЧЕНИЕ ПРОГРАММЫ

2.1. Функциональное назначение программы

Программный модуль операционной системы смарт-карт, реализующий методы хранения и доступа к данным, сохраняемым в долговременной памяти, предназначен для организации доступа к долговременной памяти в процессе жизненного цикла операционной системы смарт-карт (далее ОС смарт-карт), платежных приложений (далее ПП), позволяющих осуществлять банковские транзакции, в соответствии с требованиями платежных систем МИР, VISA, MasterCard, не платежных, дополнительных приложений (далее доп. приложения) – это приложения, предназначенные для идентификации и аутентификации владельца карты, при ее применении для проезда на общественном транспорте, использовании в социальной сфере (социальные карты), использовании карты в системах контроля и управления доступом и др.

2.2. Эксплуатационное назначение программы

Программный модуль операционной системы смарт-карт обеспечивает надежное, безопасное хранение данных пользователя, приложения и операционной системы в долговременной памяти.

2.3. Описание функции модуля

Вызов функций модуля осуществляется при помощи интерфейса APDU команды со следующим синтаксисом:

2.3.1. Функция распределение блока данных (файла) для приложения

Создается файла заданного размера, типа, маски. Возвращаемым значением является дескриптор файла. Значение MBM_INVALID_HANDLE воспринимается как ошибка

```
/**
 * Создание файла
 * size: размер запрашиваемого файла
 * sharedOption: тип доступа
 * maskedOption: тип маскирования
 * возвращаемое значение: значение дескриптора выделенного файла или
 * MBM_INVALID_HANDLE, воспринимаемое как ошибка
 */
```

```
MbmHandler_t    Mbm_Allocate(
                uint16_t size,
                FileShareOptions_t      sharedOption,
                FileMaskOption_t        maskedOption)
```

Синтаксис команды представлен в таблице 2.

Таблица 2. Структура команды

Поле	**Значение**	**Описание**
CLA	'80'	
INS	‘F6’	
P1	‘01’	Создание файла
P2	‘YY’	Атрибут создаваемого файла. 0 – без маскирования, 1 – с маскированием (только для этой команды)
L_C	‘02’	Длина данных команды
OptionShare[1..2]		Длина создаваемого файла

Смарт карта возвращает последовательность элементов следующего вида, приставленных в таблице 3.

Таблица 3. Ответные данные

Длина	**Значение**
2	Статус выполнения функции
2	Дескриптор созданного файла

2.3.2. Функция чтения данных из блока (файла)

Чтение данных из файла заданного дескриптора по определённому смещении и длине в приемный буфер. Возвращаемое значение – это статус выполнения операции.

```
/**
 * Чтение данных из файла
 * hFile: дескриптор файла для чтения
 * filePos: смещение от начала файла
 * p_dst: приемный буфер, куда будут помещены данные
 * dataLen: длина запрашиваемых данных
```

```
 * возвращаемое значение: errOk, остальное воспринимается как ошибка
 */
ResultCls_t Mbm_Read(
                MbmHandler_t hFile
                uint16_t filePos,
                void huge* p_dst,
                uint16_t dataLen)
```

Синтаксис команды представлен в таблице 4.

Таблица 4. Структура команды

Поле	**Значение**	**Описание**
CLA	'80'	
INS	‘F6’	
P1	‘04’	Чтение данных из блока (файла)
P2	‘00’	
L_C	‘06’	
Handle1	‘AA’	Старший байт дескриптора файла
Handle2	‘BB’	Младший байт дескриптора файла
Смещение1	‘CC’	Старший байт смещения в файле
Смещение2	‘DD’	Младший байт смещения в файле
Длина1	‘EE’	Старший байт длины запрашиваемых данных
Длина2	‘FF’	Младший байт длины запрашиваемых данных

Смарт карта возвращает последовательность элементов следующего вида, приставленных в таблице 5.

Таблица 5. Ответные данные

Длина	**Значение**
2	Статус выполнения функции
Переменная	Массив данных файла

2.3.3. Функция записи данных в блок (файл)

Запись в файл заданного дескриптора по определённому смещению и длине из приемного буфера. Возвращаемое значение – это статус выполнения операции.

```
/**
 * Запись в файл
 * hFile: дескриптор файла, куда будет производится запись
 * filePos: смещение от начала файла
 * p_src: буффер с исходными данными для записи
 * dataLen: длина записываемых данных
 * возвращаемое значение: errOk, остальное воспринимается как ошибка
 */
ResultCls_t Mbm_Write(
                MbmHandler_t hFile,
                uint16_t filePos,
                void huge* p_src,
                uint16_t dataLen)
```

Синтаксис команды представлен в таблице 5.

Таблица 5. Структура команды

Поле	**Значение**	**Описание**
CLA	'80'	
INS	‘F6’	
P1	‘03’	Запись данных в блок (файл)
P2	‘00’	
L_C	‘XX’	6+Размер данных
Handle1	‘AA’	Старший байт дескриптора файла
Handle2	‘BB’	Младший байт дескриптора файла
Смещение1	‘CC’	Старший байт смещения в файле
Смещение2	‘DD’	Младший байт смещения в файле
Длина1	‘EE’	Старший байт длины записываемых данных
Длина2	‘FF’	Младший байт длины записываемых данных
Данные[1..N]		Массив данных для записи

Смарт карта возвращает последовательность элементов следующего вида, приставленных в таблице 6.

Таблица 6. Ответные данные

Длина	**Значение**
2	Статус выполнения функции

2.3.4. Функция начала транзакции изменения NVM

Функция открывает транзакцию, перезаписываемые данные помещаются в транзакционный буфер.

```
/**
 * Инициировать начало транзакционных изменений
 * в файловую систему
 * возвращаемое значение: errOk, остальное воспринимается как ошибка
 */
ResultCls_t Trn_Begin(void)
```

Синтаксис команды представлен в таблице 6.

Таблица 6. Структура команды

Поле	**Значение**	**Описание**
CLA	'80'	
INS	‘F6’	
P1	‘06’	Открытие транзакции
P2	‘00’	
L_C	‘00’	

Смарт карта возвращает последовательность элементов следующего вида, приставленных в таблице 7.

Таблица 7. Ответные данные

Длина	**Значение**
2	Статус выполнения функции

2.3.5. Функция завершения транзакции изменения NVM

Функция завершает транзакцию, очищается транзакционный буфер.

```
/**
 * Зафиксировать изменения в файловой системе
 * возвращаемое значение: errOk, остальное воспринимается как ошибка
 */
ResultCls_t Trn_Commit(void)
```

Синтаксис команды представлен в таблице 8.

Таблица 8. Структура команды

Поле	**Значение**	**Описание**
CLA	'80'	
INS	‘F6’	
P1	‘07’	Закрытие транзакции (фиксация изменений)
P2	‘00’	
L_C	‘00’	

Смарт карта возвращает последовательность элементов следующего вида, приставленных в таблице 9.

Таблица 9. Ответные данные

Длина	**Значение**
2	Статус выполнения функции

2.3.6. Функция отмены транзакции изменения NVM

Функция отменяет изменения данных. Данные из транзакционного буфера восстанавливаются.

```
/**
 * Отменить предыдущие изменения в файловой системе
 * возвращаемое значение: errOk, остальное воспринимается как ошибка
 */
ResultCls_t Trn_Rollback (void)
```

Синтаксис команды представлен в таблице 10.

Таблица 10. Структура команды

Поле	**Значение**	**Описание**
CLA	'80'	
INS	‘F6’	
P1	‘08’	Отмена транзакции (откат изменений)

Поле	Значение	Описание
P2	‘00’	
L_C	‘00’	

Смарт карта возвращает последовательность элементов следующего вида, приставленных в таблице 11.

Таблица 11. Ответные данные

Длина	Значение
2	Статус выполнения функции

2.3.7. Функция определения размера блока данных (файла)

Определение размер файла по заданному дескриптору.

```
/**
 * Определение размера блока
 * hFile: дексриптор удаляемого файла
* возвращаемое значение: размер файла с указанным дескриптором
* возвращаемое значение, равное «-1» обозначает ошибку
 */
long  Mbm_GetFileSize ( MbmHandler_t  hFile)
```

Синтаксис команды представлен в таблице 12.

Таблица 12. Структура команды

Поле	Значение	Описание
CLA	'80'	
INS	‘F6’	
P1	‘05’	Определение размера блока данных (файла)
P2	‘00’	
L_C	‘02’	
Handle1	‘AA’	Старший байт дескриптора файла
Handle2	‘BB’	Младший байт дескриптора файла

Смарт карта возвращает последовательность элементов следующего вида, приставленных в таблице 13.

Таблица 13. Ответные данные

Длина	**Значение**
2	Статус выполнения функции
Size1	Старший байт размера блока (файла)
Size2	Младший байт размера блока (файла)

2.3.8. Функция удаления блока данных (файла)

Удаление файла заданного дескриптора. Возвращаемое значение – это статус выполнения операции.

```
/**
 * Удаление файла
 * hFile: дексриптор удаляемого файла
* возвращаемое значение: errOk, остальное воспринимается как ошибка
 */
ResultCls_t Mbm_Free(
                MbmHandler_t  hFile
                              )
```

Синтаксис команды представлен в таблице 14.

Таблица 14. Структура команды

Поле	**Значение**	**Описание**
CLA	'80'	
INS	‘F6’	
P1	‘02’	Удаление блока данных (файла)
P2	‘00’	
L_C	‘02’	
Handle1	‘AA’	Старший байт дескриптора файла
Handle2	‘BB’	Младший байт дескриптора файла

Смарт карта возвращает последовательность элементов следующего вида, приставленных в таблице 15.

Таблица 15. Ответные данные

Длина	**Значение**
2	Статус выполнения функции

2.4. Описание команд загрузки модуля в составе ОС на МК

2.4.1. Функция запроса случайного числа МК, MUTUAL CHALLENGE

Синтаксис команды представлен в таблице 16.

Таблица 16. Структура команды

Поле	**Значение**	**Описание**
CLA	'C0'	
INS	‘C3’	
P1	‘00’	
P2	‘00’	
L_C	‘06’	Длина данных запроса случайного числа МК
Data	‘00 53 00 02 00 81 00’	Данные для запроса случайного числа

Смарт карта возвращает последовательность элементов следующего вида, приставленных в таблице 17.

Таблица 17. Ответные данные

Длина	**Значение**
90 53 00 10 xx..xx	xx..xx – случайное число, 16 байт

2.4.2. Функция аутентификации с МК, MUTUAL AUTHENTICTION

Синтаксис команды представлен в таблице 18.

Таблица 18. Структура команды

Поле	**Значение**	**Описание**
CLA	'C0'	
INS	‘C3’	

Поле	Значение	Описание
P1	‘00’	
P2	‘00’	
L_C	‘64’	Длина данных аутентификации
Data	64 00 54 00 60 Token AB 00	Token AB – данных для аутентификации для МК, 96 байт

Смарт карта возвращает последовательность элементов следующего вида, приставленных в таблице 19.

Таблица 19. Ответные данные

Длина	Значение
90 54 00 50 Token BA	Token BA – данных для аутентификации от МК, 96 байт

2.4.3. Функция загрузки микропрограммы МК FLASH LOAD DATA

Синтаксис команды представлен в таблице 20.

Таблица 20. Структура команды

Поле	Значение	Описание
CLA	'C0'	
INS	‘C3’	
P1	‘00’	
P2	‘00’	
L_C	‘XX’	Длина данных микропрограммы МК
Data	‘AA’	Данные микропрограммы МК

3. УСЛОВИЯ ВЫПОЛНЕНИЯ ПРОГРАММЫ

Модуль имеет архитектуру типа монолит и представляет собой подготовленные для подключения к основному проекту ОС смарт-карт файлы исходного кода c,h.

Загрузка модуля во встроенную память микропроцессорной карты в составе ОС смарт-карт осуществляется с помощью APDU-команд согласно протоколу по ГОСТ Р ИСО/МЭК 7816, либо ГОСТ Р ИСО/МЭК 14443.

3.1. Минимальный состав программно-аппаратных средств

Программно-аппаратные требования смарт карт представлены в таблице 1.

Таблица 1 – Программные, аппаратные требования смарт карт

Наименование требования	Значение
Среда разработки	
Среда разработки	Eclipse IDE
Компилятор	GNU ARM Embedded Toolchain
Язык разработки	C
Аппаратные требования	
Семейство микроконтроллеров	Infineon SLC36PD, Samsung S3D350A и другие
Криптографический сопроцессор	SCP
Безопасность	В составе ОС смарт-карт
Объем ПЗУ \Flash	20 KB
Объем ОЗУ\RAM	6 KB
Адресуемое пространство долговременной памяти	
Объем ПЗУ(NVM)	65KB
Объем Журнала транзакций	4KB
Объем блока данных ПЗУ(NVM)	500B-65KB
Контроль целостности передачи данных	
CRC16/32	2-4 байт

3.2. Требования к составу периферийных устройств

Включая, но не ограничиваясь, семейства МК Infineon SLC36PD и Samsung S3D350A.

Устройства чтения/записи смарт-карт, например, PC\SC кард-ридеры.

3.3. Требования к персоналу (оператору)

Пользователь должен обладать практическими навыками владения персонального ПК, иметь знания в области смарт-карт, обладать практическими навыками системного программирования для МК.

4. ПРИМЕР РАБОТЫ С ПРИЛОЖЕНИЕМ

4.1. Загрузка модуля

Ниже представлен пример загрузки модуля на МК в составе ОС смарт-карт, соответствующей требованиям раздела 3.1:

МК	APDU интерфейс	Управлявшее прикладное ПО и устройство работы со смарт-картами	Комментарий
	← Reset		
	ATR bootloader [3B 10 96/97]		
	1. Mutual Authentication		
	← GET ITEM Kc LABEL [c0 c3 00 00 08 00 50 00 04 00 80 00 00 00]		
	90 80 00 02 AA BB + SW 9000 →		
	← GET ITEM Kd LABEL [c0 c3 00 00 08 00 50 00 04 00 81 00 00 00]		
	90 81 00 02 CC DD+ SW 9000 →		
	← GET ITEM Kfdi LABEL [c0 c3 00 00 08 00 50 00 04 00 82 00 00 00]		
	90 82 00 02 EE FF+ SW 9000 →		
	← MUTUAL CHALLENGE [c0 c3 00 00 06 00 53 00 02 00 81 00]		
	90 53 00 10 fd 5e ef e0 b3 c1 24 8f 7c 6a a1 54 a0 7b ff 75 + SW 9000 →		
	← MUTUAL AUTHENTICTION [c0 c3 00 00 64 00 54 00 60 Token AB 00]		
	90 54 00 50 Token BA+ SW 9000 →		

МК	APDU интерфейс	Управлявшее прикладное ПО и устройство работы со смарт-картами	Комментарий
	←INVALIDATE USER NVM [c0 c3 00 00 04 00 60 00 00]		
	SW 9000 →		
	← FLASH LOAD DATA [c0 c3 00 00 xx +Data]		
	SW 9000 →		
	….		
	← SET STARUP OS [c0 c3 00 00 06 00 61 00 02 00 01]		
	SW 9000 →		

4.2. Персонализация модуля

Ниже представлен пример персонализации (запись данных) с помощью модуля на МК в составе ОС смарт-карт, соответствующей требованиям раздела 3.1:

Модуль	APDU интерфейс	Управлявшее прикладное ПО и устройство работы со смарт-картами	Комментарий
	← Reset		
	ATR →		
	← [80F60100020010]		Создание файла
	002D0006 + SW 9000 →		Статус и дескриптор
	← [80F6040006000600000010]		Чтение файла
	DATA + SW 9000 →		
	← [80F606000]		Открытие транзакции
	DATA + SW 9000 →		Статус
	← [80F6030016000600000010112233445566778899 00AABBCCDDEEFF]		Запись данных в файл
	DATA + SW 9000 →		Статус

Модуль	APDU интерфейс	Управлявшее прикладное ПО и устройство работы со смарт-картами	Комментарий
	← [80F60700]		Закрытие транзакции
	DATA + SW 9000 →		Статус
	← [80F6040006000600000010]		Чтение файла
	002D11223344556677889900AABBCCDDEEFF + SW 9000 →		Данные файла

4.3. Чтение данных модуля

Ниже представлен пример чтения данных из модуля, загруженного на МК в составе ОС смарт-карт, соответствующей требованиям раздела 3.1:

Модуль	APDU интерфейс	Управлявшее прикладное ПО и устройство работы со смарт-картами	Комментарий
	← Reset		
	ATR →		
	← [80F6040006000600000010]		Чтение файла
	DATA + SW 9000 →		Данные файла