НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационная технология

Карты идентификационные

КАРТЫ НА ИНТЕГРАЛЬНЫХ СХЕМАХ С КОНТАКТАМИ

Часть 4

Межотраслевые команды для обмена

Издание официальное

БЗ 1—2004/281

ГОССТАНДАРТ РОССИИ
Москва

gost.e-ecolog.ru

Предисловие

1 РАЗРАБОТАН Техническим комитетом по стандартизации ТК 22 «Информационные технологии», Федеральным государственным унитарным предприятием «Всероссийский научно-исследовательский институт стандартизации и сертификации в машиностроении» (ВНИИНМАШ), ОАО «Московский комитет по науке и технологиям»

ВНЕСЕН ТК 22 «Информационные технологии»

2 ПРИНЯТ И ВВЕДЕН В ДЕЙСТВИЕ Постановлением Госстандарта России от 9 марта 2004 г. № 116-ст

3 Настоящий стандарт представляет собой аутентичный текст международного стандарта ИСО/МЭК 7816-4:1995 «Информационная технология. Карты идентификационные. Карты на интегральной(ых) схеме(ах) с контактами. Часть 4. Межотраслевые команды для обмена» с Изменением № 1 (1997 г.)

4 ВВЕДЕН ВПЕРВЫЕ

© ИПК Издательство стандартов, 2004

Настоящий стандарт не может быть полностью или частично воспроизведен, тиражирован и распространен в качестве официального издания без разрешения Госстандарта России

Содержание

1 Область применения . . . 1
2 Нормативные ссылки . . . 1
3 Определения . . . 2
4 Сокращения и обозначения . . . 3
5 Основные организационные структуры . . . 3
5.1 Структуры данных . . . 3
5.2 Архитектура безопасности карты . . . 8
5.3 Структуры APDU-сообщений . . . 10
5.4 Соглашения по кодированию заголовков команд, полей данных и завершителей ответа . . . 13
5.5 Логические каналы . . . 18
5.6 Безопасный обмен сообщениями . . . 19
6 Основные межотраслевые команды . . . 26
6.1 Команда СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ . . . 26
6.2 Команда ВВЕСТИ ДВОИЧНОЕ ЗНАЧЕНИЕ . . . 27
6.3 Команда ОБНОВИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ . . . 28
6.4 Команда ИСКЛЮЧИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ . . . 29
6.5 Команда СЧИТАТЬ ЗАПИСЬ(И) . . . 31
6.6 Команда ВВЕСТИ ЗАПИСЬ . . . 33
6.7 Команда ПРИСОЕДИНИТЬ ЗАПИСЬ . . . 34
6.8 Команда ОБНОВИТЬ ЗАПИСЬ . . . 36
6.9 Команда ИЗВЛЕЧЬ ДАННЫЕ . . . 37
6.10 Команда ПОМЕСТИТЬ ДАННЫЕ . . . 39
6.11 Команда ВЫБРАТЬ ФАЙЛ . . . 40
6.12 Команда ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ . . . 42
6.13 Команда ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ . . . 44
6.14 Команда ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ . . . 45
6.15 Команда СОЗДАТЬ ЗАДАЧУ . . . 46
6.16 Команда ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ . . . 47
7 Межотраслевые команды, ориентированные на передачу данных . . . 48
7.1 Команда ИЗВЛЕЧЬ ОТВЕТ . . . 48
7.2 Команда КОНВЕРТ . . . 49
8 Байты предыстории . . . 49
9 Услуги карты, не зависимые от приложений . . . 54
Приложение А Транспортировка APDU-сообщений при помощи протокола передачи, обозначаемого T=0 . . . 57
Приложение Б Транспортировка APDU-сообщений при помощи протокола передачи, обозначаемого T=1 . . . 62
Приложение В Управление указателя записи . . . 64
Приложение Г Использование базовых правил кодирования АСН.1 . . . 65
Приложение Д Примеры профилей карт . . . 66
Приложение Е Использование безопасного обмена сообщениями . . . 68

Введение

Настоящий стандарт — один из серии стандартов, описывающих параметры карт на интегральных схемах с контактами и их применение в рамках обмена информацией.

Данные карты представляют собой идентификационные карты, предназначенные для обмена информацией, основанного на согласованиях между внешним источником и интегральной схемой карты. В результате такого обмена карта поставляет информацию (результаты вычислений, хранимые данные) и (или) изменяет свое содержимое (память данных, память событий).

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационная технология
Карты идентификационные

КАРТЫ НА ИНТЕГРАЛЬНЫХ СХЕМАХ С КОНТАКТАМИ

Часть 4
Межотраслевые команды для обмена

Information technology. Identification cards. Integrated circuit(s) cards with contacts
Part 4. Interindustry commands for interchange

Дата введения 2005—01—01

1 Область применения

Настоящий стандарт устанавливает:

- содержание сообщений (команд и ответов), передаваемых устройством сопряжения карте и обратно;
- структуру и содержание байтов предыстории, посылаемых картой во время ответа на восстановление;
- структуру файлов и данных, прослеживаемую на стыке между картой и устройством сопряжения при обработке межотраслевых команд;
- методы доступа к файлам и данным, содержащимся в карте;
- архитектуру безопасности, определяющую права доступа к файлам и данным, содержащимся в карте;
- методы безопасного обмена сообщениями;
- методы доступа к алгоритмам, обрабатываемым картой (исключая описание самих алгоритмов).

Стандарт не распространяется на реализацию обмена данными внутри карты и/или внешнего окружения.

Стандарт допускает дальнейшую стандартизацию дополнительных межотраслевых команд и архитектур безопасности.

2 Нормативные ссылки

В настоящем стандарте использованы ссылки на следующие стандарты и классификатор:

ГОСТ Р ИСО/МЭК 7816-6—2003 Карты идентификационные. Карты на интегральных схемах с контактами. Часть 6. Элементы данных для межотраслевого обмена

ГОСТ Р ИСО/МЭК 8825—93 Информационная технология. Взаимосвязь открытых систем. Спецификация базовых правил кодирования для абстрактно-синтаксической нотации версии 1 (АСН.1)

ГОСТ Р ИСО/МЭК 10116—93 Информационная технология. Режимы работы для алгоритма *n*-разрядного блочного шифрования

ИСО/МЭК 7812-1:2000* Карты идентификационные. Идентификация эмитентов. Часть 1. Система нумерации

ИСО/МЭК 7816-3:1997* Информационная технология. Карты идентификационные. Карты на интегральной(ых) схеме(ах) с контактами. Часть 3. Электронные сигналы и протоколы передачи

ИСО/МЭК 7816-5:1994* Карты идентификационные. Карты на интегральной(ых) схеме(ах) с контактами. Часть 5. Система нумерации и процедура регистрации для идентификаторов приложений

---

* Международные стандарты ИСО/МЭК — во ВНИИКИ Госстандарта России.

Издание официальное

ИСО/МЭК 9796-2:1997* Информационная технология. Методы защиты. Схемы цифровой подписи, обеспечивающие восстановление сообщения. Часть 2. Механизмы, использующие хэш-функцию

ИСО/МЭК 9796-3:2000* Информационная технология. Методы защиты. Схемы цифровой подписи, обеспечивающие восстановление сообщения. Часть 3. Механизмы, основанные на дискретном логарифме

ИСО/МЭК 9797:1994* Информационная технология. Методы защиты. Механизм обеспечения целостности данных с применением криптографической контрольной функции, использующей алгоритм блочного шифрования

ИСО/МЭК 9979:1999* Информационная технология. Методы защиты. Процедуры регистрации криптографических алгоритмов

ИСО/МЭК 10118-1:2000* Информационная технология. Методы защиты. Хэш-функции. Часть 1. Общие положения

ИСО/МЭК 10118-2:2000* Информационная технология. Методы защиты. Хэш-функции. Часть 2. Хэш-функции, использующие *n*-битовое блочное шифрование

ОК (МК (ИСО 3166) 004—97) 025—2001 Общероссийский классификатор стран мира

3 Определения

В настоящем стандарте используют следующие определения.

3.1 **файл ответа на восстановление:** Элементарный файл, который указывает рабочие характеристики карты.

3.2 **пара команда-ответ:** Набор из двух сообщений: команды и следующего за ней ответа.

3.3 **единица данных:** Наименьший набор битов, на который можно дать однозначную ссылку.

3.4 **элемент данных:** Смысловой элемент информации, прослеживаемый на стыке между картой и устройством сопряжения, для которого определены наименование, описание логического содержания, формат и кодирование.

3.5 **информационный объект:** Информация, прослеживаемая на стыке между картой и устройством сопряжения, состоящая из тега, длины и значения (т. е. элемента данных). В настоящем стандарте информационные объекты именуются как информационные объекты BER-TLV, COMPACT-TLV и SIMPLE-TLV.

3.6 **назначенный файл:** Файл, содержащий контрольную информацию файла и, как возможность, свободную память для распределения. Он может быть родительским файлом элементарных (EF) и/или назначенных (DF) файлов.

3.7 **имя DF:** Строка байтов, которая уникальным образом идентифицирует назначенный файл в карте.

3.8 **справочный файл:** Элементарный файл, определяемый в ИСО/МЭК 7816-5.

3.9 **элементарный файл:** Набор единиц данных или записей, которые совместно используют один и тот же идентификатор файла. Элементарный файл не может быть родительским для другого файла.

3.10 **контрольные параметры файла:** Логические, структурные атрибуты и атрибуты секретности файла.

3.11 **идентификатор файла:** Двухбайтовое двоичное значение, используемое для обращения к файлу.

3.12 **данные управления файлом:** Любая информация о файле, за исключением контрольных параметров файла (например, дата истечения срока действия, метка приложения).

3.13 **внутренний элементарный файл:** Элементарный файл для хранения данных, интерпретируемых картой.

3.14 **главный файл:** Обязательный уникальный назначенный файл, представляющий собой корень файловой структуры.

3.15 **сообщение:** Строка байтов, передаваемая устройством сопряжения карте или наоборот, исключая знаки, ориентированные на управление передачей, как определено в ИСО/МЭК 7816-3.

3.16 **родительский файл:** Назначенный файл, непосредственно предшествующий данному файлу в пределах иерархии.

3.17 **пароль:** Данные, которые могут быть затребованы приложением от пользователя карты.

3.18 **путь:** Сцепление идентификаторов файлов без разграничения. Если путь начинается с идентификатора главного файла, то это абсолютный путь.

---

* Международные стандарты ИСО/МЭК — во ВНИИКИ Госстандарта России.

3.19 **провайдер:** Орган, имеющий или получивший право на создание назначенного файла в карте.

3.20 **запись:** Строка байтов, которая может обрабатываться картой как единое целое и на которую можно дать ссылку посредством номера или идентификатора записи.

3.21 **идентификатор записи:** Значение, связываемое с записью, которое может использоваться для ссылки на эту запись. Несколько записей могут иметь один и тот же идентификатор в пределах элементарного файла.

3.22 **номер записи:** порядковый номер, присваиваемый каждой записи, который однозначно идентифицирует запись в пределах элементарного файла.

3.23 **рабочий элементарный файл:** Элементарный файл для хранения данных, не интерпретируемых картой.

4 Сокращения и обозначения

В настоящем стандарте применяют следующие сокращения.

APDU — блок данных прикладного протокола (Application protocol data unit).

ATR — ответ на восстановление (Answer to reset).

BER — базовые правила кодирования абстрактно-синтаксической нотации версии 1 (АСН.1) (Basic encoding rules of ASN.1), см. приложение Г.

CLA — байт класса (Class byte).

DIR — справочный (Directory).

DF — назначенный файл (Dedicated file).

EF — элементарный файл (Elementary file).

FCI — контрольная информация файла (File control information).

FCP — контрольный параметр файла (File control parameter).

FMD — данные управления файлом (File management data).

INS — командный байт (Instruction byte).

MF — главный файл (Master file).

P1, P2 — байты параметров (Parameter bytes).

PTS — выбор типа протокола (Protocol type selection).

RFU — зарезервировано для будущего использования (Reserved for future use).

SM — безопасный обмен сообщениями (Secure messaging).

SW1, SW2 — байты состояния (Status bytes).

TLV — тег, длина, значение (Tag, length, value).

TPDU — блок данных протокола передачи (Transmission protocol data unit).

В настоящем стандарте применяют следующие обозначения.

'0'—'9' и 'A'—'F' — шестнадцать шестнадцатеричных цифр.

(B_1) — значение байта B_1.

$B_1 \| B_2$ — сцепление байтов B_1 (старший байт) и B_2 (младший байт).

($B_1 \| B_2$) — значение сцепления байтов B_1 и B_2.

— номер.

5 Основные организационные структуры

5.1 Структуры данных

Настоящий подраздел содержит информацию о логической структуре данных, прослеживаемой на стыке между картой и устройством сопряжения при обработке межотраслевых команд, используемых в информационном обмене. Размещение данных в физической памяти и структурная информация сверх той, что представлена в данном подразделе, находятся за пределами компетенции настоящего стандарта и стандартов серии ИСО/МЭК 7816.

5.1.1 Организация файлов

Настоящий стандарт поддерживает следующие две категории файлов:

- назначенный файл (DF);
- элементарный файл (EF).

Логическая организация данных в карте представляет собой следующую структурную иерархию назначенных файлов:

- DF в основании иерархии, называемый главным файлом (MF). MF является обязательным;
- прочие DF, являющиеся необязательными.

Определены следующие два типа файлов EF:

- внутренние, предназначаемые для хранения данных, интерпретируемых картой, т. е. данных, анализируемых и используемых картой в целях управления и контроля;
- рабочие, предназначаемые для хранения данных, не интерпретируемых картой, т. е. данных, подлежащих использованию исключительно внешним окружением.

На рисунке 1 показан пример логической организации файлов в карте.

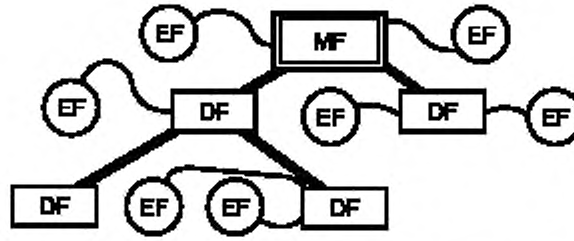


Рисунок 1 — Пример логической организации файлов

5.1.2 Методы обращения к файлу

В случаях, когда файл не может быть выбран неявно, должны быть предусмотрены возможности для его выбора при помощи, по меньшей мере, одного из следующих методов.

а) Обращение посредством идентификатора файла

Обращение к любому файлу может быть осуществлено при помощи идентификатора файла, кодируемого в двух байтах. Если через идентификатор файла осуществляется обращение к MF, то должно использоваться значение '3F00' (зарезервированное значение). Значение 'FFFF' зарезервировано для будущего использования. Значение '3FFF' также зарезервировано (см. перечисление б)). Для того чтобы однозначно выбирать любой файл при помощи его идентификатора, все файлы EF и DF, находящиеся в иерархии непосредственно под данным DF, должны иметь разные идентификаторы.

б) Обращение через путь

Обращение к любому файлу может быть осуществлено при помощи пути (сцепления идентификаторов файлов). Путь начинается с идентификатора MF или текущего DF и заканчивается идентификатором выбираемого файла. Между этими двумя идентификаторами путь состоит из идентификаторов последовательных (в рамках иерархии) родительских DF, если они имеются. Порядок следования идентификаторов файлов — всегда в направлении от родительского файла к дочернему. Если идентификатор текущего DF не известен, в начале пути может использоваться значение '3FFF' (зарезервированное значение). Использование пути позволяет осуществлять однозначный выбор любого файла из MF или из текущего DF.

в) Обращение посредством короткого идентификатора EF

Обращение к любому EF может быть осуществлено при помощи короткого идентификатора EF, кодируемого в пяти битах, представляющих значение от 1 до 30. Значение 0, используемое в качестве короткого идентификатора EF, указывает на выбираемый в текущий момент EF. Короткие идентификаторы файлов EF не могут быть использованы в последовательности пути или в качестве идентификатора файла (например, в команде ВЫБРАТЬ ФАЙЛ).

г) Обращение посредством имени DF

Обращение к любому DF может быть осуществлено по имени DF, кодируемому в 1—16 байтах. Для того чтобы однозначно выбирать по имени DF (например, когда выбор осуществляется путем использования идентификаторов приложений, как определено в ИСО/МЭК 7816-5), имя каждого DF не должно повторяться в данной карте.

5.1.3 Структуры элементарных файлов

Определены следующие структуры файлов EF:

- прозрачная структура, при которой EF прослеживается на стыке между картой и устройством сопряжения как последовательность единиц данных;
- структура из записей, при которой EF прослеживается на стыке между картой и устройством сопряжения как последовательность отдельно идентифицируемых записей.

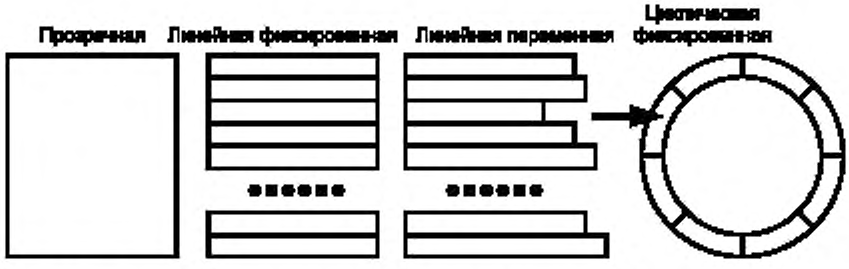
Для файлов EF со структурой из записей определены следующие атрибуты:

- размер записей — либо фиксированный, либо переменный;
- способ организации записей: либо в виде последовательного ряда (линейная структура), либо в виде кольца (циклическая структура).

Карта должна поддерживать по меньшей мере один из следующих четырех методов структурирования файлов EF:

- «прозрачный EF»;
- «линейный EF с записями фиксированного размера»;
- «линейный EF с записями переменного размера»;
- «циклический EF с записями фиксированного размера».

На рисунке 2 представлены четыре структуры файлов EF, соответствующие четырем методам структурирования.



Примечание Стрелка указывает на последнюю сделанную запись.

Рисунок 2 — Структуры файлов EF

5.1.4 Методы обращения к данным

Обращение к данным может осуществляться как к записям, единицам данных или информационным объектам. Предполагается, что данные должны храниться в виде одной непрерывной последовательности записей (в пределах EF со структурой из записей) или единиц данных (в пределах EF с прозрачной структурой). Обращение к записи или единице данных вне EF является ошибкой.

Метод обращения к данным, метод нумерации записей и размер единиц данных являются характеристиками, зависящими от EF. Карта может давать соответствующие указания в ATR, файле ATR и контрольной информации любого файла. Если карта дает указания в нескольких местах, то действительным для данного EF является ближайшее к нему указание в пределах пути от MF до этого EF.

5.1.4.1 Обращение к записи

В пределах каждого EF со структурой из записей обращение к каждой записи может осуществляться при помощи идентификатора записи и/или номера записи. Идентификаторы и номера записей представляют собой целые восьмибитовые числа без знака со значениями от '01' до 'FE'. Значение '00' зарезервировано для особых целей. Значение 'FF' является RFU.

Обращение посредством идентификатора записи должно приводить в действие управление указателя записи. Процедура восстановления карты, команда ВЫБРАТЬ ФАЙЛ и любая команда, несущая разрешенный короткий идентификатор EF, могут воздействовать на указатель записи. Обращение посредством номера записи не должно воздействовать на указатель записи.

а) Обращение посредством идентификатора записи

Каждый идентификатор записи предоставляется приложением. Если запись представляет собой информационный объект SIMPLE-TLV в поле данных сообщения (см. 5.4.4), то идентификатором записи является первый байт этого информационного объекта. В пределах EF со структурой из записей записи могут иметь один и тот же идентификатор, тогда данные, содержащиеся в записях, могут использоваться для их различения.

Всякий раз, когда обращение осуществляется с идентификатором записи, должна быть указана логическая позиция целевой записи: первое или последнее вхождение записи, следующее или предыдущее вхождение по отношению к указателю записи.

В пределах каждого EF с линейной структурой логические позиции должны последовательно

присваиваться при осуществлении операции записи или операции присоединения записи, т.е. в порядке создания. Поэтому первая созданная запись будет находиться в первой логической позиции.

В пределах каждого EF с циклической структурой логические позиции должны последовательно присваиваться в обратном порядке, т. е. в первой логической позиции будет находиться последняя созданная запись.

Как для линейных, так и для циклических структур установлены следующие дополнительные правила:

- первым вхождением должна быть запись с заданным идентификатором и в первой логической позиции; последним вхождением должна быть запись с заданным идентификатором и в последней логической позиции;
- когда текущая запись отсутствует, следующее вхождение должно быть эквивалентно первому вхождению; предыдущее вхождение должно быть эквивалентно последнему вхождению;
- когда текущая запись имеется, следующим вхождением должна быть ближайшая запись с заданным идентификатором, но в восходящей логической позиции по отношению к текущей записи; предыдущим вхождением должна быть ближайшая запись с заданным идентификатором, но в нисходящей логической позиции по отношению к текущей записи;
- значение '00' должно относиться к первой, последней, следующей или предыдущей записи в порядке нумерации, независимо от идентификатора записи.

б) Обращение посредством номера записи

В пределах каждого EF со структурой из записей номера записей являются уникальными и последовательными.

В пределах каждого EF с линейной структурой номера записей должны последовательно присваиваться при осуществлении операции записи или операции присоединения записи, т. е. в порядке создания. Поэтому первая запись (запись номер один, запись # 1) будет представлять собой первую созданную запись.

В пределах каждого EF с циклической структурой номера записей должны последовательно присваиваться в обратном порядке, т. е. первая запись (запись номер один, запись # 1) будет представлять собой последнюю созданную запись.

Как для линейных, так и для циклических структур установлено следующее дополнительное правило: значение '00' должно относиться к текущей записи, т. е. к записи, зафиксированной указателем записи.

5.1.4.2 Обращение к единице данных

В пределах каждого EF с прозрачной структурой обращение к любой единице данных может осуществляться при помощи смещения (например, в команде СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ, см. 6.1), представляющего собой целое число без знака, ограниченное восемью либо 15 битами, что определяется опцией в соответствующей команде. Для первой единицы данных файла EF смещению присваивают значение 0, для каждой последующей единицы данных смещение увеличивается на единицу.

По умолчанию, т.е. в том случае, если карта не дает никакого указания, размер единицы данных составляет один байт.

Примечания

1 EF со структурой из записей может поддерживать обращение к единице данных, и в этом случае единицы данных могут содержать наряду с данными еще и структурную информацию, например номера записей в линейной структуре.

2 В пределах EF со структурой из записей обращение к единице данных может не обеспечивать ожидаемый результат, поскольку порядок хранения записей в EF неизвестен (записи, например, могут храниться в циклической структуре).

5.1.4.3 Обращение к информационному объекту

Каждый информационный объект, определяемый в 5.4.4, начинается с тега, который служит для обращения к информационному объекту. Теги установлены в настоящем стандарте и других стандартах серии ГОСТ Р ИСО/МЭК 7816 (ИСО/МЭК 7816).

5.1.5 Контрольная информация файла

Контрольная информация файла (FCI) представляет собой строку байтов данных, содержащуюся в ответе на команду ВЫБРАТЬ ФАЙЛ. Контрольная информация может быть у любого файла.

В таблице 1 приведены три шаблона, предназначаемые для передачи контрольной информации файла в том случае, когда она кодируется в виде информационных объектов BER-TLV.

Таблица 1 — Шаблоны для FCI

Тег	Значение
'62'	Контрольные параметры файла (шаблон FCP)
'64'	Данные управления файлом (шаблон FMD)
'6F'	Контрольная информация файла (шаблон FCI)

Шаблон FCP предназначается для передачи контрольных параметров файла (FCP), т. е. любых информационных объектов BER-TLV, определяемых в таблице 2.

Шаблон FMD предназначается для передачи данных управления файлом (FMD), т. е. информационных объектов BER-TLV, указанных в других разделах настоящего стандарта или в других стандартах серии ГОСТ Р ИСО/МЭК 7816 (ИСО/МЭК 7816) (например, метки приложения по ИСО/МЭК 7816-5 и даты истечения срока действия приложения по ГОСТ Р ИСО/МЭК 7816-6).

Шаблон FCI предназначается для передачи контрольных параметров файла и данных управления файлом.

Извлечение этих трех шаблонов может осуществляться по опциям выбора команды ВЫБРАТЬ ФАЙЛ (см. таблицу 59). Если установлена опция FCP или FMD, то использование соответствующего шаблона является обязательным. Если установлена опция FCI, то использование шаблона FCI не является обязательным.

Часть контрольной информации файла может быть дополнительно представлена в рабочем файле EF под управлением приложения, ссылка на который приводится под тегом '87'. В таком EF использование шаблона FCP или FCI для кодирования контрольной информации файла является обязательным.

Контрольная информация файла, не закодированная в соответствии с настоящим стандартом, может вводиться следующим образом:

тег, равный '00', или тег с любым значением выше чем '9F' — кодирование последующей строки байтов является оригинальным;

тег, равный '53' — поле значения информационного объекта состоит из произвольных данных, не закодированных в структуре TLV;

тег, равный '73' — поле значения информационного объекта состоит из произвольных информационных объектов BER-TLV.

Таблица 2 — Контрольные параметры файла

Тег	L	Значение	Применимость
'80'	2	Число байтов данных в файле, исключая структурную информацию	Прозрачный EF
'81'	2	Число байтов данных в файле, включая структурную информацию, если она имеется	Любой файл
'82'	1	Байт описателя файла (см. таблицу 3)	Любой файл
	2	Байт описателя файла, сопровождаемый байтом кодирования данных (см. таблицу 86)	То же
	3 или 4	Байт описателя файла, сопровождаемый байтом кодирования данных и максимальной длиной записи	EF со структурой из записей
'83'	2	Идентификатор файла	Любой файл
'84'	От 1 до 16	Имя DF	DF
'85'	Переменная	Оригинальная информация	Любой файл
'86'	Переменная	Атрибуты секретности (кодирование за пределами компетенции настоящего стандарта)	Любой файл
'87'	2	Идентификатор файла EF, содержащего расширение FCI	Любой файл
От '88' до '9E'		RFU	
'9FXY'		RFU	

Таблица 3 — Байт описателя файла

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
0	×	—	—	—	—	—	—	Доступность файла:
0	0	—	—	—	—	—	—	- файл несовместного доступа
0	1	—	—	—	—	—	—	- файл совместного доступа
0	—	×	×	×	—	—	—	Тип файла:
0	—	0	0	0	—	—	—	- рабочий EF
0	—	0	0	1	—	—	—	- внутренний EF
0	—	0	1	0	—	—	—	- зарезервировано
0	—	0	1	1	—	—	—	для
0	—	1	0	0	—	—	—	оригинальных
0	—	1	0	1	—	—	—	типов
0	—	1	1	0	—	—	—	файлов EF
0	—	1	1	1	—	—	—	- DF
0	—	—	—	—	×	×	×	Структура EF:
0	—	—	—	—	0	0	0	- информация не предоставлена
0	—	—	—	—	0	0	1	- прозрачная
0	—	—	—	—	0	1	0	- линейная фиксированная; нет дополнительной информации
0	—	—	—	—	0	1	1	- линейная фиксированная; информационные объекты SIMPLE-TLV
0	—	—	—	—	1	0	0	- линейная переменная; нет дополнительной информации
0	—	—	—	—	1	0	1	- линейная переменная; информационные объекты SIMPLE-TLV
0	—	—	—	—	1	1	0	- циклическая; нет дополнительной информации
0	—	—	—	—	1	1	1	- циклическая; информационные объекты SIMPLE-TLV
1	×	×	×	×	×	×	×	RFU

Примечание — «Совместный доступ» означает, что файл поддерживает по меньшей мере одновременный доступ по разным логическим каналам.

5.2 Архитектура безопасности карты

В настоящем подразделе рассмотрены следующие аспекты:
- состояние защиты,
- атрибуты секретности,
- механизмы защиты.

Атрибуты секретности сравниваются с состоянием защиты для выполнения команд и/или получения доступа к файлам.

5.2.1 Состояние защиты

Состояние защиты представляет собой текущее состояние, которое может достигаться после завершения:

- ответа на восстановление (ATR) и возможного выбора типа протокола (PTS) и/или
- отдельной команды или последовательности команд, возможно выполняющих процедуры аутентификации.

Состояние защиты может являться также результатом завершения процедуры защиты, связанной с идентификацией участвующих сторон, если таковая применяется, например посредством:

- проверки знания пароля (например, с использованием команды ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ),

- проверки знания ключа (например, с использованием команды СОЗДАТЬ ЗАДАЧУ, сопровождаемой командой ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ),

- безопасного обмена сообщениями (например, на основе аутентификации сообщений).

Рассматриваются следующие три состояния защиты.

а) Глобальное состояние защиты

Глобальное состояние защиты может видоизменяться в результате завершения процедуры аутентификации, относящейся к MF (например, аутентификации участвующей стороны по паролю или ключу, присоединенному к MF).

б) Файловое состояние защиты (т.е. ориентированное на файл)

Файловое состояние защиты может видоизменяться в результате завершения процедуры аутентификации, относящейся к DF (например, аутентификации участвующей стороны по паролю или ключу, присоединенному к конкретному DF). Оно может быть сохранено, восстановлено или утрачено при выборе файла (см. 6.10.2). Данная модификация состояния защиты может быть уместна только для приложения, с которым связана процедура аутентификации.

в) Командное состояние защиты (т.е. ориентированное на команду)

Командное состояние защиты имеет место лишь во время выполнения команды, предусматривающей аутентификацию с использованием безопасного обмена сообщениями (см. 5.6); такая команда может оставлять без изменений другое состояние защиты.

Если применяется концепция логических каналов, файловое состояние защиты может зависеть от логического канала (см. 5.5.1).

5.2.2 Атрибуты секретности

Атрибуты секретности, если они имеются, определяют разрешенные действия, а также процедуры, подлежащие выполнению для завершения таких действий.

Атрибуты секретности могут быть связаны с каждым файлом и фиксируют условия секретности, которые необходимо соблюсти для осуществления операций на файле. Атрибуты секретности файла зависят от:

- его категории (DF или EF),

- возможных параметров в его контрольной информации и/или в контрольной информации его родительского(их) файла(ов).

Примечание — Атрибуты секретности могут сопутствовать и другим объектам (например ключам).

5.2.3 Механизмы защиты

Настоящий стандарт устанавливает следующие механизмы защиты.

а) Аутентификация участвующей стороны по паролю

Карта сравнивает данные, полученные от внешнего окружения, с секретными внутренними данными. Этот механизм может использоваться для защиты прав пользователя.

б) Аутентификация участвующей стороны по ключу

Участвующая сторона, подвергаемая аутентификации, должна доказать знание соответствующего ключа в ходе процедуры аутентификации (например, с использованием команды СОЗДАТЬ ЗАДАЧУ, сопровождаемой командой ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ).

в) Аутентификация данных

Используя внутренние данные, секретные или открытые, карта проверяет избыточные данные, полученные от внешнего окружения. В свою очередь, используя секретные внутренние данные, карта вычисляет элемент данных (криптографическую контрольную сумму или электронную цифровую подпись) и вставляет его в данные, посылаемые внешнему окружению. Данный механизм может использоваться для защиты прав провайдера.

г) Шифрование данных

Используя секретные внутренние данные, карта осуществляет дешифрование криптограммы, полученной в поле данных. В свою очередь, используя внутренние данные, секретные или открытые, карта вычисляет криптограмму и вставляет ее в поле данных, возможно вместе с другими данными. Данный механизм может использоваться для обеспечения услуги конфиденциальности, например для управления ключами и условного доступа. В дополнение к механизму с криптограммой, конфиденциальность данных может достигаться за счет сокрытия данных. В этом случае карта вычисляет строку скрывающих байтов и прибавляет ее при помощи операции сложения «исключающее ИЛИ» к байтам данных, полученным от внешнего окружения или посылаемым внешнему

окружению. Данный механизм может использоваться для защиты личных секретных данных и для снижения возможностей фильтрации сообщений.

Результат аутентификации может регистрироваться во внутреннем EF в соответствии с требованиями приложения.

5.3 Структуры APDU-сообщений

Шаг в прикладном протоколе состоит из посылки команды, обработки ее принимающей стороной и посылки ответа в обратном направлении. Таким образом, конкретный ответ соответствует конкретной команде; вместе они именуются парой команда—ответ.

Блок данных прикладного протокола (APDU) содержит либо командное, либо ответное сообщение, посылаемое карте с устройства сопряжения, или наоборот.

В паре команда—ответ командное и ответное сообщения могут содержать данные, следствием чего являются четыре случая, обобщенные в таблице 4.

Таблица 4 — Данные в паре команда—ответ

Случай	Данные команды	Ожидаемые данные ответа
1	Нет данных	Нет данных
2	То же	Данные
3	Данные	Нет данных
4	»	Данные

5.3.1 Командный APDU

Представленный на рисунке 3 (см. также таблицу 6) командный APDU, определяемый в настоящем стандарте, состоит из:

- обязательного заголовка из четырех байтов (CLA, INS, P1, P2),
- условного тела переменной длины.

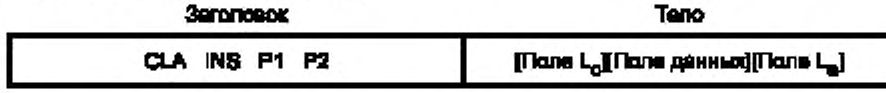


Рисунок 3 — Структура командного APDU

Число байтов, представленных в поле данных командного APDU, обозначается L_c.

Максимальное число байтов, ожидаемых в поле данных ответного APDU, обозначается L_e (длина ожидаемых данных). Если поле L_e содержит только нули, то запрашивается максимальное число имеющихся байтов данных.

На рисунке 4 представлены четыре структуры командных APDU, соответствующие четырем случаям из таблицы 4.



Рисунок 4 — Четыре структуры командных APDU

В случае 1 длина данных L_c является нулевой; следовательно поле L_c и поле данных являются пустыми. Длина данных L_e также является нулевой; следовательно и поле L_e является пустым. В результате тело является пустым.

В случае 2 длина данных L_c является нулевой; следовательно поле L_c и поле данных являются пустыми. Длина данных L_e не является нулевой; следовательно поле L_e присутствует. В результате тело состоит из поля L_e.

В случае 3 длина данных L_c не является нулевой; следовательно поле L_c присутствует, а поле данных состоит из L_c последующих байтов. Длина данных L_e является нулевой; следовательно поле L_e является пустым. В результате тело состоит из поля L_c, сопровождаемого полем данных.

В случае 4 длина данных L_c не является нулевой; следовательно поле L_c присутствует, а поле данных состоит из L_c последующих байтов. Длина данных L_e также не является нулевой; следовательно поле L_e также присутствует. В результате тело состоит из поля L_c, сопровождаемого полем данных и полем L_e.

5.3.2 Соглашения по декодированию тела команды

В случае 1 тело командного APDU является пустым. Такой командный APDU не несет поле длины.

В случаях 2—4 тело командного APDU состоит из строки, образованной L байтами, обозначаемыми символами B_1 ... B_L, как показано на рисунке 5. Такое тело несет одно или два поля длины; байт B_1 представляет собой первое поле длины или его часть.



Рисунок 5 — Непустое тело

В данных, представляющих функциональные возможности карты (см. 8.3.6), карта устанавливает, что в командном APDU поля L_c и L_e:

- либо должны быть короткими (один байт по умолчанию),
- либо могут быть расширенными (явное сообщение).

Следовательно, случаи 2—4 являются либо короткими (один байт для каждого поля длины), либо расширенными (байту B_1 присваивается значение '00', а значение каждой длины кодируется в двух других байтах).

В таблице 5 представлено декодирование командных APDU, соответствующих четырем случаям, приведенным в таблице 4 и на рисунке 4, и возможному расширению полей L_c и L_e. Любой другой командный APDU является недействительным.

Таблица 5 — Декодирование командных APDU

Условия			Случай
$L = 0$	—	—	1
$L = 1$	—	—	2, короткий (2K)
$L = 1+(B_1)$;	$(B_1) \neq 0$;	—	3, короткий (3K)
$L = 2+(B_1)$;	$(B_1) \neq 0$;	—	4, короткий (4K)
$L = 3$;	$(B_1) = 0$;		2, расширенный (2P)
$L = 3+(B_2 \parallel B_3)$;	$(B_1) = 0$;	$(B_2 \parallel B_3) \neq 0$	3, расширенный (3P)
$L = 5+(B_2 \parallel B_3)$;	$(B_1) = 0$;	$(B_2 \parallel B_3) \neq 0$	4, расширенный (4P)

Соглашения по декодированию поля L_e следующие.

Если значение L_e закодировано в одном (двух) байте(ах), где не все биты являются нулевыми, то оно равно значению байта(ов), которое находится в диапазоне от 1 до 255 (от 1 до 65535); нулевые значения всех битов означают максимальное значение L_e: 256 (65536).

Следующие первые четыре случая относятся ко всем картам.

Случай 1

$L = 0$; тело является пустым. Для этого случая:

- ни один байт не используется для L_c (L_c = 0);
- не представлен ни один байт данных;
- ни один байт не используется для L_e (L_e =0).

Случай 2K

L = 1. Для этого случая:
- ни один байт не используется для L_c (L_c = 0);
- не представлен ни один байт данных;
- байт B_1 кодирует значение L_e от 1 до 256.

Случай 3K

L = 1 + (B_1) и (B_1) ≠ 0 . Для этого случая:
- байт B_1 кодирует значение L_c (L_c ≠ 0) от 1 до 255;
- байты от B_2 до B_L представляют собой L_c байтов поля данных;
- ни один байт не используется для L_e (L_e =0).

Случай 4K

L = 2 + (B_1) и (B_1) ≠ 0. Для этого случая:
- байт B_1 кодирует значение L_c (L_c ≠ 0) от 1 до 255;
- байты от B_2 до B_{L-1} представляют собой L_c байтов поля данных;
- байт B_L кодирует значение L_e от 1 до 256.

Для карт, указывающих на расширение полей L_c и L_e (см. 8.3.6), применяются также следующие три случая.

Случай 2P

L = 3 и (B_1) = 0. Для этого случая:
- ни один байт не используется для L_c (L_c = 0);
- не представлен ни один байт данных;
- поле L_e состоит из трех байтов, где байты B_2 и B_3 кодируют значение L_e от 1 до 65536.

Случай 3P

L = 3 + (B_2 ‖ B_3), (B_1) = 0 и (B_2 ‖ B_3) ≠ 0. Для этого случая:
- поле L_c состоит из первых трех байтов, где байты B_2 и B_3 кодируют значение L_c (L_c ≠ 0) от 1 до 65535;
- байты от B_4 до B_L представляют собой L_c байтов поля данных;
- ни один байт не используется для L_e (L_e = 0).

Случай 4P

L = 5 + (B_2 ‖ B_3), (B_1) = 0 и (B_2 ‖ B_3) ≠ 0. Для этого случая:
- поле L_c состоит из первых трех байтов, где байты B_2 и B_3 кодируют значение L_c (L_c ≠ 0) от 1 до 65535;
- байты от B_4 до B_{L-2} представляют собой L_c байтов поля данных;
- поле L_e состоит из последних двух байтов B_{L-1} и B_L, кодирующих значение L_e от 1 до 65536.

Для каждого протокола передачи по ИСО/МЭК 7816-3 в настоящем стандарте предусмотрено приложение (см. приложения А и Б), определяющее транспортировку блоков данных APDU пары команда—ответ в каждом из семи рассмотренных случаев.

5.3.3 Ответный APDU

Представленный на рисунке 6 (см. также таблицу 7) ответный APDU, определяемый в настоящем стандарте, состоит из:
- условного тела переменной длины,
- обязательного завершителя из двух байтов (SW1, SW2).

Число байтов, представленных в поле данных ответного APDU, обозначается через L_r.

Тело	Завершитель
[Поле данных]	SW1 SW2

Рисунок 6 — Структура ответного APDU

Завершитель кодирует состояние принимающей стороны после обработки пары команда—ответ.

Примечание — Если команда прерывается, то ответный APDU представляет завершитель, кодирующий состояние ошибки в двух байтах состояния.

5.4 Соглашения по кодированию заголовков команд, полей данных и завершителей ответа

Таблица 6 представляет содержимое командного APDU.

Таблица 6 — Содержимое командного APDU

Код	Наименование	Длина	Описание
CLA	Класс	1	Класс команды
INS	Команда	1	Код команды
P1	Параметр 1	1	Параметр команды 1
P2	Параметр 2	1	Параметр команды 2
Поле L_c	Длина	Переменная: 1 или 3	Число байтов, представленных в поле данных команды
Поле данных	Данные	Переменная: равна L_c	Строка байтов, посылаемая в поле данных команды
Поле L_e	Длина	Переменная: ≤3	Максимальное число байтов, ожидаемых в поле данных ответа на команду

Таблица 7 представляет содержимое ответного APDU.

Таблица 7 — Содержимое ответного APDU

Код	Наименование	Длина	Описание
Поле данных	Данные	Переменная: равна L_r	Строка байтов, принимаемая в поле данных ответа
SW1	Байт состояния 1	1	Состояние обработки команды
SW2	Байт состояния 2	1	Квалификатор обработки команды

Последующие пункты устанавливают соглашения по кодированию байта класса, командного байта, байтов параметров, байтов поля данных и байтов состояния.

Если не указано иначе, то в этих байтах биты RFU кодируются нулем, а байты RFU кодируются значением '00'.

5.4.1 Байт класса

В соответствии с таблицами 8 и 9 байт класса CLA команды применяется для указания:

- степени соответствия команды и ответа настоящему стандарту;
- формата безопасного обмена сообщениями и номера логического канала (в случае применимости, см. таблицу 9).

Таблица 8 — Кодирование и смысловое содержание CLA

Значение	Смысловое содержание
'0X'	Структура и кодирование команды и ответа — в соответствии с настоящим стандартом (кодирование 'X' см. в таблице 9)
От '10' до '7F'	RFU
'8X', '9X'	Структура команды и ответа — в соответствии с настоящим стандартом. За исключением 'X' (кодируется по таблице 9), кодирование и смысловое содержание команды и ответа являются оригинальными
'AX'	Если не указано иначе контекстом приложения, структура и кодирование команды и ответа — в соответствии с настоящим стандартом (кодирование 'X' см. в таблице 9)
От 'B0' до 'CF'	Структура команды и ответа — в соответствии с настоящим стандартом
От 'D0' до 'FE'	Оригинальные структура и кодирование команды и ответа
'FF'	Зарезервировано для PTS

Таблица 9 — Кодирование и смысловое содержание полубайта 'X' в байте CLA, равном '0X', '8X', '9X' или 'AX'

b4	b3	b2	b1	Смысловое содержание
×	×	—	—	Формат безопасного обмена сообщениями (SM)
0	×	—	—	Никакого SM или SM, не соответствующий 5.6:
0	0	—	—	- никакого SM или SM не указан
0	1	—	—	- оригинальный формат SM
1	×	—	—	Безопасный обмен сообщениями в соответствии с 5.6:
1	0	—	—	- неаутентифицируемый заголовок команды
1	1	—	—	- аутентифицируемый заголовок команды (использование заголовка команды см. в 5.6.3.1)
—	—	×	×	Номер логического канала (в соответствии с 5.5)
				(b2b1 = 00, если логические каналы не используются или выбран логический канал # 0)

5.4.2 Командный байт

Командный байт INS следует кодировать, чтобы передача была возможна с любым из протоколов, определяемых в ИСО/МЭК 7816-3. В таблице 10 представлены коды INS (в виде последовательностей), являющиеся недействительными.

Таблица 10 — Недействительные коды INS

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
×	×	×	×	×	×	×	1	Нечетные значения
0	1	1	0	×	×	×	×	'6X'
1	0	0	1	×	×	×	×	'9X'

В таблице 11 представлены коды INS, определяемые в настоящем стандарте. Если значение CLA находится в диапазоне от '00' до '7F', другие значения кодов INS должны назначаться подкомитетом № 17 совместного технического комитета № 1 ИСО/МЭК (ПК 17 СТК 1 ИСО/МЭК).

Таблица 11 — Коды INS, определяемые в настоящем стандарте

Значение	Имя команды	Подраздел
'0E'	ИСКЛЮЧИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ	6.4
'20'	ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ	6.12
'70'	ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ	6.16
'82'	ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ	6.14
'84'	СОЗДАТЬ ЗАДАЧУ	6.15
'88'	ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ	6.13
'A4'	ВЫБРАТЬ ФАЙЛ	6.11
'B0'	СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ	6.1
'B2'	СЧИТАТЬ ЗАПИСЬ(И)	6.5
'C0'	ИЗВЛЕЧЬ ОТВЕТ	7.1
'C2'	КОНВЕРТ	7.2
'CA'	ИЗВЛЕЧЬ ДАННЫЕ	6.9
'D0'	ВВЕСТИ ДВОИЧНОЕ ЗНАЧЕНИЕ	6.2
'D2'	ВВЕСТИ ЗАПИСЬ	6.6
'D6'	ОБНОВИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ	6.3
'DA'	ПОМЕСТИТЬ ДАННЫЕ	6.10
'DC'	ОБНОВИТЬ ЗАПИСЬ	6.8
'E2'	ПРИСОЕДИНИТЬ ЗАПИСЬ	6.7

5.4.3 Байты параметров

Байты параметров P1, P2 команды могут иметь любое значение. Если байт параметра не обеспечивает никакого дальнейшего уточнения, то он должен быть установлен в состояние '00'.

5.4.4 Байты поля данных

Каждое поле данных должно иметь одну из следующих трех структур:

- каждое закодированное в структуре TLV поле данных должно состоять из одного или большего числа закодированных в структуре TLV информационных объектов;
- каждое не закодированное в структуре TLV поле данных должно состоять из одного или большего числа элементов данных, отвечающих требованиям соответствующей команды;
- структуру полей данных с оригинальным кодированием настоящий стандарт и стандарты серии ИСО/МЭК 7816 не устанавливают.

Настоящий стандарт поддерживает следующие два типа закодированных в структуре TLV информационных объектов в полях данных:

- информационный объект BER-TLV;
- информационный объект SIMPLE-TLV.

Настоящий стандарт не использует ни '00', ни 'FF' в качестве значения тега.

Каждый информационный объект BER-TLV должен состоять из двух или трех последовательных полей (см. ГОСТ Р ИСО/МЭК 8825 и приложение Г):

- поля тега T, состоящего из одного или большего числа последовательных байтов. Оно кодирует класс, тип конструкции и номер;
- поля длины, состоящего из одного или большего числа последовательных байтов. Оно кодирует целое число L;
- поля значения V (если L≠0), состоящего из L последовательных байтов. Если L=0, то информационный объект является пустым: поле значения отсутствует.

Каждый информационный объект SIMPLE-TLV должен состоять из двух или трех последовательных полей:

- поля тега T, состоящего из одиночного байта, кодирующего только номер от 1 до 254 (например, идентификатор записи). Оно не кодирует ни класс, ни тип конструкции;
- поля длины, состоящего из одного или трех последовательных байтов. Если значение начального байта поля длины находится в диапазоне от '00' до 'FE', то поле длины состоит из одиночного байта, кодирующего значение целого числа L от 0 до 254. Если начальный байт равен 'FF', то поле длины продолжается на два последующих байта, кодирующих значение целого числа L от 0 до 65535;
- поля значения V (если L≠0), состоящего из L последовательных байтов. Если L=0, то информационный объект является пустым: поле значения отсутствует.

Поля данных некоторых команд (например, команды ВЫБРАТЬ ФАЙЛ), поля значений информационных объектов SIMPLE-TLV и поля значений некоторых простых информационных объектов BER-TLV предназначаются для кодирования одного или большего числа элементов данных.

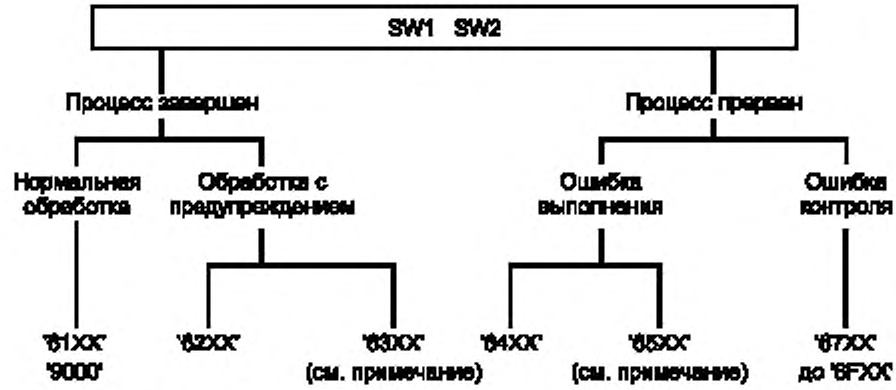
Поля данных некоторых других команд (например, команд, ориентированных на запись) и поля значений других простых информационных объектов BER-TLV предназначаются для кодирования одного или большего числа информационных объектов SIMPLE-TLV.

Поля данных прочих команд (например команд, ориентированных на объект) и поля значений составных информационных объектов BER-TLV предназначаются для кодирования одного или большего числа информационных объектов BER-TLV.

Примечание — Перед и между закодированными в структуре TLV информационными объектами или после них могут возникать байты со значениями '00' или 'FF' без какого-либо смыслового содержания (например, как следствие удаленных или измененных TLV-закодированных информационных объектов).

5.4.5 Байты состояния

Байты состояния SW1, SW2 ответа обозначают состояние обработки команды в карте. На рисунке 7 представлена структурная схема их значений, определяемых в настоящем стандарте.



Примечание — Если SW1 = '63' (или '65'), состояние энергонезависимой памяти изменено. Если SW1= '6X' (за исключением '63' и '65'), состояние энергонезависимой памяти не изменено.

Рисунок 7 — Структурная схема байтов состояния

Благодаря описанию байтов состояния, данному в ИСО/МЭК 7816-3, настоящий стандарт не определяет следующих значений последовательности байтов SW1-SW2:

- '60XX';
- '67XX', '6BXX', '6DXX', '6EXX', '6FXX' в каждом случае, если 'XX' ≠ '00';
- '9XXX', если 'XXX' ≠ '000'.

Следующие значения байтов SW1, SW2 определены для любого протокола (см. примеры в приложении А):

- если выполнение команды прерывается с ответом, где SW1 = '6C', то SW2 указывает значение (точную длину запрашиваемых данных), которое должно быть дано короткому полю L_e при повторной подаче той же команды до подачи любой другой команды;
- если команда (которая может относиться к случаю 2 или 4, см. таблицу 4 и рисунок 4) обрабатывается с ответом, где SW1 = '61', то SW2 указывает максимальное значение (длину дополнительных еще имеющихся данных), которое должно быть дано короткому полю L_e в команде ИЗВЛЕЧЬ ОТВЕТ, подаваемой перед подачей любой другой команды.

Примечание — Функциональные возможности, аналогичные тем, которые предлагаются при помощи значения '61XX', могут быть предложены на уровне приложения при помощи значения '9FXX'. Вместе с тем приложения могут использовать значение '9FXX' для иных целей.

В таблице 12, дополняемой таблицами 13—18, представлено общее смысловое содержание значений последовательности байтов SW1-SW2, определяемых в настоящем стандарте. Для каждой команды в соответствующем разделе представлено их более детальное содержание.

Таблицы 13—18 определяют значения байта SW2 для случаев, когда у байта SW1 значения составляют соответственно '62', '63', '65', '68', '69' и '6A'. Значения байта SW2, не определяемые в таблицах 13—18, являются RFU, за исключением значений от 'F0' до 'FF', которые настоящий стандарт не определяет.

Таблица 12 — Кодирование последовательности байтов SW1-SW2

SW1-SW2	Смысловое содержание
	Нормальная обработка
'9000'	Нет дальнейшего уточнения
'61XX'	Байт SW2 указывает число еще имеющихся ответных байтов (см. текст ниже)
	Обработка с предупреждением
'62XX'	Состояние энергонезависимой памяти без изменений (дальнейшее уточнение в байте SW2, см. таблицу 13)
'63XX'	Состояние энергонезависимой памяти изменено (дальнейшее уточнение в байте SW2, см. таблицу 14)

Продолжение таблицы 12

SW1-SW2	Смысловое содержание
	Ошибка выполнения
'64XX'	Состояние энергонезависимой памяти без изменений (SW2 = '00', другие значения являются RFU)
'65XX'	Состояние энергонезависимой памяти изменено (дальнейшее уточнение в байте SW2, см. таблицу 15)
'66XX'	Зарезервированы для сообщений, связанных с безопасностью (не определяются в настоящем стандарте)
	Ошибка контроля
'6700'	Неверно указанная длина
'68XX'	Функции, указанные в байте CLA, не поддерживаются (дальнейшее уточнение в байте SW2, см. таблицу 16)
'69XX'	Команда не разрешена (дальнейшее уточнение в байте SW2, см. таблицу 17)
'6AXX'	Неверно указанный(е) параметр(ы) P1 и(или) P2 (дальнейшее уточнение в байте SW2, см. таблицу 18)
'6B00'	Неверно указанный(е) параметр(ы) P1 и(или) P2
'6CXX'	Неверно указанная длина L_e: байт SW2 указывает точную длину (см. текст ниже)
'6D00'	Код команды не поддерживается или недействителен
'6E00'	Класс не поддерживается
'6F00'	Нет точного диагноза

Таблица 13 — Кодирование байта SW2, если SW1 = '62'

SW2	Смысловое содержание
'00'	Информация не предоставлена
'81'	Часть выдаваемых данных может быть искажена
'82'	Конец файла/записи достигнут до считывания L_e байтов
'83'	Выбранный файл недействителен
'84'	FCI не форматирована по 5.1.5

Таблица 14 — Кодирование байта SW2, если SW1 = '63'

SW2	Смысловое содержание
'00'	Информация не предоставлена
'81'	Файл заполнен при последней операции записи
'CX'	Счетчик, представленный в 'X' (имеет значения от 0 до 15) (точное смысловое содержание зависит от команды)

Таблица 15 — Кодирование байта SW2, если SW1 = '65'

SW2	Смысловое содержание
'00'	Информация не предоставлена
'81'	Отказ памяти

Таблица 16 — Кодирование байта SW2, если SW1 = '68'

SW2	Смысловое содержание
'00'	Информация не предоставлена
'81'	Логический канал не поддерживается
'82'	Безопасный обмен сообщениями не поддерживается

Таблица 17 — Кодирование байта SW2, если SW1 = '69'

SW2	Смысловое содержание
'00'	Информация не предоставлена
'81'	Команда несовместима со структурой файла
'82'	Состояние защиты неудовлетворительное
'83'	Метод аутентификации заблокирован
'84'	Ссылочные данные недействительны
'85'	Условия использования не удовлетворены
'86'	Команда невозможна (нет текущего EF)
'87'	Пропадание ожидаемых информационных объектов, связанных с SM
'88'	Информационные объекты, связанные с SM, некорректны

Таблица 18 — Кодирование байта SW2, если SW1 = '6A'

SW2	Смысловое содержание
'00'	Информация не предоставлена
'80'	Некорректные параметры в поле данных
'81'	Функция не поддерживается
'82'	Файл не найден
'83'	Запись не найдена
'84'	Область памяти в файле недостаточна
'85'	L_c не согласуется со структурой TLV
'86'	Некорректные параметры P1, P2
'87'	L_c не согласуется с P1, P2
'88'	Ссылочные данные не найдены

5.5 Логические каналы

5.5.1 Общая концепция

Логический канал, прослеживаемый на стыке между картой и устройством сопряжения, работает как логическая линия связи, ведущая к DF.

Функционирование одного логического канала должно осуществляться независимо от функционирования другого логического канала. То есть взаимосвязи команд на одном логическом канале не должны зависеть от взаимосвязей команд на другом логическом канале. Тем не менее логические каналы могут совместно использовать зависимое от приложения состояние защиты и, следовательно, для всех логических каналов могут иметь место взаимозависимости в части команд, связанных с защитными функциями (например, верификацией по паролю).

Команды, относящиеся к конкретному логическому каналу, несут соответствующий номер логического канала в байте CLA (см. таблицы 8 и 9). Логические каналы нумеруются от 0 до 3. Если карта поддерживает механизм логических каналов, то максимальное число имеющихся логических каналов указывается в данных о функциональных возможностях карты (см. 8.3.6).

Пары команда—ответ работают в соответствии со следующим описанием. Настоящий стандарт поддерживает только те пары команда—ответ, которые должны полностью завершаться до инициирования следующей пары команда—ответ. Не должно происходить разбиения команд и ответов на них по логическим каналам; между получением команды и посылкой ответа на эту команду активным должен быть только один логический канал. Если логический канал открывается, то он остается открытым до тех пор, пока не будет закрыт явным образом командой ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ.

Примечания

1 Может быть открыто более одного логического канала к одному и тому же DF, если это не исключается (о доступности файла см. в 5.1.5).

2 Более чем один логический канал может выбирать один и тот же EF, если это не исключается (о доступности файла см. в 5.1.5).

3 Команда ВЫБРАТЬ ФАЙЛ на любом логическом канале откроет текущий DF и, возможно, текущий EF. Следовательно на один логический канал приходится один текущий DF и, возможно, один текущий EF вследствие действия команды ВЫБРАТЬ ФАЙЛ и команд, использующих для достижения файла короткий идентификатор EF.

5.5.2 Основной логический канал

Основной логический канал постоянно доступен. Если ему присваивается номер, то это номер 0. Если байт класса кодируется по таблицам 8 и 9, то его биты b1 и b2 кодируют номер логического канала.

5.5.3 Открытие логического канала

Логический канал открывается в результате успешного завершения:

- либо команды ВЫБРАТЬ ФАЙЛ, обращающейся к DF путем назначения номера логического канала, отличного от 0, в байте класса;

- либо функции открытия команды ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ, которая или назначает номер логического канала, отличный от 0, в командном APDU, или запрашивает его назначение. Тогда он должен быть назначен картой и выдан в ответе.

5.5.4 Закрытие логического канала

Функция закрытия команды ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ может применяться для явного закрытия логического канала, использующего номер логического канала. После закрытия номер логического канала будет доступен для повторного использования. Основной логический канал не должен закрываться.

5.6 Безопасный обмен сообщениями

Безопасный обмен сообщениями (SM) предназначается для защиты передаваемых на карту и с карты сообщений (или их части) посредством обеспечения двух основных защитных функций: аутентификации данных и конфиденциальности данных.

Безопасный обмен сообщениями достигается путем применения одного или большего числа механизмов защиты. Каждый такой механизм включает в себя алгоритм, ключ, аргумент и, зачастую, исходные данные.

Передача и прием полей данных могут прерываться исполнением механизмов защиты. Настоящее описание не препятствуют определению посредством последовательного анализа, какие механизмы и элементы защиты должны использоваться для обработки оставшейся части поля данных.

Два или большее число механизмов защиты могут использовать один и тот же алгоритм вместе с разными режимами работы (см. ГОСТ Р ИСО/МЭК 10116). Представленные описания правил заполнения блоков данных незначащей информацией не препятствуют такой возможности.

Настоящий подраздел определяет три типа информационных объектов, связанных с SM:

- информационные объекты с незашифрованным значением, предназначаемые для переноса незашифрованных данных;

- информационные объекты механизмов защиты, предназначаемые для переноса вычисленных результатов механизмов защиты;

- информационные объекты вспомогательной функции защиты, предназначаемые для переноса управляющих ссылок и описателей ответа.

5.6.1 Концепция формата SM

В каждом сообщении, вовлекающем механизмы защиты, основанные на криптографии, поле

данных должно подчиняться базовым правилам кодирования АСН.1 (см. ГОСТ Р ИСО/МЭК 8825 и приложение Г), если в байте класса не указано иначе (см. 5.4.1).

Представленный в поле данных формат SM может быть выбранным:

- неявно, т.е. быть известным до подачи команды;
- явно, т.е. быть зафиксированным при помощи байта класса (см. таблицу 9).

Формат SM, определяемый в настоящем стандарте, кодируется с использованием информационных объектов BER-TLV, при этом:

- для SM зарезервирован контекстно-зависимый класс тегов (диапазон от '80' до 'BF');
- могут быть представлены информационные объекты других классов (например, информационные объекты прикладного класса);
- некоторые информационные объекты, связанные с SM, являются рекурсивными: их поле незашифрованного значения также кодируется информационными объектами BER-TLV, и там для SM также зарезервирован контекстно-зависимый класс.

В контекстно-зависимом классе бит b1 тега указывает, подлежит (b1=1) или не подлежит (b1=0) информационный объект, связанный с SM, объединению при вычислении информационного объекта для аутентификации. Если представлены информационные объекты других классов, то они подлежат объединению при таком вычислении.

5.6.2 Информационные объекты с незашифрованным значением

Для данных, не закодированных в информационных объектах BER-TLV, и для информационных объектов BER-TLV, включающих в себя объекты, связанные с SM, обязательной является инкапсуляция. Она не является обязательной для информационных объектов BER-TLV, не включающих в себя объекты, связанные с SM. В таблице 19 представлены незашифрованные информационные объекты для инкапсуляции.

Таблица 19 — Информационные объекты с незашифрованным значением

Тег	Значение
	Незашифрованное значение состоит из:
'B0', B1'	- информационных объектов BER-TLV, включающих объекты, связанные с SM
'B2', 'B3'	- информационных объектов BER-TLV, но не связанных с SM
'80', '81'	- данных, не закодированных информационными объектами BER-TLV
'96', '97'	- значения L_e в незащищенной команде
'99'	- информации о состоянии (например, из байтов SW1, SW2)

5.6.3 Информационные объекты для аутентификации

5.6.3.1 Информационный объект «криптографическая контрольная сумма»

Вычисление криптографических контрольных сумм (см. ИСО/МЭК 9797) предполагает наличие исходного контрольного блока, секретного ключа и алгоритма блочного шифрования, который не может быть обратимым. Алгоритм под управлением связанного с ним ключа, по существу, преобразует блок текущего ввода из *k* байтов (обычно восемь или 16) в блок текущего вывода той же длины.

Вычисление криптографической контрольной суммы выполняется последовательно по этапам.

а) Начальный этап

Начальный этап задает исходный контрольный блок, которым должен быть один из следующих блоков:

- нулевой блок, т.е. *k* байтов со значением '00';
- связующий блок, т.е. результат предыдущих вычислений, а именно: для команды — результирующий контрольный блок предшествующей команды, для ответа — результирующий контрольный блок предшествующего ответа;
- блок с начальным значением, предоставленный, например, внешним окружением;
- вспомогательный блок, являющийся результатом преобразования вспомогательных данных по соответствующему ключу. Если вспомогательные данные составляют менее *k* байтов, то они озаглавливаются битами, установленными в ноль, до достижения длины блока.

б) Промежуточный этап

Если таблица 9 применима(CLA = '0X', '8X', '9X' или 'AX') и биты b4 и b3 байта класса

установлены в единицу, то первый блок данных состоит из заголовка командного APDU (CLA, INS, P1, P2), за которым следуют один байт со значением '80' и *k*—5 байтов со значением '00'.

Криптографическая контрольная сумма должна объединять любой информационный объект, связанный с SM и имеющий тег с битом b1=1, и любой информационный объект с тегом, имеющим значение вне диапазона '80'—'BF'. Эти информационные объекты должны объединяться блок данных за блоком данных в текущем контрольном блоке. Разбивка на блоки данных должна осуществляться по следующим правилам:

- объединение в блоки должно быть непрерывным на границе между смежными информационными объектами, подлежащими объединению;
- заполнение незначащей информацией должно применяться в конце каждого подлежащего объединению информационного объекта, за которым либо следует информационный объект, не подлежащий объединению, либо отсутствует дальнейший информационный объект.

Незначащая информация состоит из одного обязательного байта со значением '80', за которым, если требуется, должны следовать от 0 до *k*—1 байтов, установленных в '00', пока соответствующий блок данных не будет заполнен *k* байтами. Заполнение незначащей информацией для аутентификации не оказывает влияния на передачу, поскольку заполняющие байты не должны передаваться.

Режимом работы является «последовательное блочное шифрование» (см. ГОСТ Р ИСО/МЭК 10116). Первый ввод представляет собой операцию сложения «Исключающее ИЛИ» над исходным контрольным блоком и первым блоком данных. Первый вывод является результатом первого ввода. Текущий ввод представляет собой операцию сложения «Исключающее ИЛИ» над предыдущим выводом и текущим блоком данных. Текущий вывод является результатом текущего ввода. Результирующий контрольный блок является последним выводом.

в) Заключительный этап

Заключительный этап выделяет криптографическую контрольную сумму (первые *m* байтов, но не менее четырех) из результирующего контрольного блока.

В таблице 20 представлен информационный объект «криптографическая контрольная сумма».

Таблица 20 — Информационный объект «криптографическая контрольная сумма»

Тег	Значение
'8E'	Криптографическая контрольная сумма (не менее четырех байтов)

5.6.3.2 Информационный объект «электронная цифровая подпись»

Вычисление электронной цифровой подписи, как правило, основывается на асимметричных криптографических методах. Существуют два типа электронных цифровых подписей:

- электронная цифровая подпись с присоединением,
- электронная цифровая подпись, обеспечивающая восстановление сообщения.

Вычисление электронной цифровой подписи с присоединением предполагает использование хэш-функции (см. ИСО/МЭК 10118-1 и ИСО/МЭК 10118-2). Ввод данных либо состоит из значения информационного объекта с данными ввода для выработки электронной цифровой подписи (см. таблицу 21), либо вычисляется согласно механизму, определяемому в 5.6.3.1.

Вычисление электронной цифровой подписи, обеспечивающей восстановление сообщения (см. ИСО/МЭК 9796), не предусматривает использование хэш-функции. Тем не менее в соответствии с потребностями приложения хэш-код может присутствовать как часть восстановленного сообщения, которое само может быть закодировано в структуре BER-TLV.

В таблице 21 представлены информационные объекты, относящиеся к электронной цифровой подписи.

Таблица 21 — Информационные объекты, относящиеся к электронной цифровой подписи

Тег	Значение
'9A', 'AC', 'BC' '9E'	Данные ввода для выработки электронной цифровой подписи Электронная цифровая подпись

5.6.4 Информационные объекты для конфиденциальности

Информационные объекты, используемые для конфиденциальности, предназначаются для переноса криптограммы, незашифрованное значение которой должно быть представлено одним из следующих трех случаев:

- информационными объектами BER-TLV, включающими в себя объекты, связанные с SM;
- информационными объектами BER-TLV, не включающими в себя объекты, связанные с SM;
- данными, не закодированными информационными объектами BER-TLV.

Если незашифрованное значение состоит из данных, не закодированных информационными объектами BER-TLV, заполнение незначащей информацией должно иметь индикацию. Если оно применяется, но индикации нет, то для него действуют правила, определяемые в 5.6.3.1.

В таблице 22 представлены информационные объекты для конфиденциальности.

Каждый такой информационный объект может использовать любой криптографический алгоритм и любой режим работы благодаря соответствующей ссылке на алгоритм (см. 5.6.5.1). При отсутствии такой ссылки и выбранного в неявной форме механизма для конфиденциальности должен применяться механизм, устанавливаемый по умолчанию.

Таблица 22 — Информационные объекты для конфиденциальности

Тег	Значение
	Криптограмма, ее незашифрованное значение, состоящее из:
'82', '83'	- информационных объектов BER-TLV, включающих объекты, связанные с SM
'84', '85'	- информационных объектов BER-TLV, но не связанных с SM
'86', '87'	Байт индикатора заполнения незначащей информацией (см. таблицу 23), сопровождаемый криптограммой (незашифрованное значение, не закодированное информационными объектами BER-TLV)

Для вычисления криптограммы, которой предшествует индикатор заполнения незначащей информацией, механизм по умолчанию представляет собой блочное шифрование в режиме «электронный кодовый справочник» (см. ГОСТ Р ИСО/МЭК 10116). Использование блочного шифрования может включать в себя заполнение блоков данных незначащей информацией. Заполнение незначащей информацией для конфиденциальности оказывает влияние на передачу, криптограмма (один или большее число блоков) получается длиннее незашифрованного текста.

В таблице 23 представлен байт индикатора заполнения незначащей информацией.

Для вычисления криптограммы, которой не предшествует байт индикатора заполнения незначащей информацией, механизм по умолчанию представляет собой поточное шифрование вместе с операцией сложения «Исключающее ИЛИ». В этом случае криптограмма является результатом операции сложения «Исключающее ИЛИ», выполняемой над строкой байтов данных, подлежащих сокрытию, и скрывающей строкой той же длины. Сокрытие, таким образом, не требует заполнения незначащей информацией, а информационные объекты, скрытые в поле значения, восстанавливаются при помощи той же операции.

Таблица 23 — Байт индикатора заполнения незначащей информацией

Значение	Смысловое содержание
'00'	Нет дальнейшей индикации
'01'	Заполнение незначащей информацией по 5.6.3.1
'02'	Заполнение отсутствует
От '80' до '8E'	Оригинальное заполнение
	Прочие значения являются RFU

5.6.5 Информационные объекты вспомогательной функции защиты

Алгоритм, ключ и, возможно, исходные данные могут выбираться для каждого механизма защиты следующим образом:

- неявно, т.е. быть известными до подачи команды;
- явно, при помощи управляющих ссылок, вложенных в шаблон управляющих ссылок.

Каждое командное сообщение может нести шаблон описателя ответа, задающий информационные объекты, требуемые в ответе. Внутри описателя ответа механизмы защиты еще не применяются; принимающая сторона должна применить их для составления ответа.

5.6.5.1 Управляющие ссылки

В таблице 24 представлены шаблоны управляющих ссылок.

Таблица 24 — Шаблоны управляющих ссылок

Тег	Смысловое содержание
'B4', 'B5'	Шаблон, действительный для криптографической контрольной суммы
'B6', 'B7'	Шаблон, действительный для электронной цифровой подписи
'B8', 'B9'	Шаблон, действительный для конфиденциальности

Последняя возможная позиция шаблона управляющих ссылок располагается непосредственно перед первым информационным объектом, для которого применяется указываемый механизм. Например, последняя возможная позиция шаблона для криптографической контрольной суммы располагается непосредственно перед первым информационным объектом, объединяемым при вычислении.

Каждая управляющая ссылка действует до тех пор, пока не будет предоставлена новая управляющая ссылка для того же механизма. Например, команда может фиксировать управляющие ссылки для следующей команды.

Каждый шаблон управляющих ссылок предназначается для переноса информационных объектов, представляющих управляющие ссылки (см. таблицу 25): на алгоритм, файл, ключ, исходные данные; и только в шаблоне управляющих ссылок для конфиденциальности осуществляется перенос еще и ссылки на содержимое криптограммы.

Ссылка на алгоритм указывает алгоритм и режим его работы (см. ИСО/МЭК 9979 и ГОСТ Р ИСО/МЭК 10116). Структуру и кодирование этой ссылки настоящий стандарт не определяет.

Ссылка на файл обозначает тот файл, в котором действительна ссылка на ключ. Если ссылка на файл не представлена, тогда ссылка на ключ действительна в текущем DF.

Ссылка на ключ идентифицирует ключ, подлежащий использованию.

Ссылка на исходные данные, будучи примененной для криптографических контрольных сумм, определяет исходный контрольный блок. Если ссылка на исходные данные не представлена и не выбран неявным образом исходный контрольный блок, то должен использоваться нулевой блок. Кроме того, перед передачей первого информационного объекта для конфиденциальности, использующего поточное шифрование, шаблон для конфиденциальности должен предоставить вспомогательные данные для инициализации вычисления строки скрывающих байтов.

Ссылка на содержимое криптограммы указывает содержание криптограммы (например, секретные ключи, исходный пароль, управляющие слова). Первый байт поля значения представляющего ее информационного объекта называется байтом описателя криптограммы и является обязательным. Диапазон его значений от '00' до '7F' является RFU, а диапазон от '80' до 'FF' отведен для собственного использования.

Таблица 25 — Информационные объекты, представляющие управляющие ссылки

Тег	Значение
'80'	Ссылка на алгоритм
	Ссылка на файл:
'81'	- идентификатор файла или путь
'82'	- имя DF
	Ссылка на ключ:
'83'	- для прямого использования
'84'	- для вычисления сеансового ключа

Продолжение таблицы 25

Тег	Значение
	Ссылка на исходные данные
	Исходный контрольный блок:
'85'	L = 0, нулевой блок
'86'	L = 0, связующий блок
'87'	L = 0, предыдущий блок с начальным значением плюс один
	L = *k*, блок с начальным значением
	Вспомогательные данные:
'88'	L = 0, данные предыдущей задачи, использованной в обмене, плюс один
	L ≠ 0, нет дальнейшей индикации
От '89' до '8D'	L = 0, индекс оригинального элемента данных
	L ≠ 0, значение оригинального элемента данных
'8E'	Ссылка на содержимое криптограммы

5.6.5.2 Описатель ответа

Шаблон описателя ответа, если он представлен в поле данных командного APDU, должен задавать структуру соответствующего ответа. Пустые информационные объекты должны перечислять все данные, необходимые для выработки ответа.

Элементы защиты (алгоритмы, ключи и исходные данные), используемые для обработки поля данных командного сообщения, могут отличаться от элементов защиты, используемых для выработки поля данных последующего ответного сообщения.

Должны применяться следующие правила:

- карта должна заполнять каждый пустой простой информационный объект;
- каждый шаблон управляющих ссылок, присутствующий в описателе ответа, должен быть представлен в ответе на том же месте с теми же управляющими ссылками для алгоритма, файла и ключа. Если описатель ответа предоставляет вспомогательные данные, тогда соответствующий информационный объект в ответе должен быть пустым. Если в описателе ответа присутствует пустой информационный объект для вспомогательных данных, то в ответе он должен быть заполнен;
- применяя соответствующие механизмы защиты вместе с выбранными элементами защиты, карта должна осуществлять вывод всех запрашиваемых информационных объектов механизмов защиты.

В таблице 26 представлен шаблон описателя ответа.

Таблица 26 — Шаблон описателя ответа

Тег	Значение
'BA', 'BB'	Описатель ответа

5.6.6 Состояния после обработки, связанные с SM

При любой команде, использующей безопасный обмен сообщениями, могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '69', а байт SW2 равен:

'87' — пропадание ожидаемых информационных объектов, связанных с SM;

'88' — информационные объекты, связанные с SM, некорректны.

5.7 Влияние безопасного обмена сообщениями на структуры APDU-сообщений

Структуры APDU-сообщений установлены в 5.3. В соответствии с 5.3.1 командный APDU состоит из обязательного заголовка команды из четырех байтов, за которым условно следует тело команды (см. рисунки 3 и 4); декодирование тела команды установлено в 5.3.2 (см. рисунок 5 и

таблицу 5). В соответствии с 5.3.3 ответный APDU состоит из условного тела ответа, за которым следует обязательный завершитель ответа из двух байтов (см. рисунок 6). На рисунке 8 представлены структуры APDU-сообщений.

Заголовок команды	Тело команды
CLA INS P1 P2 (четыре байта)	[Поле L_c] [Поле данных] [Поле L_e] (L байтов, обозначаемых B_1 .. B_L)

Тело ответа	Завершитель ответа
[Поле данных] (L_r байтов данных)	SW1 SW2 (два байта)

Рисунок 8 — Структуры APDU-сообщений

Раздел 6 устанавливает APDU-команды и APDU-ответы для основных межотраслевых команд. Раздел 7 устанавливает APDU-команды и APDU-ответы для межотраслевых команд, ориентированных на передачу данных. Разделы 6 и 7 не описывают влияние безопасного обмена сообщениями (см. 5.6) на структуры APDU-сообщений. Поэтому семантические значения полей длины и полей данных в разделах 6 и 7 кажутся противоречащими их синтаксическим значениям в 5.3.

Настоящий подраздел определяет влияние безопасного обмена сообщениями, установленного в 5.6, на структуры APDU-сообщений, установленные в 5.3, с тем чтобы избежать вышеупомянутого возможного неправильного понимания.

Для обеспечения защиты APDU-команды, где байт CLA имеет определенное значение, соответствующее таблице 9, а именно '0X', '8X', '9X' или 'AX', бит b4 в байте CLA должен быть установлен в единицу, что обозначено CLA* на рисунке 9 и в приложении E; если тело команды присутствует, оно должно быть декодируемым в соответствии с 5.3.2 и инкапсулировано следующим образом.

При наличии поля данных перенос L_c байтов данных должен осуществляться:

- либо посредством информационного объекта с незашифрованным значением (тег равен '80', '81', 'B2', 'B3', см. таблицу 19);
- либо посредством информационного объекта для конфиденциальности (тег со значением от '84' до '87', см. таблицу 22).

При наличии поля L_e перенос значения L_e должен осуществляться посредством информационного объекта, представляющего L_e (тег равен '96' или '97', см. таблицу 19); его поле значения кодирует положительное целое число без знака в одном или двух байтах; нулевое значение и пустой информационный объект означают максимум.

Аналогично ответный APDU должен быть инкапсулирован следующим образом.

При наличии поля данных перенос L_r байтов данных должен осуществляться:

- либо посредством информационного объекта с незашифрованным значением (тег равен '80', '81', 'B2', 'B3', см. таблицу 19);
- либо посредством информационного объекта для конфиденциальности (тег со значением от '84' до '87', см. таблицу 22).

При необходимости перенос завершителя ответа должен осуществляться посредством информационного объекта, представляющего информацию о состоянии (тег равен '99', см. таблицу 19); пустой информационный объект означает, что последовательность байтов SW1—SW2 = '9000'.

На рисунке 9 представлены структуры защищенных APDU-сообщений, при этом:

- каждое новое поле данных может нести дополнительные информационные объекты, связанные с SM, например в конце – криптографическую контрольную сумму (тег равен '8E'). В приложении E приведены примеры для иллюстрации;
- новое поле L_e дает длину нового поля данных защищенного командного APDU;
- новое поле L_e должно быть пустым, если не ожидается поле данных в защищенном ответном APDU; в противном случае оно должно содержать только нули;
- новый завершитель ответа кодирует состояние принимающей стороны после обработки защищенной команды. Для защиты он может быть инкапсулирован.

Заголовок команды	Тело команды
CLA* INS P1 P2 (четыре байта)	[Новое поле L_c] { [Новое поле данных] = [T L_c Байты данных] [T '01' или '02' L_e] } [Новое поле L_e]

Тело ответа	Завершитель ответа
[Новое поле данных] = [T L_r Байты данных] [T '02' Новые SW1 SW2]	Новые SW1 SW2 (два байта)

Примечания

1 Длина от 1 до 127 кодируется в поле длины информационных объектов BER-TLV также как в полях длины APDU-сообщений. Для длины 128 и более способы кодирования различаются.

2 Как указано выше, в новых полях данных могут быть представлены дополнительные (дальнейшие) или другие информационные объекты, связанные с SM.

3 При безопасном обмене сообщениями не всегда выявляется, имеют ли данные, подлежащие защите, структуру BER-TLV. В таких случаях рекомендуются теги '80', '81', '86' и '87'.

Рисунок 9 — Структуры защищенных APDU-сообщений

6 Основные межотраслевые команды

Поддержка всех описываемых команд или всех опций поддерживаемых команд не является обязательной для всех карт, соответствующих настоящему стандарту.

Когда требуется участие в международном информационном обмене, набор системных услуг, предоставляемых картой, и связанных с ними команд и опций должен использоваться, как определено в разделе 9.

В таблице 11 представлен полный перечень команд, определяемых в настоящем стандарте.

Настоящий раздел не предусматривает описаний влияния безопасного обмена сообщениями (см. 5.6) на структуры сообщений.

Перечни состояний ошибки и состояний предупреждения, приводимые в пятом пункте каждого подраздела настоящего раздела не являются исчерпывающими (см. 5.4.5).

6.1 Команда СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ

6.1.1 Определение и область применения

Ответное сообщение команды СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ передает содержимое (его часть) файла EF с прозрачной структурой.

6.1.2 Условия использования и защиты

Если команда содержит разрешенный короткий идентификатор EF, она устанавливает указываемый файл в качестве текущего EF.

Обработка команды осуществляется в выбранном в текущий момент EF. Команда может быть выполнена только в том случае, если состояние защиты удовлетворяет атрибутам секретности, определенным для данного EF для функции считывания.

Команда должна прерываться в том случае, если она применяется к EF без прозрачной структуры.

6.1.3 Командное сообщение

Командный APDU команды СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ представлен в таблице 27.

Таблица 27 — Командный APDU команды СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ

CLA	Как определено в 5.4.1
INS	'B0'
P1, P2	См. текст ниже
Поле L_c	Пустое
Поле данных	«
Поле L_e	Число байтов, подлежащих считыванию

Если бит b8=1 в байте P1, то биты b7 и b6 байта P1 устанавливаются в ноль (биты RFU), биты b5—b1 байта P1 являются коротким идентификатором EF, а байт P2 представляет собой смещение первого байта, подлежащего считыванию, в единицах данных от начала файла.

Если бит b8=0 в байте P1, то сцепление байтов P1 ‖ P2 представляет собой смещение первого байта, подлежащего считыванию, в единицах данных от начала файла.

6.1.4 Ответное сообщение (номинальный случай)

Если поле L_e содержит только нули, то в пределах максимума 256 (для короткого поля) или 65536 (для расширенного поля) все байты до конца файла должны быть считаны.

Ответный APDU команды СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ представлен в таблице 28.

Таблица 28 — Ответный APDU команды СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ

Поле данных	Считанные данные (L_e байтов)
SW1, SW2	Байты состояния

6.1.5 Состояния после обработки

Могут возникать следующие специфические состояния предупреждения.

Если байт SW1 = '62', а байт SW2 равен:

'81' — часть выдаваемых данных может быть искажена;

'82' — конец файла достигнут до считывания L_e байтов.

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '67', а байт SW2 равен:

'00' — неверно указанная длина (несоответствующее поле L_e).

Если байт SW1 = '69' , а байт SW2 равен:

'81' — команда несовместима со структурой файла;

'82' — состояние защиты неудовлетворительное;

'86' — команда невозможна (нет текущего EF).

Если байт SW1 = '6A', а байт SW2 равен:

'81' — функция не поддерживается;

'82' — файл не найден.

Если байт SW1 = '6B', а байт SW2 равен:

'00' — неверно указанные параметры (смещение выходит за пределы EF).

Если байт SW1 = '6C', а байт SW2 равен:

'XX' — неверно указанная длина (несоответствующее поле L_e; 'XX' указывает точную длину).

6.2 Команда ВВЕСТИ ДВОИЧНОЕ ЗНАЧЕНИЕ

6.2.1 Определение и область применения

Командное сообщение команды ВВЕСТИ ДВОИЧНОЕ ЗНАЧЕНИЕ инициирует запись двоичных значений в EF.

В зависимости от атрибутов файла команда должна выполнять одну из следующих операций:

- логического сложения «ИЛИ» над битами, уже присутствующими в карте, и битами, передаваемыми в командном APDU (логическое состояние битов файла после стирания представлено нулем);

- логического умножения «И» над битами, уже присутствующими в карте, и битами, передаваемые в командном APDU (логическое состояние битов файла после стирания представлено единицей);

- однократную запись в карту битов, передаваемых в командном APDU.

Если индикация не дается в байте кодирования данных (см. таблицу 86), должен применяться режим логического сложения «ИЛИ».

6.2.2 Условия использования и защиты

Если команда содержит разрешенный короткий идентификатор EF, она устанавливает указываемый файл в состояние текущего EF.

Обработка команды осуществляется в выбранном в текущий момент EF. Команда может быть выполнена только в том случае, если состояние защиты удовлетворяет атрибутам секретности для функций записи.

Если команда ВВЕСТИ ДВОИЧНОЕ ЗНАЧЕНИЕ однажды уже была применена к единице данных файла EF, допускающего только однократную запись, то любая последующая операция записи, обращающаяся к этой же единице данных, прерывается, если содержание этой единицы

данных или присоединенный к ней индикатор логического состояния после стирания (если он применяется) отличается от самого логического состояния после стирания.

Команда должна прерываться в том случае, если она применяется к EF без прозрачной структуры.

6.2.3 Командное сообщение

Командный APDU команды ВВЕСТИ ДВОИЧНОЕ ЗНАЧЕНИЕ представлен в таблице 29.

Таблица 29 — Командный APDU команды ВВЕСТИ ДВОИЧНОЕ ЗНАЧЕНИЕ

CLA	Как определено в 5.4.1
INS	'D0'
P1, P2	См. текст ниже
Поле L_c	Длина последующего поля данных
Поле данных	Строка единиц данных, подлежащих записи
Поле L_e	Пустое

Если бит b8=1 в байте P1, то биты b7 и b6 байта P1 устанавливаются в ноль (биты RFU), биты b5—bl байта P1 являются коротким идентификатором EF, а байт P2 представляет собой смещение первого байта, подлежащего записи, в единицах данных от начала файла.

Если бит b8=0 в байте P1, то сцепление байтов P1 ‖ P2 представляет собой смещение первого байта, подлежащего записи, в единицах данных от начала файла.

6.2.4 Ответное сообщение (номинальный случай)

Ответный APDU команды ВВЕСТИ ДВОИЧНОЕ ЗНАЧЕНИЕ представлен в таблице 30.

Таблица 30 — Ответный APDU команды ВВЕСТИ ДВОИЧНОЕ ЗНАЧЕНИЕ

Поле данных	Пустое
SW1, SW2	Байты состояния

6.2.5 Состояния после обработки

Может возникать следующее специфическое состояние предупреждения.

Если байт SW1 = '63', а байт SW2 равен:

'CX' — счетчик (успешная запись, но после использования внутренней программы повторений; 'X' ≠ '0' указывает число повторных попыток; 'X' = '0' означает, что счетчик не предусмотрен).

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '65', а байт SW2 равен:

'81' — отказ памяти (безуспешная запись).

Если байт SW1 " '67', а байт SW2 равен:

'00' — неверно указанная длина (несоответствующее поле L_c).

Если байт SW1 = '69', а байт SW2 равен:

'81' — команда несовместима со структурой файла;

'82' — состояние защиты неудовлетворительное;

'86' — команда невозможна (нет текущего EF).

Если байт SW1 = '6A', а байт SW2 равен:

'81' — функция не поддерживается;

'82' — файл не найден.

Если байт SW1 = '6B', а байт SW2 равен:

'00' — неверно указанные параметры (смещение выходит за пределы EF).

6.3 Команда ОБНОВИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ

6.3.1 Определение и область применения

Командное сообщение команды ОБНОВИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ инициирует обновление битов, уже присутствующих в карте, битами, передаваемыми в командном APDU.

6.3.2 Условия использования и защиты

Если команда содержит разрешенный короткий идентификатор EF, она устанавливает указываемый файл в состояние текущего EF.

Обработка команды осуществляется в выбранном в текущий момент EF. Команда может быть

выполнена только в том случае, если состояние защиты удовлетворяет атрибутам секретности для функции обновления.

Команда должна прерываться в том случае, если она применяется к EF без прозрачной структуры.

6.3.3 Командное сообщение

Командный APDU команды ОБНОВИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ представлен в таблице 31.

Таблица 31 — Командный APDU команды ОБНОВИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ

CLA	Как определено в 5.4.1
INS	'D6'
P1, P2	См. текст ниже
Поле L_c	Длина последующего поля данных
Поле данных	Строка единиц данных для обновления
Поле L_e	Пустое

Если бит b8=1 в байте P1, то биты b7 и b6 байта P1 устанавливаются в ноль (биты RFU), биты b5—b1 байта P1 являются коротким идентификатором EF, а байт P2 представляет собой смещение первого байта, подлежащего обновлению, в единицах данных от начала файла.

Если бит b8=0 в байте P1, то сцепление байтов P1 ‖ P2 представляет собой смещение первого байта, подлежащего обновлению, в единицах данных от начала файла.

6.3.4 Ответное сообщение (номинальный случай)

Ответный APDU команды ОБНОВИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ представлен в таблице 32.

Таблица 32 — Ответный APDU команды ОБНОВИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ

Поле данных	Пустое
SW1, SW2	Байты состояния

6.3.5 Состояния после обработки

Может возникать следующее специфическое состояние предупреждения.

Если байт SW1 = '63', а байт SW2 равен:

'CX' — счетчик (успешное обновление, но после использования внутренней программы повторений; 'X' ≠ '0' указывает число повторных попыток; 'X' = '0' означает, что счетчик не предусмотрен).

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '65', а байт SW2 равен:

'81' — отказ памяти (безуспешное обновление).

Если байт SW1 ¨ '67', а байт SW2 равен:

'00' — неверно указанная длина (несоответствующее поле L_c).

Если байт SW1 = '69', а байт SW2 равен:

'81' — команда несовместима со структурой файла;

'82' — состояние защиты неудовлетворительное;

'86' — команда невозможна (нет текущего EF).

Если байт SW1 = '6A', а байт SW2 равен:

'81' — функция не поддерживается;

'82' — файл не найден.

Если байт SW1 = '6B', а байт SW2 равен:

'00' — неверно указанные параметры (смещение выходит за пределы EF).

6.4 Команда ИСКЛЮЧИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ

6.4.1 Определение и область применения

Командное сообщение команды ИСКЛЮЧИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ устанавливает содержимое (его часть) файла EF в его логическое состояние после стирания последовательно, начиная с заданного смещения.

6.4.2 Условия использования и защиты

Если команда содержит разрешенный короткий идентификатор EF, она устанавливает указываемый файл в состояние текущего EF.

Обработка команды осуществляется в выбранном в текущий момент EF. Команда может быть выполнена только в том случае, если состояние защиты удовлетворяет атрибутам секретности для функции стирания.

Команда должна прерываться в том случае, если она применяется к EF без прозрачной структуры.

6.4.3 Командное сообщение

Командный APDU команды ИСКЛЮЧИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ представлен в таблице 33.

Таблица 33 — Командный APDU команды ИСКЛЮЧИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ

CLA	Как определено в 5.4.1
INS	'0E'
P1, P2	См. текст ниже
Поле L_c	Пустое или '02'
Поле данных	См. текст ниже
Поле L_e	Пустое

Если бит b8=1 в байте P1, то биты b7 и b6 байта P1 устанавливаются в ноль (биты RFU), биты b5—bl байта P1 являются коротким идентификатором EF, а байт P2 представляет собой смещение первого байта, подлежащего стиранию, в единицах данных от начала файла.

Если бит b8=0 в байте P1, то сцепление байтов P1 ‖ P2 представляет собой смещение первого байта, подлежащего стиранию, в единицах данных от начала файла.

Если поле данных представлено, оно кодирует смещение первой единицы данных, не подлежащей стиранию. Это смещение должно превышать смещение, закодированное в байтах P1, P2. Если поле данных является пустым, команда стирает файл до конца.

6.4.4 Ответное сообщение (номинальный случай)

Ответный APDU команды ИСКЛЮЧИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ представлен в таблице 34.

Таблица 34 — Ответный APDU команды ИСКЛЮЧИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ

Поле данных	Пустое
SW1, SW2	Байты состояния

6.4.5 Состояния после обработки

Может возникать следующее специфическое состояние предупреждения.

Если байт SW1 = '63', а байт SW2 равен:

'CX' — счетчик (успешное стирание, но после использования внутренней программы повторений; 'X' ≠ '0' указывает число повторных попыток; 'X' = '0' означает, что счетчик не предусмотрен).

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '65', а байт SW2 равен:

'81' — отказ памяти (безуспешное стирание).

Если байт SW1 = '67', а байт SW2 равен:

'00' — неверно указанная длина (несоответствующее поле L_c).

Если байт SW1 = '69', а байт SW2 равен:

'81' — команда несовместима со структурой файла;

'82' — состояние защиты неудовлетворительное;

'86' — команда невозможна (нет текущего EF).

Если байт SW1 = '6A', а байт SW2 равен:

'81' — функция не поддерживается;

'82' — файл не найден.

Если байт SW1 = '6B', а байт SW2 равен:

'00' — неверно указанные параметры (смещение выходит за пределы EF).

6.5 Команда СЧИТАТЬ ЗАПИСЬ(И)

6.5.1 Определение и область применения

Ответное сообщение команды СЧИТАТЬ ЗАПИСЬ(И) передает содержимое указанной(ых) записи(ей) (или начальной части одной записи) файла EF.

6.5.2 Условия использования и защиты

Команда может быть выполнена только в том случае, если состояние защиты удовлетворяет атрибутам секретности данного EF для функции считывания.

Если выбор EF осуществляется одновременно с подачей команды, то эта команда может обрабатываться без идентификации данного файла.

Если команда содержит разрешенный короткий идентификатор EF, она устанавливает указываемый файл в состояние текущего EF, а указатель текущей записи — в исходное положение.

Команда должна прерываться в том случае, если она применяется к EF без структуры из записей.

6.5.3 Командное сообщение

Командный APDU команды СЧИТАТЬ ЗАПИСЬ(И) представлен в таблице 35.

Таблица 35 — Командный APDU команды СЧИТАТЬ ЗАПИСЬ(И)

CLA	Как определено в 5.4.1
INS	'B2'
P1	Номер или идентификатор первой подлежащей считыванию записи ('00' указывает текущую запись)
P2	Управление ссылками в соответствии с таблицей 36
Поле L_c	Пустое
Поле данных	«
Поле L_e	Число байтов, подлежащих считыванию

Таблица 36 — Кодирование байта управления ссылками P2

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
0	0	0	0	0	—	—	—	Выбираемый в текущий момент EF
×	×	×	×	×	—	—	—	Короткий идентификатор EF
(не все равны)								
1	1	1	1	1	—	—	—	RFU
—	—	—	—	—	1	×	×	Использование номера записи в байте P1:
—	—	—	—	—	1	0	0	- считывание записи с номером в байте P1
—	—	—	—	—	1	0	1	- считывание всех записей, начиная с записи с номером в байте P1 и заканчивая последней
—	—	—	—	—	1	1	0	- считывание всех записей, начиная с последней и заканчивая записью с номером в байте P1
—	—	—	—	—	1	1	1	RFU
—	—	—	—	—	0	×	×	Использование идентификатора записи в байте P1:
—	—	—	—	—	0	0	0	- считывание первого вхождения
—	—	—	—	—	0	0	1	- считывание последнего вхождения
—	—	—	—	—	0	1	0	- считывание следующего вхождения
—	—	—	—	—	0	1	1	- считывание предыдущего вхождения

6.5.4 Ответное сообщение (номинальный случай)

Если поле L_e содержит только нули, то в зависимости от последовательности битов b3b2b1 в байте P2 и в пределах максимума 256 (для короткого поля) или 65536 (для расширенного поля) команда должна обеспечивать полное считывание:

- либо одной запрашиваемой записи;
- либо запрашиваемой последовательности записей.

Ответный APDU команды СЧИТАТЬ ЗАПИСЬ(И) представлен в таблице 37.

Таблица 37 — Ответный APDU команды СЧИТАТЬ ЗАПИСЬ(И)

Поле данных	L_r байтов (L_r может быть равно L_e), см. таблицы 38-1 и 38-2
SW1, SW2	Байты состояния

Если записи являются информационными объектами SIMPLE-TLV (см. 5.4.4), то для этого случая в таблицах 38-1 и 38-2 для наглядности представлен формат поля данных ответного сообщения.

Таблица 38-1 — Поле данных ответа в случае считывания одной записи

Случай а). Частичное считывание одной записи

T_n (один байт)	L_n (один или три байта)	Первые байты данных записи
L_e байтов		

Этот случай применим, когда поле L_e содержит не только нули.

Случай б). Полное считывание одной записи

T_n (один байт)	L_n (один или три байта)	Все байты данных записи (L_n байтов)

Этот случай применим, когда поле L_e содержит одни нули.

Таблица 38-2 — Поле данных ответа в случае считывания нескольких записей

Случай в). Частичное считывание последовательности записей

Запись # n $T_n \parallel L_n \parallel V_n$	... ...	Первые байты записи # n+m $T_{n+m} \parallel L_{n+m} \parallel V_{n+m}$
L_e байтов		

Этот случай применим, когда поле L_e содержит не только нули.

Случай г). Считывание многочисленных записей до конца файла

Запись # n $T_n \parallel L_n \parallel V_n$	... ...	Запись # n+m $T_{n+m} \parallel L_{n+m} \parallel V_{n+m}$

Этот случай применим, когда поле L_e содержит одни нули.

Сравнение длины поля данных с его структурой TLV показывает характер данных: единственная (считывание одной записи) или последняя (считывание всех записей) запись является неполной, полной или дополненной незначащей информацией.

Примечание — Если не используется кодирование в структуре TLV, то результатом функции считывания всех записей будет получение нескольких записей без их стандартного разграничения.

6.5.5 Состояния после обработки

Могут возникать следующие специфические состояния предупреждения.

Если байт SW1 = '62', а байт SW2 равен:

'81' — часть выдаваемых данных может быть искажена;

'82' — конец записи достигнут до считывания L_e байтов.

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '67', а байт SW2 равен:

'00' — неверно указанная длина (пустое поле L_e).

Если байт SW1 = '69' , а байт SW2 равен:

'81' — команда несовместима со структурой файла;

'82' — состояние защиты неудовлетворительное.

Если байт SW1 = '6A', а байт SW2 равен:

'81' — функция не поддерживается;

'82' — файл не найден;
'83' — запись не найдена.
Если байт SW1 = '6C', а байт SW2 равен:
'XX' — неверно указанная длина (несоответствующее поле L_e; 'XX' указывает точную длину).

6.6 Команда ВВЕСТИ ЗАПИСЬ

6.6.1 Определение и область применения

Командное сообщение команды ВВЕСТИ ЗАПИСЬ инициирует одну из следующих операций:
- однократную запись передаваемых данных (в виде записи);
- операцию логического сложения «ИЛИ» над байтами данных записи, уже присутствующей в карте, и байтами данных записи, передаваемой в командном APDU;
- операцию логического умножения «И» над байтами данных записи, уже присутствующей в карте, и байтами данных записи, передаваемой в командном APDU.

Если индикация не дается в байте кодирования данных (см. таблицу 86), должна применяться операция логического сложения «ИЛИ».

При использовании адресации текущей записи команда должна устанавливать указатель записи на успешно записанную запись.

6.6.2 Условия использования и защиты

Команда может быть выполнена только в том случае, если состояние защиты удовлетворяет атрибутам секретности данного EF для функций записи.

Если выбор EF осуществляется одновременно с подачей команды, то эта команда может обрабатываться без идентификации данного файла.

Если команда содержит разрешенный короткий идентификатор EF, она устанавливает указываемый файл в состояние текущего EF, а указатель текущей записи — в исходное положение.

Команда должна прерываться в том случае, если она применяется к EF без структуры из записей.

Опция команды «предыдущая» (P2 = xxxxx011), примененная к циклическому файлу, ведет себя также, как команда ПРИСОЕДИНИТЬ ЗАПИСЬ.

6.6.3 Командное сообщение

Командный APDU команды ВВЕСТИ ЗАПИСЬ представлен в таблице 39.

Таблица 39 — Командный APDU команды ВВЕСТИ ЗАПИСЬ

CLA	Как определено в 5.4.1
INS	'D2'
P1	P1 = '00' обозначает текущую запись, P1 ≠ '00' — номер указываемой записи
P2	См. таблицу 40
Поле L_c	Длина последующего поля данных
Поле данных	Запись, подлежащая операции записи
Поле L_e	Пустое

Таблица 40 — Кодирование байта управления ссылками P2

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
0	0	0	0	0	—	—	—	Выбираемый в текущий момент EF
×	×	×	×	×	—	—	—	Короткий идентификатор EF
(не все равны)								
—	—	—	—	—	0	0	0	Первая запись
—	—	—	—	—	0	0	1	Последняя запись
—	—	—	—	—	0	1	0	Следующая запись
—	—	—	—	—	0	1	1	Предыдущая запись
—	—	—	—	—	1	0	0	Номер записи дан в байте P1
Любое другое значение								RFU

Если записи являются информационными объектами SIMPLE-TLV (см. 5.4.4), то для этого случая в таблице 41 для наглядности представлен формат поля данных командного сообщения.

Таблица 41 — Поле данных команды

Полная запись одной записи

T_n (один байт)	L_n (один или три байта)	Все байты данных записи (Ln байтов)

6.6.4 Ответное сообщение (номинальный случай)

Ответный APDU команды ВВЕСТИ ЗАПИСЬ представлен в таблице 42.

Таблица 42 — Ответный APDU команды ВВЕСТИ ЗАПИСЬ

Поле данных	Пустое
SW1, SW2	Байты состояния

6.6.5 Состояния после обработки

Может возникать следующее специфическое состояние предупреждения.

Если байт SW1 = '63', а байт SW2 равен:

'CX' — счетчик (успешная запись, но после использования внутренней программы повторений; 'X' ≠ '0' указывает число повторных попыток; 'X' = '0' означает, что счетчик не предусмотрен).

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '65', а байт SW2 равен:

'81' — отказ памяти (безуспешная запись).

Если байт SW1 = '67', а байт SW2 равен:

'00' — неверно указанная длина (пустое поле L_c).

Если байт SW1 = '69', а байт SW2 равен:

'81' — команда несовместима со структурой файла;

'82' — состояние защиты неудовлетворительное;

'86' — команда невозможна (нет текущего EF).

Если байт SW1 = '6A', а байт SW2 равен:

'81' — функция не поддерживается;

'82' — файл не найден;

'83' — запись не найдена;

'84' — область памяти в файле недостаточна;

'85' — L_c не согласуется со структурой TLV.

6.7 Команда ПРИСОЕДИНИТЬ ЗАПИСЬ

6.7.1 Определение и область применения

Командное сообщение команды ПРИСОЕДИНИТЬ ЗАПИСЬ инициирует либо присоединение записи в конце EF с линейной структурой, либо запись данных (в виде записи с номером 1) в EF с циклической структурой (см. 5.1.4).

Команда должна устанавливать указатель записи на успешно присоединенную запись.

6.7.2 Условия использования и защиты

Команда может быть выполнена только в том случае, если состояние защиты удовлетворяет атрибутам секретности данного EF для функции присоединения.

Если выбор EF осуществляется одновременно с подачей команды, то эта команда может обрабатываться без идентификации данного файла.

Если команда содержит разрешенный короткий идентификатор EF, она устанавливает указываемый файл в состояние текущего EF, а указатель текущей записи — в исходное положение.

Команда должна прерываться в том случае, если она применяется к EF без структуры из записей.

Примечание — Если данная команда применяется к файлу EF с циклической структурой, заполненному записями, то запись с наибольшим номером заменяется и становится записью номер 1.

6.7.3 Командное сообщение

Командный APDU команды присоединить ЗАПИСЬ представлен в таблице 43.

Таблица 43 — Командный APDU команды ПРИСОЕДИНИТЬ ЗАПИСЬ

CLA	Как определено в 5.4.1
INS	'E2'
P1	Действителен только байт P1 = '00'
P2	См. таблицу 44
Поле L_c	Длина последующего поля данных
Поле данных	Запись, подлежащая присоединению
Поле L_e	Пустое

Таблица 44 — Кодирование байта управления ссылками P2

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
0	0	0	0	0	0	0	0	Выбираемый в текущий момент EF
×	×	×	×	×	0	0	0	Короткий идентификатор EF
(не все равны)								
Любое другое значение								RFU

Если записи являются информационными объектами SIMPLE-TLV (см. 5.4.4), то для этого случая в таблице 45 для наглядности представлен формат поля данных командного сообщения.

Таблица 45 — Поле данных команды

Полное присоединение одной записи

T_n (один байт)	L_n (один или три байта)	Все байты данных записи (L_n байтов)

6.7.4 Ответное сообщение (номинальный случай)

Ответный APDU команды ПРИСОЕДИНИТЬ ЗАПИСЬ представлен в таблице 46.

Таблица 46 — Ответный APDU команды ПРИСОЕДИНИТЬ ЗАПИСЬ

Поле данных	Пустое
SW1, SW2	Байты состояния

6.7.5 Состояния после обработки

Может возникать следующее специфическое состояние предупреждения.

Если байт SW1 = '63', а байт SW2 равен:

'CX' — счетчик (успешное присоединение записи, но после использования внутренней программы повторений; 'X' ≠ '0' указывает число повторных попыток; 'X' = '0' означает, что счетчик не предусмотрен).

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '65', а байт SW2 равен:

'81' — отказ памяти (безуспешное присоединение записи).

Если байт SW1 = '67', а байт SW2 равен:

'00' — неверно указанная длина (пустое поле L_c).

Если байт SW1 = '69', а байт SW2 равен:

'81' — команда несовместима со структурой файла;

'82' — состояние защиты неудовлетворительное;

'86' — команда невозможна (нет текущего EF).

Если байт SW1 = '6A', а байт SW2 равен:

'81' — функция не поддерживается;

'82' — файл не найден;

'84' — область памяти в файле недостаточна;

'85' — L_c не согласуется со структурой TLV.

6.8 Команда ОБНОВИТЬ ЗАПИСЬ

6.8.1 Определение и область применения

Командное сообщение команды ОБНОВИТЬ ЗАПИСЬ инициирует обновление конкретной записи битами, передаваемыми в командном APDU.

При использовании адресации текущей записи команда должна устанавливать указатель записи на успешно обновленную запись.

6.8.2 Условия использования и защиты

Команда может быть выполнена только в том случае, если состояние защиты удовлетворяет атрибутам секретности данного EF для функции обновления.

Если выбор EF осуществляется одновременно с подачей команды, то эта команда может обрабатываться без идентификации данного файла.

Если команда содержит разрешенный короткий идентификатор EF, она устанавливает указываемый файл в состояние текущего EF, а указатель текущей записи — в исходное положение.

Команда должна прерываться в том случае, если она применяется к EF без структуры из записей.

Если команда применяется к EF с линейной фиксированной или циклической структурой, то она должна прерываться в том случае, если длина передаваемой записи отличается от длины существующей записи.

Если команда применяется к EF с линейной переменной структурой, то она может быть выполнена, когда длина передаваемой записи отличается от длины существующей записи.

Опция команды «предыдущая» (P2 = xxxxx011), примененная к циклическому файлу, ведет себя также, как команда ПРИСОЕДИНИТЬ ЗАПИСЬ.

6.8.3 Командное сообщение

Командный APDU команды ОБНОВИТЬ ЗАПИСЬ представлен в таблице 47.

Таблица 47 — Командный APDU команды ОБНОВИТЬ ЗАПИСЬ

CLA	Как определено в 5.4.1
INS	'DC'
P1	P1 = '00' обозначает текущую запись, P1 ≠ '00' — номер указываемой записи
P2	См. таблицу 48
Поле L_c	Длина последующего поля данных
Поле данных	Запись для обновления
Поле L_e	Пустое

Таблица 48 — Кодирование байта управления ссылками P2

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
0	0	0	0	0	—	—	—	Выбираемый в текущий момент EF
×	×	×	×	×	—	—	—	Короткий идентификатор EF
(не все равны)								
—	—	—	—	—	0	0	0	Первая запись
—	—	—	—	—	0	0	1	Последняя запись
—	—	—	—	—	0	1	0	Следующая запись
—	—	—	—	—	0	1	1	Предыдущая запись
—	—	—	—	—	1	0	0	Номер записи дан в байте P1
Любое другое значение								RFU

Если записи являются информационными объектами SIMPLE-TLV (см. 5.4.4), то для этого случая в таблице 49 для наглядности представлен формат поля данных командного сообщения.

Таблица 49 — Поле данных команды

Полное обновление одной записи

T_n (один байт)	L_n (один или три байта)	Все байты данных записи (L_n байтов)

6.8.4 Ответное сообщение (номинальный случай)

Ответный APDU команды ОБНОВИТЬ ЗАПИСЬ представлен в таблице 50.

Таблица 50 — Ответный APDU команды ОБНОВИТЬ ЗАПИСЬ

Поле данных SW1, SW2	Пустое Байты состояния

6.8.5 Состояния после обработки

Может возникать следующее специфическое состояние предупреждения.

Если байт SW1 = '63', а байт SW2 равен:

'CX' — счетчик (успешное обновление, но после использования внутренней программы повторений; 'X' ≠ '0' указывает число повторных попыток; 'X' = '0' означает, что счетчик не предусмотрен).

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '65', а байт SW2 равен:

'81' — отказ памяти (безуспешное обновление).

Если байт SW1 = '67', а байт SW2 равен:

'00' — неверно указанная длина (пустое поле L_c).

Если байт SW1 = '69', а байт SW2 равен:

'81' — команда несовместима со структурой файла;

'82' — состояние защиты неудовлетворительное;

'86' — команда невозможна (нет текущего EF).

Если байт SW1 = '6A', а байт SW2 равен:

'81' — функция не поддерживается;

'82' — файл не найден;

'83' — запись не найдена;

'84' — область памяти в файле недостаточна;

'85' — L_c не согласуется со структурой TLV.

6.9 Команда ИЗВЛЕЧЬ ДАННЫЕ

6.9.1 Определение и область применения

Команда ИЗВЛЕЧЬ ДАННЫЕ применяется для поиска одного простого информационного объекта либо одного или большего числа информационных объектов, входящих в составной информационный объект, в пределах текущего контекста (например, среды, связанной с конкретным приложением, или текущего DF).

6.9.2 Условия использования и защиты

Команда может быть выполнена только в том случае, если состояние защиты удовлетворяет условиям секретности, определяемым приложением в пределах контекста для функции.

6.9.3 Командное сообщение

Командный APDU команды ИЗВЛЕЧЬ ДАННЫЕ представлен в таблице 51.

Таблица 51 — Командный APDU команды ИЗВЛЕЧЬ ДАННЫЕ

CLA	Как определено в 5.4.1
INS	'CA'
P1, P2	См. таблицу 52
Поле L_c	Пустое
Поле данных	«
Поле L_e	Число байтов, ожидаемых в ответе

Таблица 52 — Кодирование последовательности параметров P1—P2

Значение	Смысловое содержание
От '0000' до '003F'	RFU
От '0040' до '00FF'	Тег (один байт) информационного объекта BER-TLV в байте P2
От '0100' до '01FF'	Данные приложения (оригинальное кодирование)
От '0200' до '02FF'	Тег информационного объекта SIMPLE-TLV в байте P2
От '0300' до '3FFF'	RFU
От '4000' до 'FFFF'	Тег (два байта) информационного объекта BER-TLV в байтах P1, P2

а) Извлечение данных приложения

Если значение последовательности байтов P1—P2 находится в диапазоне от '0100' до '01FF', то оно должно являться идентификатором, зарезервированным для внутренних проверок, осуществляемых картой, и оригинальных услуг, значимых в пределах данного контекста приложения.

б) Извлечение информационных объектов

Если значение последовательности байтов P1—P2 находится в диапазоне от '0040' до '00FF', то значение байта P2 должно являться тегом информационного объекта BER-TLV, состоящим из одного байта. Значение '00FF' зарезервировано для получения всех общих информационных объектов BER-TLV, читаемых в контексте.

Если значение последовательности байтов P1—P2 находится в диапазоне от '0200' до '02FF', то значение байта P2 должно являться тегом информационного объекта SIMPLE-TLV. Значение '0200' является RFU. Значение '02FF' зарезервировано для получения всех общих информационных объектов SIMPLE-TLV, читаемых в контексте.

Если значение последовательности байтов P1—P2 находится в диапазоне от '4000' до 'FFFF', то оно должно являться тегом информационного объекта BER-TLV, состоящим из двух байтов. Значения '4000' и 'FFFF' являются RFU.

Если запрашивается простой информационный объект, поле данных ответного сообщения должно содержать значение соответствующего простого информационного объекта.

Если запрашивается составной информационный объект, поле данных ответного сообщения должно содержать значение составного информационного объекта, т.е. информационные объекты, включая их тег, длину и значение.

6.9.4 Ответное сообщение (номинальный случай)

Если поле L_e содержит только нули, то в пределах максимума 256 (для короткого поля) или 65536 (для расширенного поля) вся запрашиваемая информация должна быть выдана.

Ответный APDU команды ИЗВЛЕЧЬ ДАННЫЕ представлен в таблице 53.

Таблица 53 — Ответный APDU команды ИЗВЛЕЧЬ ДАННЫЕ

Поле данных	L_r байтов (L_r может быть равно L_e)
SW1, SW2	Байты состояния

6.9.5 Состояния после обработки

Может возникать следующее специфическое состояние предупреждения.

Если байт SW1 = '62', а байт SW2 равен:

'81' — часть выдаваемых данных может быть искажена.

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '67', а байт SW2 равен:

'00' — неверно указанная длина (пустое поле L_e).

Если байт SW1 = '69' , а байт SW2 равен:

'82' — состояние защиты неудовлетворительное;

'85' — условия использования не удовлетворены.

Если байт SW1 = '6A', а байт SW2 равен:

'81' — функция не поддерживается;

'88' — ссылочные данные (информационные объекты) не найдены.

Если байт SW1 = '6C', а байт SW2 равен:

'XX' — неверно указанная длина (несоответствующее поле L_e; 'XX' указывает точную длину).

6.10 Команда ПОМЕСТИТЬ ДАННЫЕ

6.10.1 Определение и область применения

Команда ПОМЕСТИТЬ ДАННЫЕ применяется для сохранения одного простого информационного объекта либо одного или большего числа информационных объектов, входящих в составной информационный объект, в пределах текущего контекста (например среды, связанной с конкретным приложением, или текущего DF). Нужные функции сохранения (однократной записи и/или обновления и/или присоединения) должны вызываться по определению или в зависимости от характера информационных объектов.

Примечание — Эта команда могла бы использоваться, например, для обновления информационных объектов.

6.10.2 Условия использования и защиты

Команда может быть выполнена только в том случае, если состояние защиты удовлетворяет условиям секретности, определяемым приложением в пределах контекста для функции(й).

6.10.3 Командное сообщение

Ответный APDU команды ПОМЕСТИТЬ ДАННЫЕ представлен в таблице 54.

Таблица 54 — Командный APDU команды ПОМЕСТИТЬ ДАННЫЕ

CLA	Как определено в 5.4.1
INS	'DA'
P1, P2	См. таблицу 55
Поле L_c	Длина последующего поля данных
Поле данных	Параметры и данные, подлежащие записи
Поле L_e	Пустое

Таблица 55 — Кодирование последовательности параметров P1—P2

Значение	Смысловое содержание
От '0000' до '003F'	RFU
От '0040' до '00FF'	Тег (один байт) информационного объекта BER-TLV в байте P2
От '0100' до '01FF'	Данные приложения (оригинальное кодирование)
От '0200' до '02FF'	Тег информационного объекта SIMPLE-TLV в байте P2
От '0300' до '3FFF'	RFU
От '4000' до 'FFFF'	Тег (два байта) информационного объекта BER-TLV в байтах P1, P2

а) Сохранение данных приложения

Если значение последовательности байтов P1—P2 находится в диапазоне от '0100' до '01FF', то оно должно являться идентификатором, зарезервированным для внутренних проверок, осуществляемых картой, и оригинальных услуг, значимых в пределах данного контекста приложения.

б) Сохранение информационных объектов

Если значение последовательности байтов P1—P2 находится в диапазоне от '0040' до '00FF', то значение байта P2 должно являться тегом информационного объекта BER-TLV, состоящим из одного байта. Значение '00FF' зарезервировано для указания, что поле данных несет информационные объекты BER-TLV.

Если значение последовательности байтов P1—P2 находится в диапазоне от '0200' до '02FF', то значение байта P2 должно являться тегом информационного объекта SIMPLE-TLV. Значение '0200' является RFU. Значение '02FF' зарезервировано для указания, что поле данных несет информационные объекты SIMPLE-TLV.

Если значение последовательности байтов P1—P2 находится в диапазоне от '4000' до 'FFFF', то оно должно являться тегом информационного объекта BER-TLV, состоящим их двух байтов. Значения '4000' и 'FFFF' являются RFU.

Если предоставляется простой информационный объект, поле данных командного сообщения должно содержать значение соответствующего простого информационного объекта.

Если предоставляется составной информационный объект, поле данных командного сообщения должно содержать значение составного информационного объекта, т.е. информационные объекты, включая их тег, длину и значение.

6.10.4 Ответное сообщение (номинальный случай)

Ответный APDU команды ПОМЕСТИТЬ ДАННЫЕ представлен в таблице 56.

Таблица 56 — Ответный APDU команды ПОМЕСТИТЬ ДАННЫЕ

Поле данных	Пустое
SW1, SW2	Байты состояния

6.10.5 Состояния после обработки

Может возникать следующее специфическое состояние предупреждения.

Если байт SW1 = '63', а байт SW2 равен:

'CX' — счетчик (успешное сохранение, но после использования внутренней программы повторений; 'X' ≠ '0' указывает число повторных попыток; 'X' = '0' означает, что счетчик не предусмотрен).

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '65', а байт SW2 равен:

'81' — отказ памяти (безуспешное сохранение).

Если байт SW1 = '67', а байт SW2 равен:

'00' — неверно указанная длина (несоответствующее поле L_c).

Если байт SW1 = '69', а байт SW2 равен:

'82' — состояние защиты неудовлетворительное;

'85' — условия использования не удовлетворены.

Если байт SW1 = '6A', а байт SW2 равен:

'80' — некорректные параметры в поле данных;

'81' — функция не поддерживается;

'84' — область памяти в файле недостаточна;

'85' — L_c не согласуется со структурой TLV.

6.11 Команда ВЫБРАТЬ ФАЙЛ

6.11.1 Определение и область применения

Успешно завершенная команда ВЫБРАТЬ ФАЙЛ устанавливает текущий файл внутри логического канала (см. 5.5). Последующие команды могут в неявной форме обращаться к текущему файлу через этот логический канал.

Выбор DF (который может быть файлом MF) устанавливает его в состояние текущего DF. После такого выбора обращение к неявному текущему EF может осуществляться через этот логический канал.

Выбор EF устанавливает пару текущих файлов: EF и его родительский файл.

После ответа на восстановление осуществляется неявный выбор файла MF через основной логический канал (см. 5.5.2), если отсутствуют другие указания в байтах предыстории (см. раздел 8) или строке начальных данных (см. раздел 9).

Примечание — Прямой выбор по имени DF может использоваться для выбора приложений, зарегистрированных в соответствии с ИСО/МЭК 7816-5.

6.11.2 Условия использования и защиты

Следующие условия должны соблюдаться для каждого открытого логического канала.

Если не дается иных указаний, то правильное выполнение команды изменяет состояние защиты (см. 5.2.1) по следующим правилам.

Если заменяется текущий EF, или текущего EF больше нет, состояние защиты, если оно применялось, ориентированное на предшествующий текущий EF, утрачивается.

Если текущий DF является потомком предшествующего текущего DF или ему идентичен, состояние защиты, ориентированное на предшествующий текущий DF, сохраняется.

Если текущий DF не является потомком предшествующего текущего DF и не идентичен ему,

состояние защиты, ориентированное на предшествующий текущий DF, утрачивается. Состояние защиты, общее для всех общих предков предшествующего и нового текущего DF, сохраняется.

6.11.3 Командное сообщение

Командный APDU команды ВЫБРАТЬ ФАЙЛ представлен в таблице 57.

Таблица 57 — Командный APDU команды ВЫБРАТЬ ФАЙЛ

CLA	Как определено в 5.4.1
INS	'A4'
P1	Управление выбором, см. таблицу 58
P2	Опции выбора, см. таблицу 59
Поле L_c	Пустое или длина последующего поля данных
Поле данных	Если представлено, то в соответствии с байтами P1, P2: - идентификатор файла - путь из MF - путь из текущего DF - имя DF
Поле L_e	Пустое или максимальная длина данных, ожидаемых в ответе

Таблица 58 — Кодирование байта управления выбором P1

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
0	0	0	0	0	0	×	×	Выбор посредством идентификатора файла:
0	0	0	0	0	0	0	0	выбирать MF, DF или EF (поле данных — идентификатор или пустое)
0	0	0	0	0	0	0	1	выбирать дочерний DF (поле данных — идентификатор DF)
0	0	0	0	0	0	1	0	выбирать EF под текущим DF (поле данных — идентификатор EF)
0	0	0	0	0	0	1	1	выбирать родительский DF текущего DF (пустое поле данных)
0	0	0	0	0	1	×	×	Выбор по имени DF:
0	0	0	0	0	1	0	0	прямой выбор (поле данных — имя DF)
0	0	0	0	0	1	0	1	RFU
0	0	0	0	0	1	1	0	RFU
0	0	0	0	0	1	1	1	RFU
0	0	0	0	1	0	×	×	Выбор через путь (см. 5.1.2):
0	0	0	0	1	0	0	0	выбирать из MF (поле данных — путь без идентификатора MF)
0	0	0	0	1	0	0	1	выбирать из текущего DF (поле данных — путь без идентификатора текущего DF)
0	0	0	0	1	0	1	0	RFU
0	0	0	0	1	0	1	1	RFU
Любое другое значение								RFU

Если байт P1 = '00', то карта распознает либо благодаря особому кодированию идентификатора файла, либо из-за контекста выполнения команды, является ли выбираемый файл файлом MF, DF или EF.

Если последовательность байтов P1—P2 = '0000' и предоставляется идентификатор файла, то он должен быть уникальным в следующих средах:

- непосредственные потомки текущего DF,

- родительский DF,
- непосредственные потомки родительского DF.

Если последовательность байтов P1—P2 = '0000' и поле данных пустое или равно '3F00', то осуществляется выбор MF.

Если байт P1 = '04', то поле данных представляет собой имя DF, возможно укороченное справа. Если выбор по укороченному имени DF поддерживается, то следующие одна за другой такие команды с одним и тем же полем данных должны выбирать файлы DF, чьи имена совпадают с их полем данных, т.е. с него начинаются. Если карта принимает команду ВЫБРАТЬ ФАЙЛ с пустым полем данных, то может осуществляться последовательный выбор всех файлов DF или их подмножества.

Примечание — О методах выбора, поддерживаемых картой, см. 8.3.6.

Таблица 59 — Кодирование байта опций выбора P2

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
0	0	0	0	—	—	0	0	Первое или единственное вхождение
0	0	0	0	—	—	0	1	Последнее вхождение
0	0	0	0	—	—	1	0	Следующее вхождение
0	0	0	0	—	—	1	1	Предыдущее вхождение
0	0	0	0	×	×	—	—	Опция контрольной информации файла (см. 5.1.5):
0	0	0	0	0	0	—	—	выдать FCI, необязательный шаблон
0	0	0	0	0	1	—	—	выдать шаблон FCP
0	0	0	0	1	0	—	—	выдать шаблон FMD
Любое другое значение								RFU

6.11.4 Ответное сообщение (номинальный случай)

Если поле L_e содержит только нули, то в пределах максимума 256 (для короткого поля) или 65536 (для расширенного поля) все байты, соответствующие опции выбора, должны быть выданы.

Ответный APDU команды ВЫБРАТЬ ФАЙЛ представлен в таблице 60.

Таблица 60 — Ответный APDU команды ВЫБРАТЬ ФАЙЛ

Поле данных SW1, SW2	Информация, соответствующая байту P2 (не более L_e байтов) Байты состояния

6.11.5 Состояния после обработки

Могут возникать следующие специфические состояния предупреждения.

Если байт SW1 = '62', а байт SW2 равен:

'83' — выбранный файл недействителен;

'84' — FCI не форматирована по 5.1.5.

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '6A', а байт SW2 равен:

'81' — функция не поддерживается;

'82' — файл не найден;

'86' — некорректные параметры P1, P2;

'87' — L_c не согласуется с P1, P2.

6.12 Команда ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ

6.12.1 Определение и область применения

Команда ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ инициирует сравнение в карте верификационных данных, посылаемых с устройства сопряжения, с контрольными данными, хранимыми в карте (например паролем).

6.12.2 Условия использования и защиты

В результате сравнения состояние защиты может изменяться. Безуспешные операции сравне-

ния могут регистрироваться в карте (например, с целью ограничения числа дальнейших попыток использования контрольных данных).

6.12.3 Командное сообщение

Командный APDU команды ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ представлен в таблице 61.

Таблица 61 — Командный APDU команды ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ

CLA	Как определено в 5.4.1
INS	'20'
P1	'00' (другие значения являются RFU)
P2	Квалификатор контрольных данных, см. таблицу 62
Поле L_c	Пустое или длина последующего поля данных
Поле данных	Пустое или верификационные данные
Поле L_e	Пустое

Таблица 62 — Кодирование байта управления ссылками P2

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
0	0	0	0	0	0	0	0	Информация не предоставлена
0	—	—	—	—	—	—	—	Глобальные контрольные данные (например, пароль карты)
1	—	—	—	—	—	—	—	Специфические контрольные данные (например, собственный пароль файла DF)
—	×	×	—	—	—	—	—	00 (другие значения являются RFU)
—	—	—	×	×	×	×	×	Номер контрольных данных

Примечания

1 В картах, где команда ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ однозначно обращается к секретным данным, байт P2 = '00' зарезервирован для указания, что никакой детальный квалификатор не используется.

2 Номер контрольных данных может представлять собой, например, номер пароля или короткий идентификатор EF.

3 Если тело является пустым, команда может использоваться либо для получения числа дальнейших разрешенных попыток 'X' (последовательность SW1—SW2 = '63CX'), либо для проверки того, что верификация не требуется (последовательность SW1—SW2 = '9000').

6.12.4 Ответное сообщение (номинальный случай)

Ответный APDU команды ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ представлен в таблице 63.

Таблица 63 — Ответный APDU команды ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ

Поле данных	Пустое
SW1, SW2	Байты состояния

6.12.5 Состояния после обработки

Могут возникать следующие специфические состояния предупреждения.

Если байт SW1 = '63', а байт SW2 равен:

'00' — информация не предоставлена (неудавшаяся верификация);

'CX' — счетчик (неудавшаяся верификация; 'X' указывает число дальнейших разрешенных попыток).

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '69', а байт SW2 равен:

'83' — метод аутентификации заблокирован;

'84' — ссылочные данные недействительны.

Если байт SW1 = '6A', а байт SW2 равен:

'86' — некорректные параметры P1, P2;

'88' — ссылочные данные не найдены.

6.13 Команда ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ

6.13.1 Определение и область применения

Команда ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ инициирует вычисление картой аутентификационных данных с использованием данных задачи, посылаемой с устройства сопряжения, и соответствующего секрета (например ключа), хранящегося в карте.

Если соответствующий секрет присоединен к MF, то команда может применяться для аутентификации карты в целом.

Если соответствующий секрет присоединен к другому DF, команда может применяться для аутентификации этого DF.

6.13.2 Условия использования и защиты

Успешное выполнение команды может подчиняться успешному завершению предшествующих команд (например, команд ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ, ВЫБРАТЬ ФАЙЛ) или выбору (например, соответствующего секрета).

Если выбор ключа и алгоритма осуществляется одновременно с подачей команды, то команда может неявно использовать этот ключ и этот алгоритм.

Число подач команды может регистрироваться в карте для ограничения числа дальнейших попыток использования соответствующего секрета или алгоритма.

6.13.3 Командное сообщение

Командный APDU команды ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ представлен в таблице 64.

Таблица 64 — Командный APDU команды ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ

CLA	Как определено в 5.4.1
INS	'88'
P1	Указатель алгоритма в карте
P2	Указатель секрета, см. таблицу 65
Поле L_c	Длина последующего поля данных
Поле данных	Данные, относящиеся к аутентификации (например, задача)
Поле L_e	Максимальное число байтов, ожидаемых в ответе

Байт P1 = '00' указывает, что информация не предоставлена. Указатель алгоритма либо известен до подачи команды, либо содержится в поле данных.

Байт P2 = '00' указывает, что информация не предоставлена. Указатель секрета либо известен до подачи команды, либо содержится в поле данных.

Таблица 65 — Кодирование байта управления ссылками P2

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
0	0	0	0	0	0	0	0	Информация не предоставлена
0	—	—	—	—	—	—	—	Глобальные контрольные данные (например, собственный ключ файла MF)
1	—	—	—	—	—	—	—	Специфические контрольные данные (например, собственный ключ файла DF)
—	×	×	—	—	—	—	—	00 (другие значения являются RFU)
—	—	—	×	×	×	×	×	Номер секрета

Примечание — Номер секрета может представлять собой, например, номер ключа или короткий идентификатор EF.

6.13.4 Ответное сообщение (номинальный случай)

Ответный APDU команды ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ представлен в таблице 66.

Таблица 66 — Ответный APDU команды ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ

Поле данных	Данные, относящиеся к аутентификации (например, ответ на задачу)
SW1, SW2	Байты состояния

Примечание — Ответное сообщение может включать в себя данные, пригодные для использования дальнейшими защитными функциями приложений (например, случайное число).

6.13.5 Состояния после обработки

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '69', а байт SW2 равен:

'84' — ссылочные данные недействительны;

'85' — условия использования не удовлетворены.

Если байт SW1 = '6A', а байт SW2 равен:

'86' — некорректные параметры P1, P2;

'88' — ссылочные данные не найдены.

6.14 Команда ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ

6.14.1 Определение и область применения

Команда ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ условно обновляет состояние защиты, используя результат (да или нет) вычисления, осуществляемого картой, основанного на данных задачи, ранее выданной картой (например, на команду СОЗДАТЬ ЗАДАЧУ), ключе, возможно секретном, хранящемся в карте, и аутентификационных данных, передаваемых устройством сопряжения.

6.14.2 Условия использования и защиты

Успешное выполнение команды требует, чтобы данные последней задачи, полученной с карты, были действительны.

Безуспешные операции сравнения могут регистрироваться в карте (например, с целью ограничения числа дальнейших попыток использования контрольных данных).

6.14.3 Командное сообщение

Командный APDU команды ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ представлен в таблице 67.

Таблица 67 — Командный APDU команды ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ

CLA	Как определено в 5.4.1
INS	'82'
P1	Указатель алгоритма в карте
P2	Указатель секрета, см. таблицу 68
Поле L_c	Пустое или длина последующего поля данных
Поле данных	Пустое или данные, относящиеся к аутентификации (например, ответ на задачу)
Поле L_e	Пустое

Байт P1 = '00' указывает, что информация не предоставлена. Указатель алгоритма либо известен до подачи команды, либо содержится в поле данных.

Байт P2 = '00' указывает, что информация не предоставлена. Указатель секрета либо известен до подачи команды, либо содержится в поле данных.

Таблица 68 — Кодирование байта управления ссылками P2

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
0	0	0	0	0	0	0	0	Информация не предоставлена
0	—	—	—	—	—	—	—	Глобальные контрольные данные (например, собственный ключ файла MF)
1	—	—	—	—	—	—	—	Специфические контрольные данные (например, собственный ключ файла DF)
—	×	×	—	—	—	—	—	00 (другие значения являются RFU)
—	—	—	×	×	×	×	×	Номер секрета

Примечания

1 Номер секрета может представлять собой, например, номер ключа или короткий идентификатор EF.

2 Если тело является пустым, команда может использоваться либо для получения числа дальнейших разрешенных попыток 'X' (последовательность SW1—SW2 = '63CX'), либо для проверки того, что верификация не требуется (последовательность SW1—SW2 = '9000').

6.14.4 Ответное сообщение (номинальный случай)

Ответный APDU команды ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ представлен в таблице 69.

Таблица 69 — Ответный APDU команды ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ

Поле данных SW1, SW2	Пустое Байты состояния

6.14.5 Состояния после обработки

Могут возникать следующие специфические состояния предупреждения.

Если байт SW1 = '63', а байт SW2 равен:

'00' — информация не предоставлена (неудавшаяся аутентификация);

'CX' — счетчик (неудавшаяся аутентификация; 'X' указывает число дальнейших разрешенных попыток).

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '67', а байт SW2 равен:

'00' — неверно указанная длина (поле L_c некорректное);

Если байт SW1 = '69', а байт SW2 равен:

'83' — метод аутентификации заблокирован;

'84' — ссылочные данные недействительны;

'85' — условия использования не удовлетворены (команда невозможна в контексте).

Если байт SW1 = '6A', а байт SW2 равен:

'86' — некорректные параметры P1, P2;

'88' — ссылочные данные не найдены.

6.15 Команда СОЗДАТЬ ЗАДАЧУ

6.15.1 Определение и область применения

Команда СОЗДАТЬ ЗАДАЧУ требует выдачи задачи (например, в виде случайного числа) для использования в процедуре, связанной с защитой (например, в команде ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ).

6.15.2 Условия использования и защиты

Данные задачи действительны, по меньшей мере, для следующей команды. Никаких дополнительных условий настоящий стандарт не устанавливает.

6.15.3 Командное сообщение

Командный APDU команды СОЗДАТЬ ЗАДАЧУ представлен в таблице 70.

Таблица 70 — Командный APDU команды СОЗДАТЬ ЗАДАЧУ

CLA	Как определено в 5.4.1
INS	'84'
P1-P2	'0000' (другие значения являются RFU)
Поле L_c	Пустое
Поле данных	«
Поле L_e	Максимальная длина ожидаемого ответа

6.15.4 Ответное сообщение (номинальный случай)

Ответный APDU команды СОЗДАТЬ ЗАДАЧУ представлен в таблице 71.

Таблица 71 — Ответный APDU команды СОЗДАТЬ ЗАДАЧУ

Поле данных SW1, SW2	Задача Байты состояния

6.15.5 Состояния после обработки

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '6A', а байт SW2 равен:

'81' — функция не поддерживается;

'86' — некорректные параметры P1, P2.

6.16 Команда ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ

6.16.1 Определение и область применения

Команда ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ открывает и закрывает логические каналы.

Функция открытия открывает новый логический канал, отличный от основного канала. Предусматриваются опции для назначения номера логического канала картой либо для сообщения карте назначенного извне номера логического канала.

Функция закрытия в явной форме закрывает логический канал, отличный от основного канала. После успешного закрытия логический канал должен быть доступен для повторного использования.

6.16.2 Условия использования и защиты

Если функция открытия приводится в действие с основного логического канала, то после успешного открытия должен осуществляться неявный выбор файла MF как текущего файла DF, а состояние защиты для нового логического канала должно быть таким же, что и для основного логического канала после ATR. Состояние защиты нового логического канала должно быть отдельным от состояния защиты любого другого логического канала.

Если функция открытия приводится в действие с логического канала, не являющегося основным, то после успешного открытия должен осуществляться выбор текущего файла DF логического канала, с которого была подана команда, как текущего файла DF нового логического канала; состояние защиты для нового логического канала должно быть таким же, что и для логического канала, с которого функция открытия была приведена в действие.

После успешного осуществления функции закрытия состояние защиты, связанное с соответствующим логическим каналом, утрачивается.

6.16.3 Командное сообщение

Командный APDU команды ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ представлен в таблице 72.

Таблица 72 — Командный APDU команды ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ

CLA	Как определено в 5.4.1
INS	'70'
P1	P1 = '00' — открыть логический канал
	P1 = '80' — закрыть логический канал (другие значения являются RFU)
P2	'00', '01', '02' или '03' (другие значения являются RFU)
Поле L_c	Пустое
Поле данных	« »
Поле L_e	'01', если последовательность байтов P1—P2 = '0000'
	Пустое, если последовательность байтов P1—P2 ≠ '0000'

Бит b8 байта P1 используется для индикации функций открытия и закрытия; если бит b8 установлен в ноль, то команда ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ должна открывать логический канал; если бит b8 установлен в единицу, то команда ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ должна закрывать логический канал.

Для функции открытия (P1 = '00') биты b1 и b2 в байте P2 используются для кодирования номера логического канала таким же образом, как и в байте класса (см. 5.4.1); другие биты байта P2 являются RFU.

Если биты b1 и b2 байта P2 являются нулевыми, то карта назначает номер логического канала, выдаваемый в битах b1 и b2 поля данных ответного сообщения.

Если бит b1 и/или бит b2 байта P2 не являются нулевыми, то они кодируют номер логического канала, отличный от номера основного логического канала; тогда карта открывает назначенный извне номер логического канала.

6.16.4 Ответное сообщение (номинальный случай)

Ответный APDU команды ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ представлен в таблице 73.

Таблица 73 — Ответный APDU команды ВВЕСТИ УПРАВЛЕНИЕ КАНАЛОМ

Поле данных	Номер логического канала, если последовательность байтов P1—P2 = '0000'
	Пустое, если последовательность байтов P1—P2 ≠ '0000'
SW1, SW2	Байты состояния

6.16.5 Состояния после обработки

Может возникать следующее специфическое состояние предупреждения.

Если байт SW1 = '62', а байт SW2 равен:

'00' — информация не предоставлена.

7 Межотраслевые команды, ориентированные на передачу данных

Поддержка всех описываемых команд или всех опций поддерживаемых команд не является обязательной для всех карт, соответствующих настоящему стандарту.

Когда требуется участие в международном информационном обмене, набор из системных услуг, предоставляемых картой, и связанных с ними команд и опций должен использоваться, как определено в разделе 9.

В таблице 11 представлен полный перечень команд, определяемых в настоящем стандарте.

Настоящий раздел не предусматривает описаний влияния безопасного обмена сообщениями (см. 5.6) на структуры сообщений.

Перечни состояний ошибки и состояний предупреждения, приводимые в пятом пункте каждого подраздела настоящего раздела, не являются исчерпывающими (см. 5.4.5).

7.1 Команда ИЗВЛЕЧЬ ОТВЕТ

7.1.1 Определение и область применения

Команда ИЗВЛЕЧЬ ОТВЕТ используется для передачи с карты устройству сопряжения APDU (или его части), который по-другому не может быть передан при помощи доступных протоколов.

7.1.2 Условия использования и защиты

Не установлены.

7.1.3 Командное сообщение

Командный APDU команды ИЗВЛЕЧЬ ОТВЕТ представлен в таблице 74.

Таблица 74 — Командный APDU команды ИЗВЛЕЧЬ ОТВЕТ

CLA	Как определено в 5.4.1
INS	'C0'
P1—P2	'0000' (другие значения являются RFU)
Поле L_c	Пустое
Поле данных	—
Поле L_e	Максимальная длина данных, ожидаемых в ответе

7.1.4 Ответное сообщение (номинальный случай)

Если поле L_e содержит только нули, то в пределах максимума 256 (для короткого поля) или 65536 (для расширенного поля) все имеющиеся байты должны быть выданы.

Ответный APDU команды ИЗВЛЕЧЬ ОТВЕТ представлен в таблице 75.

Таблица 75 — Ответный APDU команды ИЗВЛЕЧЬ ОТВЕТ

Поле данных	APDU (его часть), соответствующий L_e
SW1, SW2	Байты состояния

7.1.5 Состояния после обработки

Может возникать следующее специфическое состояние, соответствующее нормальной обработке.

Если байт SW1 = '61', а байт SW2 равен:

'XX' — нормальная обработка: имеется больше байтов данных ('XX' указывает число дополнительных имеющихся байтов данных, доступных для последующей команды ИЗВЛЕЧЬ ОТВЕТ).

Может возникать следующее специфическое состояние предупреждения.

Если байт SW1 = '62', а байт SW2 равен:

'81' — часть выдаваемых данных может быть искажена.

Могут возникать следующие специфические состояния ошибки.

Если байт SW1 = '67', а байт SW2 равен:

'00' — неверно указанная длина (некорректное поле L_c).

Если байт SW1 = '6A', а байт SW2 равен:

'86' — некорректные параметры P1, P2.

Если байт SW1 = '6C', а байт SW2 равен:

'XX' — неверно указанная длина (несоответствующее поле L_c; 'XX' указывает точную длину).

7.2 Команда КОНВЕРТ

7.2.1 Определение и область применения

Команда КОНВЕРТ используется для передачи APDU или части APDU, либо любой строки данных, которая по-другому не может быть передана при помощи доступных протоколов.

Примечание — Использование команды КОНВЕРТ для SM представлено в приложении Е.

7.2.2 Условия использования и защиты

Не установлены.

7.2.3 Командное сообщение

Командный APDU команды КОНВЕРТ представлен в таблице 76.

Таблица 76 — Командный APDU команды КОНВЕРТ

CLA	Как определено в 5.4.1
INS	'C2'
P1—P2	'0000' (другие значения являются RFU)
Поле L_c	Длина последующего поля данных
Поле данных	APDU (его часть)
Поле L_e	Пустое или длина ожидаемых данных

Если команда КОНВЕРТ используется при T=0 (см. ИСО/МЭК 7816-3) для передачи строк данных, пустое поле данных в ее командном APDU означает «конец строки данных».

7.2.4 Ответное сообщение (номинальный случай)

Ответный APDU команды КОНВЕРТ представлен в таблице 77.

Таблица 77 — Ответный APDU команды КОНВЕРТ

Поле данных	Пустое или APDU (его часть), соответствующий L_e
SW1, SW2	Байты состояния

Примечание — Байты состояния в таблице 77 принадлежат команде КОНВЕРТ. Байты состояния команды, передаваемой в поле данных команды КОНВЕРТ, могут находиться в поле данных ответа КОНВЕРТ.

7.2.5 Состояния после обработки

Может возникать следующее специфическое состояние ошибки.

Если байт SW1 = '67', а байт SW2 равен:

'00' — неверно указанная длина (некорректное поле L_c).

8 Байты предыстории

8.1 Назначение и общая структура

Байты предыстории сообщают внешнему окружению, как использовать карту, если транспортный протокол установлен в соответствии с ИСО/МЭК 7816-3.

Число байтов предыстории (не более 15) указывается и кодируется, как определено в ИСО/МЭК 7816-3.

Информация, переносимая байтами предыстории, также может находиться в файле ATR (идентификатор EF, присваиваемый по умолчанию, равен '2F01').

Если байты предыстории представлены, то они образуют следующие три поля:

- обязательный индикатор категории (один байт),
- необязательные информационные объекты COMPACT-TLV,
- условный индикатор состояния (три байта).

8.2 Индикатор категории (обязательный)

Индикатор категории представляет собой первый байт предыстории. Если он равен '00', '10' или '8X', то формат байтов предыстории должен соответствовать настоящему стандарту.

Кодирование индикатора категории представлено в таблице 78.

Таблица 78 — Кодирование индикатора категории

Значение	Смысловое содержание
'00'	Информация о состоянии должна быть представлена в конце байтов предыстории (не в структуре TLV)
'10'	Указано в 8.5
'80'	Информация о состоянии, если она представлена, содержится в необязательном информационном объекте COMPACT-TLV
От '81' до '8F'	RFU
Другие значения	Оригинальное

8.3 Необязательные информационные объекты COMPACT-TLV

Кодирование информационных объектов COMPACT-TLV выведено из базовых правил кодирования ACH.1 (см. ГОСТ Р ИСО/МЭК 8825 и приложение Г) для информационных объектов BER-TLV с тегом, равным '4X', и длиной, равной '0Y'. Кодировка таких информационных объектов заменяется на байт, равный 'XY', сопровождаемый 'Y' байтами данных. В настоящем разделе 'X' именуется номером тега, а 'Y' — длиной.

Помимо информационных объектов, определяемых в настоящем разделе, байты предыстории могут содержать информационные объекты, определяемые в ИСО/МЭК 7816-5. В этом случае кодирование тегов и полей длины, установленное в ИСО/МЭК 7816-5, должно быть видоизменено, как указано выше.

Если информационные объекты COMPACT-TLV, определяемые в настоящем разделе, появляются в файле ATR, они должны быть закодированы в соответствии с базовыми правилами кодирования ACH.1 (т.е. тег должен быть равен '4X', длина должна быть равна '0Y').

Все теги прикладного класса, не определяемые в настоящем стандарте и стандартах серии ГОСТ Р ИСО/МЭК 7816 (ИСО/МЭК 7816), зарезервированы для ИСО.

8.3.1 Указатель страны/эмитента

Если этот информационный объект представлен, он указывает страну или эмитента.

Данный информационный объект вводится при помощи байта, равного '1Y' или '2Y'.

Кодирование указателя страны/эмитента представлено в таблице 79.

Таблица 79 — Кодирование указателя страны/эмитента

Тег	Длина	Значение
'1'	Переменная	Код страны и национальные данные
'2'	»	Идентификационный номер эмитента

Тег '1' сопровождается соответствующей длиной (в один полубайт) и тремя цифрами, обозначающими страну в соответствии с ОК 025. Последующие данные (нечетное число полубайтов) выбираются соответствующим национальным органом по стандартизации.

Тег '2' сопровождается соответствующей длиной (в один полубайт) и идентификационным номером эмитента по ИСО/МЭК 7812-1. Если идентификационный номер эмитента содержит нечетное число цифр, то он должен быть дополнен справа полубайтом со значением 'F'.

8.3.2 Данные об услугах, предоставляемых картой

Этот информационный объект указывает методы, предусмотренные в карте для поддержки услуг, описываемых в разделе 9.

Он вводится при помощи байта, равного '31'.

Если данный информационный объект не представлен, карта поддерживает лишь неявный выбор приложений.

Кодирование данных об услугах, предоставляемых картой, представлено в таблице 80.

Таблица 80 — Профиль карты, отражающий услуги, не зависимые от приложений

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
1	—	—	—	—	—	—	—	Прямой выбор приложения по полному имени DF
—	1	—	—	—	—	—	—	Выбор по частичному имени DF (см. 9.3.2)
								Информационные объекты доступны:
—	—	1	—	—	—	—	—	- в файле DIR
—	—	—	1	—	—	—	—	- в файле ATR
								Услуги по вводу-выводу содержимого файлов посредством:
—	—	—	—	1	—	—	—	- команды СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ
—	—	—	—	0	—	—	—	- команды СЧИТАТЬ ЗАПИСЬ(И)
—	—	—	—	—	×	×	×	000 (другие значения являются RFU)

Примечание — Информацию о методах выбора может давать содержимое файлов DIR и ATR.

8.3.3 Исходные данные доступа

Этот необязательный информационный объект предоставляет возможности для поиска строки из информационных объектов, определяемых в настоящем стандарте и стандартах серии ГОСТ Р ИСО/МЭК 7816 (ИСО/МЭК 7816). Строка, извлекаемая при помощи данного информационного объекта, называется строкой начальных данных.

Данный информационный объект вводится при помощи байта, равного '41', '42' или '45'.

Любой командный APDU, описываемый в настоящем разделе, предполагается в качестве первой команды, посылаемой после ответа на восстановление. Поэтому информация, доступная в указанный момент, позже может быть не извлекаема.

8.3.3.1 Длина равна '1'

Если предоставляется только один байт информации, он указывает длину в команде на выполнение для извлечения строки начальных данных. Командой на выполнение является команда СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ, структура которой представлена в таблице 81.

Таблица 81 — Кодирование команды, если длина равна '1'

CLA	'00' (см. 5.4.1)
INS	'B0'
P1—P2	'0000'
Поле L_c	Пустое
Поле данных	»
Поле L_e	Первый и единственный байт поля значения исходных данных доступа (указывающий число байтов, подлежащих считыванию)

8.3.3.2 Длина равна '2'

Если предоставляются два байта информации, то первый байт указывает структуру (прозрачная или из записей) и короткий идентификатор файла EF, подлежащего считыванию. Второй байт указывает длину в команде на выполнение СЧИТАТЬ для извлечения строки начальных данных.

Структура первого байта представлена в таблице 82.

Таблица 82 — Структура первого байта

b8	0 — файл, ориентированный на записи 1 — прозрачный файл
b7—b6	00 (другие значения являются RFU)
b5—b1	Короткий идентификатор EF

Если бит b8=0, то командой на выполнение является команда СЧИТАТЬ ЗАПИСЬ(И), структура которой представлена в таблице 83.

Таблица 83 — Кодирование команды, если b8=0

CLA	'00' (см. 5.4.1)
INS	'B2'
P1	'01'
P2	Короткий идентификатор EF (из первого байта исходных данных доступа), сопровождаемый последовательностью битов b3—b2—b1 = 110
Поле L_c	Пустое
Поле данных	»
Поле L_e	Второй и последний байт поля значения исходных данных доступа (указывающий число байтов, подлежащих считыванию)

Если бит b8=1, то командой на выполнение является команда СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ, структура которой представлена в таблице 84.

Таблица 84 — Кодирование команды, если b8 = 1

CLA	'00' (см. 5.4.1)
INS	'B0'
P1	Значение первого байта исходных данных доступа
P2	'00'
Поле L_c	Пустое
Поле данных	»
Поле L_e	Второй и последний байт поля значения исходных данных доступа (указывающий число байтов, подлежащих считыванию)

8.3.3.3 Длина равна '5'

Значение, содержащееся в информационном объекте «исходные данные доступа», состоит из командного APDU команды на выполнение. При выполнении эта команда обеспечивает получение строки начальных данных в поле данных ответа.

8.3.4 Данные эмитента

Этот информационный объект является необязательным и имеет переменную длину. Структуру и кодирование определяет эмитент карты.

Данный информационный объект вводится при помощи байта, равного '5Y'.

8.3.5 Данные, предваряющие эмиссию карты

Этот информационный объект является необязательным и имеет переменную длину. Структуру и кодирование настоящий стандарт не устанавливает. Он может использоваться для указания:

- изготовителя карты;
- типа интегральных схем;
- изготовителя интегральных схем;
- версии маски постоянного запоминающего устройства;
- версии операционной системы.

Данный информационный объект вводится при помощи байта, равного '6Y'.

8.3.6 Функциональные возможности карты

Этот информационный объект является необязательным и имеет переменную длину. Его поле значения состоит либо из первой, либо из первых двух, либо из трех таблиц программных функций.

Данный информационный объект вводится при помощи байта, равного '71', '72' или '73'.

В таблице 85 представлена первая таблица программных функций.

Таблица 85 — Первая таблица программных функций

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
								Выбор файлов DF:
1	—	—	—	—	—	—	—	- по полному имени DF
—	1	—	—	—	—	—	—	- по частичному имени DF
—	—	1	—	—	—	—	—	- через путь
—	—	—	1	—	—	—	—	- по идентификатору файла
—	—	—	—	1	—	—	—	- неявный
								Управление файлами EF:
—	—	—	—	—	1	—	—	- поддерживается короткий идентификатор EF
—	—	—	—	—	—	1	—	- поддерживается номер записи
—	—	—	—	—	—	—	1	- поддерживается идентификатор записи

В таблице 86 представлена вторая таблица программных функций, которая представляет собой байт кодирования данных. Байт кодирования данных также может быть представлен как второй элемент данных в контрольном параметре файла с тегом '82' (см. таблицу 2 в 5.1.5).

Таблица 86 — Вторая таблица программных функций (байт кодирования данных)

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
—	×	×	—	—	—	—	—	Режим работы функций записи:
—	0	0	—	—	—	—	—	- однократная запись
—	0	1	—	—	—	—	—	- оригинальный
—	1	0	—	—	—	—	—	- запись с использованием операции «ИЛИ»
—	1	1	—	—	—	—	—	- запись с использованием операции «И»
—	—	—	—	—	×	×	×	Размер единицы данных в полубайтах (степень числа 2; например, 001 — два полубайта) (значение по умолчанию составляет один байт)
×	—	—	×	×	—	—	—	0...00... (другие значения являются RFU)

В таблице 87 представлена третья таблица программных функций.

Таблица 87 — Третья таблица программных функций

b8	b7	b6	b5	b4	b3	b2	b1	Смысловое содержание
×	—	—	—	—	—	—	—	0 (1 является RFU)
—	1	—	—	—		—	—	Расширенные поля L_c и L_e
—	—	×	—	—	—	—	—	0 (1 является RFU)
—	—	—	×	×	—	—	—	Назначение логических каналов осуществляет:
			1					- карта
				1				- устройство сопряжения
—	—	—	0	0	—	—	—	Нет логического канала
—	—	—	—	—	×	—	—	0 (1 является RFU)
—	—	—	—	—	—	×	y	Максимальное число логических каналов (равно 2x+y+1)

8.4 Информация о состоянии

Информация о состоянии образована из трех байтов: одного байта состояния жизненного цикла карты и двух байтов состояния SW1, SW2.

Значение '00' у байта состояния жизненного цикла карты указывает, что информация о состоянии жизненного цикла карты не предоставлена. Значения от '80' до 'FE' предназначаются для собственного использования. Все другие значения являются RFU.

Значение '9000' последовательности байтов SW1—SW2 указывает на нормальную обработку по 5.4.5.

Значение '0000' последовательности байтов SW1—SW2 указывает, что индикация состояния не представлена.

Если индикатору категории присваивается значение '80', то информация о состоянии может быть представлена в информационном объекте COMPACT-TLV. В этом случае номер тега равен '8'. Если длина равна '1', то значением является байт состояния жизненного цикла карты. Если длина равна '2', то значением являются байты SW1, SW2. Если длина равна '3', то значением является байт состояния жизненного цикла карты, сопровождаемый байтами SW1, SW2. Другие значения длины зарезервированы для ИСО.

8.5 Ссылка на данные DIR

Если индикатор категории имеет значение '10', то следующий за ним байт представляет собой ссылку на данные DIR. Кодирование и содержание этого байта находятся за пределами компетенции настоящего стандарта.

9 Услуги карты, не зависимые от приложений

9.1 Определения и область применения

В настоящем разделе приводится описание услуг, предоставляемых картой, не зависимых от приложений и именуемых далее в тексте как услуги карты. Их назначение заключается в обеспечении механизмов обмена между картой и устройством сопряжения, не имеющими никакой информации друг о друге, за исключением той, что они оба удовлетворяют требованиям настоящего стандарта.

Услуги карты поддерживаются благодаря любой комбинации из:

- байтов предыстории;
- содержимого одного или нескольких зарезервированных файлов EF;
- последовательностей межотраслевых команд.

Команды используют байт CLA = '00' (см. 5.4.1), т.е. используют основной логический канал и не используют безопасный обмен сообщениями.

Приложению не обязательно соответствовать положениям настоящего раздела, как только оно будет идентифицировано и выбрано в карте. Для осуществления подобных функций оно может использовать другие механизмы, совместимые с настоящим стандартом. Поэтому такие решения могут и не гарантировать обмен.

Определены следующие услуги карты:

- услуга по идентификации карты, позволяющая устройству сопряжения идентифицировать карту, а также выяснять, как с ней взаимодействовать;
- услуга по выбору приложения, позволяющая устройству сопряжения выяснять, какое приложение является в карте активным (если таковое имеется), а также, как осуществлять выбор и запуск приложения в карте;
- услуга по поиску информационных объектов, позволяющая осуществлять поиск информационных объектов, определяемых либо в настоящем стандарте, либо в других стандартах серии ГОСТ Р ИСО/МЭК 7816 (ИСО/МЭК 7816). Настоящий раздел дает описание стандартных механизмов поиска только для межотраслевых информационных объектов;
- услуга по выбору файла, позволяющая осуществлять выбор наименованных файлов DF и файлов EF;
- услуга по вводу/выводу содержимого файлов, предоставляющая доступ к данным, хранящимся в файлах EF.

9.2 Услуга по идентификации карты

Эта функция заключается в предоставлении картой внешнему окружению информации о своем логическом содержании, а также о некоторых общих информационных объектах, которые могут

использоваться всеми приложениями (например, о межотраслевых информационных объектах). Информация, называемая идентификационными данными карты, дается картой в байтах предыстории и, возможно, в файле, выбираемом неявным образом сразу после ответа на восстановление.

Доступ к этому файлу указывают исходные данные доступа (см. 8.3.3).

Если исходные данные доступа из байтов предыстории не обозначают команду СЧИТАТЬ, то ответное сообщение команды на выполнение содержит идентификационные данные карты.

9.3 Услуга по выбору приложения

Выбор приложения в карте осуществляется либо в неявной форме, либо в явной по его имени.

9.3.1 Неявный выбор приложения

Если выбор приложения в карте осуществляется неявно, то идентификатор приложения, определяемый в ИСО/МЭК 7816-5, должен быть указан в идентификационных данных карты. Если он там не представлен, то должен быть представлен в файле ATR.

9.3.2 Прямой выбор приложения

Карта в многоприкладной среде должна быть способна положительно реагировать на прямой выбор приложения с использованием команды ВЫБРАТЬ ФАЙЛ, указывающей идентификатор приложения в качестве имени DF.

Командный APDU должен предоставлять полный идентификатор приложения. В случае выбора приложения по частичному имени DF, может осуществляться выбор следующего приложения, согласующегося с предложенным именем, а полное имя DF будет доступно в ответном сообщении команды ВЫБРАТЬ ФАЙЛ в качестве контрольного параметра файла с тегом '84' (см. табл. 2 в 5.1.5).

Командный APDU команды на выполнение представлен в таблице 88.

Таблица 88 — Кодирование команды для прямого выбора приложения

CLA	'00' (см. 5.4.1)
INS	'A4'
P1—P2	'0400'
Поле L_c	Длина в байтах поля данных
Поле данных	Полное или частичное имя DF
Поле L_e	Должно быть представлено, содержит одни нули

9.4 Услуга по поиску информационных объектов

Информационные объекты, используемые для обмена, независимого от приложений, определены в настоящем стандарте и других стандартах серии ГОСТ Р ИСО/МЭК 7816 (ИСО/МЭК 7816).

Поиск этих информационных объектов полагается на один (или оба) из следующих методов:

- присутствие информационного объекта в идентификационных данных карты (см. 9.2);
- присутствие информационного объекта в файле DIR (путь равен '3F002F00') или файле ATR (путь равен '3F002F01').

Информация, необходимая для поиска информационных объектов косвенным методом, определена в ГОСТ Р ИСО/МЭК 7816-6.

9.5 Услуга по выбору файла

Если путь к файлу EF известен, то число подаваемых команд ВЫБРАТЬ ФАЙЛ равно длине пути, деленной на два, минус один (путь всегда начинается с текущего файла DF).

Если длина пути превышает четыре байта, то пока не будут использованы все имеющиеся в последовательности пути идентификаторы файлов DF, должна выполняться (одна или несколько), команда ВЫБРАТЬ ФАЙЛ с командным APDU, представленным в таблице 89.

Таблица 89 — Кодирование команды для выбора DF, использующего идентификатор файла

CLA	'00' (см. 5.4.1)
INS	'A4'
P1—P2	'0100'
Поле L_c	'02'
Поле данных	Идентификатор DF (из третьего и четвертого байтов пути)
Поле L_e	Пустое

Последним и, возможно, единственным выбором является выбор файла EF с использованием командного APDU, представленного в таблице 90.

Таблица 90 — Кодирование команды для выбора EF

CLA	'00' (см. 5.4.1)
INS	'A4'
P1—P2	'0200'
Поле L_c	'02'
Поле данных	Идентификатор EF (последние два байта пути)
Поле L_e	Пустое

9.6 Услуга по вводу/выводу содержимого файлов

Как только файл, используемый для межотраслевого обмена, будет выбран, его содержимое, предназначаемое для обмена, должно выдаваться в ответ на один из следующих командных APDU.

Если первая таблица программных функций отсутствует или не обозначает поддержку команд, ориентированных на запись, то должна выполняться команда, представленная в таблице 91.

Таблица 91 — Кодирование команды для считывания прозрачного файла

CLA	'00' (см. 5.4.1)
INS	'B0'
P1—P2	'0000'
Поле L_c	Пустое
Поле данных	»
Поле L_e	Должно быть представлено, содержит одни нули

Если первая таблица программных функций обозначает поддержку команд, ориентированных на запись, то должна выполняться команда, представленная в таблице 92.

Таблица 92 — Кодирование команды для считывания файла, ориентированного на записи

CLA	'00' (см. 5.4.1)
INS	'B2'
P1—P2	'0005'
Поле L_c	Пустое
Поле данных	»
Поле L_e	Должно быть представлено, содержит одни нули

ПРИЛОЖЕНИЕ А
(обязательное)

Транспортировка APDU-сообщений при помощи протокола передачи, обозначаемого T=0

А.1 Случай 1

Командный APDU отображается на командный TPDU путем присвоения параметру P3 значения '00':

командный APDU	CLA	INS	P1	P2	
командный TPDU	CLA	INS	P1	P2	P3 = '00'

Ответный TPDU отображается на ответный APDU без какого-либо изменения:

ответный TPDU	SW1	SW2
ответный APDU	SW1	SW2

А.2 Случай 2, короткий

В этом случае L_e принимает значения от 1 до 256 и кодируется в байте B_1 (B_1 = '00' означает максимум, т.е. L_e = 256).

Командный APDU отображается на командный TPDU без какого-либо изменения:

командный APDU	CLA	INS	P1	P2	$L_e = B_1$
командный TPDU	CLA	INS	P1	P2	$P3 = B_1$

Ответный TPDU отображается на ответный APDU в соответствии с вариантом принятия L_e и согласно обработке команды.

Случай 2К.1. L_e принята

Ответный TPDU отображается на ответный APDU без какого-либо изменения:

ответный TPDU	L_e байтов	SW1 SW2
ответный APDU	L_e байтов	SW1 SW2

Случай 2К.2. L_e определенно не принята

L_e не принята картой, не поддерживающей услугу по предоставлению данных, если длина указана неверно.

Ответный TPDU с карты указывает, что карта прерывает выполнение команды из-за неверно указанной длины: (SW1) = '67'. Ответный TPDU отображается на ответный APDU без какого-либо изменения:

ответный TPDU	SW1 = '67'	SW2
ответный APDU	SW1 = '67'	SW2

Случай 2К.3. L_e не принята, L_a указана

L_e не принята картой, и карта указывает действительную длину L_a .

Ответный TPDU с карты указывает, что выполнение команды прерывается вследствие неверно указанной длины и что правильная длина составляет L_a: (SW1) = '6C', а байт SW2 кодирует L_a.

Если передающая система не поддерживает услугу по повторной подаче одной и той же команды, она должна лишь отображать ответный TPDU на ответный APDU без какого-либо изменения:

ответный TPDU	SW1 = '6C'	SW2 = L_a
ответный APDU	SW1 = '6C'	SW2 = L_a

Если передающая система поддерживает услугу по повторной подаче одной и той же команды, она должна осуществлять повторную подачу того же командного TPDU, присваивая параметру P3 значение L_a:

командный TPDU	CLA	INS	P1	P2	P3 = SW2

Ответный TPDU состоит из L_a байтов, сопровождаемых двумя байтами состояния.

Если L_a меньше или равна L_e, то ответный TPDU отображается на ответный APDU без какого-либо изменения:

ответный TPDU	L_a байтов	SW1 SW2
ответный APDU	L_a байтов	SW1 SW2

Если L_a больше L_e, то ответный TPDU отображается на ответный APDU путем сохранения только первых L_e байтов тела и байтов состояния SW1, SW2:

ответный TPDU	L_a байтов	SW1 SW2
ответный APDU	L_e ($< L_a$) байтов	SW1 SW2

Случай 2К.4. Последовательность SW1—SW2 = '9XYZ', исключая '9000'

Ответный TPDU отображается на ответный APDU без какого-либо изменения.

А.3 Случай 3, короткий

В этом случае L_c принимает значения от 1 до 255 и кодируется в байте B_1 ($B_1 \neq$ '00').

Командный APDU отображается на командный TPDU без какого-либо изменения:

командный APDU	CLA	INS	P1	P2	$L_c = B_1$	L_c байтов
командный TPDU	CLA	INS	P1	P2	P3 = B_1	L_c байтов

Ответный TPDU отображается на ответный APDU без какого-либо изменения:

ответный TPDU	SW1	SW2
ответный APDU	SW1	SW2

А.4 Случай 4, короткий

В этом случае L_c принимает значения от 1 до 255 и кодируется в байте B_1, L_e принимает значения от 1 до 256 и кодируется в байте B_L (B_L = '00' означает максимум, т.е. L_e = 256).

Командный APDU отображается на командный TPDU путем отсечения последнего байта тела:

командный APDU	CLA	INS	P1	P2	$B_1 = L_c$	Lc байтов	B_L
командный TPDU	CLA	INS	P1	P2	P3 = B_1	L_c байтов	

Случай 4К.1. Команда не принята

Первый ответный TPDU с карты указывает, что карта прервала выполнение команды: SW1 = '6X', исключая '61'.

Ответный TPDU отображается на ответный APDU без какого-либо изменения:

ответный TPDU	SW1 = '6X '	SW2
ответный APDU	SW1 = '6X '	SW2

Случай 4К.2. Команда принята

Первый ответный TPDU с карты указывает, что карта выполнила команду: последовательность SW1—SW2 = '9000'.

Передающая система должна осуществлять подачу карте командного TPDU команды ИЗВЛЕЧЬ ОТВЕТ, присваивая параметру P3 значение L_e:

командный TPDU	CLA	INS = ИЗВЛЕЧЬ ОТВЕТ	P1	P2	P3 = B_L

В зависимости от второго ответного TPDU, исходящего с карты, передающая система должна реагировать, как описано выше для случаев 2К.1—2К.4.

Случай 4К.3. Команда принята с предоставлением дополнительной информации

Первый ответный TPDU с карты указывает, что карта выполнила команду и дает информацию о длине имеющихся байтов данных L_x: байт SW1 = '61', а байт SW2 кодирует L_x.

Передающая система должна осуществлять подачу карте командного TPDU команды ИЗВЛЕЧЬ ОТВЕТ, присваивая параметру Р3 меньшее из значений L_x и L_e:

командный TPDU	CLA	INS = ИЗВЛЕЧЬ ОТВЕТ	P1	P2	P3 = min (L_e, L_x)

Второй ответный TPDU отображается на ответный APDU без какого-либо изменения:

ответный TPDU	Р3 байтов	SW1 SW2
ответный APDU	Р3 байтов	SW1 SW2

Случай 4К.4. Последовательность SW1—SW2 = '9XYZ', исключая '9000'

Ответный TPDU отображается на ответный APDU без какого-либо изменения.

А.5 Случай 2, расширенный

В этом случае L_e принимает значения от 1 до 65536 и кодируется в трех байтах: (B_1) = '00', (B_2 ‖ B_3) — любое значение (байты B_2 и B_3, имеющие значение '0000', означают максимум, т.е. L_e = 65536). Командный APDU имеет следующую структуру:

командный APDU	CLA	INS	P1	P2	B_1 = '00'	B_2B_3 = L_e

Случай 2Р.1. $L_e \le 256$, B_1 = '00', последовательность B_2B_3 принимает значения от '0001' до '0100'

Командный APDU должен отображаться на командный TPDU путем присвоения параметру Р3 значения байта B_3:

командный TPDU	CLA	INS	P1	P2	P3 = B_3

Обработка команды передающей системой должна соответствовать случаю 2К.

Случай 2Р.2. $L_e > 256$, B_1 = '00', последовательность B_2B_3 либо равна '0000', либо принимает значения от '0101' до 'FFFF'

Командный APDU должен отображаться на командный TPDU путем присвоения параметру Р3 значения '00':

командный TPDU	CLA	INS	P1	P2	P3 = '00'

а) Если первый ответный TPDU с карты указывает, что карта прервала выполнение команды из-за неверно указанной длины (SW1 = '67'), то ответный TPDU должен отображаться на ответный APDU без какого-либо изменения:

ответный TPDU	SW1 = '67 '	SW2
ответный APDU	SW1 = '67 '	SW2

б) Если первый ответный TPDU с карты указывает, что выполнение команды прерывается вследствие неверно указанной длины и что правильная длина составляет L_a (SW1 = '6C' и SW2 = L_a), то передающая система должна завершать обработку, как описано для случая 2К.3.

в) Если первый ответный TPDU состоит из 256 байтов данных, сопровождаемых последовательностью байтов SW1—SW2 = '9000', это означает, что карта не располагает более чем 256 байтами данных и/или не поддерживает команду ИЗВЛЕЧЬ ОТВЕТ. Передающая система должна в таком случае отображать ответный TPDU на ответный APDU без какого-либо изменения:

ответный TPDU	256 байтов	SW1= '90'	SW2 = '00'
ответный APDU	256 байтов	SW1 = '90'	SW2 = '00'

г) Если первый или последующий ответный TPDU с карты начинается с байта SW1 = '61', то байт SW2 кодирует длину L_x, которая представляет дополнительное количество байтов, доступных для получения с карты (байт SW2, имеющий значение '00', указывает на 256 дополнительных байтов или более). Для определения количества L_m остающихся байтов, которые должны быть извлечены из карты, передающая система должна выполнять следующее вычисление: $L_m = L_e$ — (сумма длин тел ранее принятых ответных TPDU).

Если $L_m = 0$, то передающая система должна осуществлять последовательное объединение тел всех принятых ответных TPDU вместе с завершителем последнего принятого ответного TPDU в один ответный APDU.

Если $L_m > 0$, то передающая система должна осуществлять подачу командного TPDU команды ИЗВЛЕЧЬ ОТВЕТ, присваивая параметру P3 меньшее из значений L_x и L_m. Соответствующий ответный TPDU, исходящий с карты, должен обрабатываться:

- как указано в случае г), если SW1 = '61',
- как указано в случае, когда $L_m = 0$, если SW1 = '90'.

А.6 Случай 3, расширенный

В этом случае L_c принимает значения от 1 до 65535 и кодируется в трех байтах: $(B_1) = '00'$, $(B_2 \mid B_3) \neq '0000'$. Командный APDU имеет следующую структуру:

командный APDU	CLA	INS	P1	P2	$B_1 = '00'$	$B_2B_3 = L_c$	L_c байтов

Случай 3Р.1. $0 < L_c < 256$, $B_1 = '00'$, $B_2 = '00'$, $B_3 \neq '00'$

Командный APDU отображается на командный TPDU путем присвоения параметру P3 значения байта B_3:

командный TPDU	CLA	INS	P1	P2	$P3 = B_3$	L_c байтов

В этом случае L_c принимает значения от 1 до 255 и кодируется в одном байте.

Ответный TPDU отображается на ответный APDU без какого-либо изменения:

ответный TPDU	SW1	SW2
ответный APDU	SW1	SW2

Случай 3Р.2. $L_c > 255$, $B_1 = '00'$, $B_2 \neq '00'$, B_3 принимает любое значение

Если передающая система не поддерживает команду КОНВЕРТ, она должна возвращать ответный APDU ошибки, означающий, что длина указана неверно (SW1 = '67'):

ответный TPDU	SW1 = '67 '	SW2
ответный APDU	SW1 = '67 '	SW2

Если передающая система поддерживает команду КОНВЕРТ, она должна осуществлять разбиение APDU на сегменты длиной менее 256 и посылать эти сегменты в порядке их следования в телах последовательных командных TPDU команды КОНВЕРТ:

командный TPDU	CLA	INS = КОНВЕРТ	P1	P2	P3	P3 байтов

Если первый ответный TPDU, исходящий с карты, указывает, что карта не поддерживает команду КОНВЕРТ (SW1 = '6D'), то TPDU должен отображаться на ответный APDU без какого-либо изменения:

ответный TPDU	SW1 = '6D'	SW2
ответный APDU	SW1 = '6D'	SW2

Если первый ответный TPDU с карты указывает, что карта поддерживает команду КОНВЕРТ (последовательность SW1—SW2 = '9000'), то передающая система должна и далее посылать команды КОНВЕРТ, если потребуется:

ответный TPDU	SW1—SW2 = '9000'					
командный TPDU	CLA	INS = КОНВЕРТ	P1	P2	P3	P3 байтов

Ответный TPDU, соответствующий последней команде КОНВЕРТ, отображается на ответный APDU без какого-либо изменения:

ответный TPDU	SW1	SW2
ответный APDU	SW1	SW2

А.7 Случай 4, расширенный

В этом случае L_c принимает значения от 1 до 65535 и кодируется в трех байтах: (B_1 = '00', (B_2 || B_3) ≠ '0000', а L_e принимает значения от 1 до 65536 и кодируется в двух байтах: (B_{L-1} || B_L) — любое значение (B_{L-1} и B_L, имеющие значение '0000', означают максимум, т.е. L_e = 65536). Командный APDU имеет следующую структуру:

командный APDU	CLA	INS	P1	P2	B_1 = '00'	$B_2B_3 = L_c$	Lc байтов	B_{L-1} $B_L = L_e$

Случай 4Р.1. L_c < 256, B_1 = '00', B_2 = '00', B_3 ≠ '00'

Командный APDU отображается на командный TPDU путем отсечения последних двух байтов B_{L-1} и B_L и путем присвоения параметру P3 значения байта B_3:

командный TPDU	CLA	INS	P1	P2	P3 = B_3	L_c байтов

В этом случае L_c принимает значения от 1 до 255 байтов и кодируется в одном байте.

а) Если SW1 = '6X' в первом ответном TPDU, исходящем с карты, то ответный TPDU отображается на ответный APDU без какого-либо изменения:

ответный TPDU	SW1 = '6X'	SW2
ответный APDU	SW1 = '6X'	SW2

б) Если SW1 = '90' в первом ответном TPDU, исходящем с карты, то:

- если L_e < 257 (значение последовательности байтов B_{L-1} и B_L составляет от '0001' до '0100'), то передающая система должна осуществлять подачу командного TPDU команды ИЗВЛЕЧЬ ОТВЕТ, присваивая параметру P3 значение байта B_L. Последующая обработка команды передающей системой должна соответствовать вышеприведенным случаям 2К.1—2К.4;
- если L_e > 256 (значение последовательности байтов B_{L-1} и B_L составляет '0000' или более '0100'), то передающая система должна осуществлять подачу командного TPDU команды ИЗВЛЕЧЬ ОТВЕТ, присваивая параметру P3 значение '00'. Последующая обработка команды передающей системой должна соответствовать описанному выше случаю 2Р.2.

в) Если SW1 = '61' в первом ответном TPDU, исходящем с карты, то передающая система должна действовать далее, как установлено выше для случая 2Р.2 г).

Случай 4Р.2. L_c > 255, B_1 = '00', B_2 ≠ '00', B_3 принимает любое значение

Передающая система должна действовать в соответствии с описанным выше случаем 3Р.2 до тех пор, пока командный APDU не будет полностью послан карте. Затем передающая система должна действовать в соответствии с описанным выше случаем 4Р.1 а), б), в).

ПРИЛОЖЕНИЕ Б
(обязательное)

Транспортировка APDU-сообщений при помощи протокола передачи, обозначаемого T=1

Б.1 Случай 1

Командный APDU отображается на информационное поле I-блока без какого-либо изменения:

командный APDU	CLA	INS	P1	P2
информационное поле	CLA	INS	P1	P2

Информационное поле I-блока, получаемого в ответе, отображается на ответный APDU без какого-либо изменения:

информационное поле	SW1	SW2
ответный APDU	SW1	SW2

Б.2 Случай 2 (короткий и расширенный)

Командный APDU отображается на информационное поле I-блока без какого-либо изменения:

командный APDU	CLA	INS	P1	P2	Поле L_e
информационное поле	CLA	INS	P1	P2	Поле L_e

Ответный APDU состоит:
- либо из информационного поля I-блока, получаемого в ответе;
- либо из сцепления информационных полей последовательных I-блоков, получаемых в ответе (эти блоки должны быть связаны в цепочку):

либо информационное поле	Поле данных	SW1 SW2

либо сцепление информационных полей

	Поле	
	... ...	
	... данных	SW1 SW2
ответный APDU	Поле данных	SW1 SW2

Б.3 Случай 3 (короткий и расширенный)

Командный APDU отображается без какого-либо изменения:
- либо на информационное поле одного I-блока;
- либо на сцепление информационных полей последовательных I-блоков, которые должны связываться в цепочку:

командный APDU	CLA	INS	P1	P2	Поле L_c	Поле данных
либо информационное поле	CLA	INS	P1	P2	Поле L_c	Поле данных

либо сцепление информационных полей

CLA	INS	P1	P2	Поле L_c	Поле ...
...		...			...
...		...			данных

Информационное поле I-блока, получаемого в ответе, отображается на ответный APDU без какого-либо изменения:

информационное поле	SW1	SW2
ответный APDU	SW1	SW2

Б.4 Случай 4 (короткий и расширенный)

Командный APDU отображается без какого-либо изменения:
- либо на информационное поле одного I-блока;
- либо на сцепление информационных полей последовательных I-блоков, которые должны связываться в цепочку:

командный APDU	CLA	INS	P1	P2	Поле L_c	Поле данных	Поле L_e
либо информационное поле	CLA	INS	P1	P2	Поле L_c	Поле данных	Поле L_e

либо сцепление информационных полей

CLA	INS	P1	P2	Поле L_c	Поле ...	
...		...			...	
...		...		данных		Поле L_e

Ответный APDU состоит:
- либо из информационного поля I-блока, получаемого в ответе;
- либо из сцепления информационных полей последовательных I-блоков, получаемых в ответе (эти блоки должны быть связаны в цепочку):

либо информационное поле	Поле данных	SW1	SW2

либо сцепление информационных полей

Поле	...		...
...	...		...
...	данных	SW1	SW2
ответный APDU	Поле данных	SW1	SW2

ПРИЛОЖЕНИЕ В
(справочное)

Управление указателя записи

B.1 Случай 1

Случай 1 рассматривает ситуации, связанные с первой командой, подаваемой после осуществления функции выбора (либо в явной, либо в неявной форме). Положение указателя текущей записи (ТЗ) является неопределенным.

Команда СЧИТАТЬ ЗАПИСЬ(И)	Запись в ответе	Положение указателя ТЗ после команды
Следующая (идентификатор = aa)	Первая с идентификатором aa Если не найдена, тогда ошибка	Считанная запись Неопределенное
Предыдущая (идентификатор = bb)	Последняя с идентификатором bb Если не найдена, тогда ошибка	Считанная запись Неопределенное
Первая (идентификатор = cc)	Первая с идентификатором cc Если не найдена, тогда ошибка	Считанная запись Неопределенное
Последняя (идентификатор = dd)	Последняя с идентификатором dd Если не найдена, тогда ошибка	Считанная запись Неопределенное
Следующая (идентификатор = 00)	Первая	Считанная запись
Предыдущая (идентификатор = 00)	Последняя	То же
Первая (идентификатор = 00)	Первая	»
Последняя (идентификатор = 00)	Последняя	Считанная запись
Запись # 00	Ошибка	Неопределенное
Запись # ee	# ee Если не найдена, тогда ошибка	»
P1='00', P2=xxxx x101	Ошибка	»
P1='00', P2=xxxx x110	»	»
# jj, P2=xxxx x101	Начиная с # jj и заканчивая последней Если запись # jj не найдена, тогда ошибка	»
# kk, P2=xxxx x110	Начиная с последней и заканчивая # kk Если запись # kk не найдена, тогда ошибка	»

B.2 Случай 2

Случай 2 рассматривает ситуации, связанные с последующей командой. Положение указателя ТЗ определено.

Команда СЧИТАТЬ ЗАПИСЬ(И)	Запись в ответе	Положение указателя ТЗ после команды
Следующая (идентификатор = aa)	Следующая с идентификатором aa Если следующая отсутствует, тогда ошибка	Считанная запись Не изменилось
Предыдущая (идентификатор = bb)	Предыдущая с идентификатором bb Если предыдущая отсутствует, тогда ошибка	Считанная запись Не изменилось
Первая (идентификатор = cc)	Первая с идентификатором cc Если не найдена, тогда ошибка	Считанная запись Не изменилось

Команда СЧИТАТЬ ЗАПИСЬ(И)	Запись в ответе	Положение указателя ТЗ после команды
Последняя (идентификатор = dd)	Последняя с идентификатором dd Если не найдена, тогда ошибка	Считанная запись Не изменилось
Следующая (идентификатор = 00)	ТЗ + 1 Если ТЗ = последняя, тогда ошибка	Предшествующая ТЗ + 1 Не изменилось
Предыдущая (идентификатор = 00)	ТЗ — 1 Если ТЗ = первая, тогда ошибка	Предшествующая ТЗ — 1 Не изменилось
Первая (идентификатор = 00)	Первая	Первая запись
Последняя (идентификатор = 00)	Последняя	Последняя запись
Запись # 00	ТЗ	Не изменилось
Запись # ee	# ee Если не найдена, тогда ошибка	То же »
P1='00', P2=xxxx x101	Начиная с ТЗ и заканчивая последней	»
P1='00', P2=xxxx x110	Начиная с последней и заканчивая ТЗ	»
# jj, P2=xxxx x101	Начиная с # jj и заканчивая последней Если запись # jj не найдена, тогда ошибка	» »
# kk, P2=xxxx x110	Начиная с последней и заканчивая # kk Если запись # kk не найдена, тогда ошибка	» »

ПРИЛОЖЕНИЕ Г
(справочное)

Использование базовых правил кодирования АСН.1

Г.1 Информационный объект BER-TLV

Каждый информационный объект BER-TLV (см. ГОСТ Р ИСО/МЭК 8825) должен состоять из двух или трех последовательных полей.

Поле тега T состоит из одного или большего числа последовательных байтов. Оно кодирует класс, тип и номер.

Поле длины состоит из одного или большего числа последовательных байтов. Оно кодирует целое число L.

Если L не является нулем, то поле значения V состоит из L последовательных байтов. Если L — нуль, то информационный объект является пустым: поле значения отсутствует.

Стандарты серии ГОСТ Р ИСО/МЭК 7816 (ИСО/МЭК 7816) не используют в качестве значения тега ни '00', ни 'FF.

Примечание — Перед и между информационными объектами BER-TLV или после них могут возникать байты со значениями '00' или 'FF' без какого-либо смыслового содержания (например, как следствие удаленных или измененных TLV-закодированных информационных объектов).

Г.2 Поле тега

Биты b8 и b7 начального байта поля тега должны кодировать класс тега, т.е. класс информационного объекта.

b8—b7 = 00 вводят тег универсального класса.

b8—b7 = 01 вводят тег прикладного класса.

b8—b7 = 10 вводят тег контекстно-зависимого класса.

b8—b7 = 11 вводят тег пользовательского класса.

Бит b6 начального байта поля тега должен кодировать тип тега, т.е. тип информационного объекта.

b6 = 0 вводит простой информационный объект.

b6 = 1 вводит составной информационный объект.

Если биты b5—b1 начального байта не все установлены в состояние «1», то они должны кодировать целое

число, равное номеру тега, которое, следовательно, находится в диапазоне от 0 до 30. В этом случае поле тега состоит из единственного байта.

В противном случае (биты b5—b1 в начальном байте установлены в состояние «1») поле тега должно продолжаться на один или большее число последующих байтов.

Бит b8 каждого байта продолжения, за исключением последнего байта, должен быть установлен в состояние «1».

Биты b7—b1 первого байта продолжения не должны быть все установлены в состояние «0».

Биты b7—b1 первого байта продолжения, сцепленные с битами b7—b1 каждого из последующих байтов продолжения, включая биты b7—b1 последнего байта, должны кодировать целое число, равное номеру тега (таким образом, строго положительное).

Г.3 Поле длины

В коротком формате поле длины состоит из единственного байта, где бит b8 должен быть установлен в состояние «0», а биты b7—b1 должны кодировать целое число, равное числу байтов в поле значения. Следовательно, любая длина от 0 до 127 может быть закодирована одним байтом.

В длинном формате поле длины состоит из начального байта, где бит b8 должен быть установлен в состояние «1», а биты b7—b1 не должны быть все равны, кодируя таким образом положительное целое число, равное числу последующих байтов в поле длины. Эти последующие байты должны кодировать целое число, равное числу байтов в поле значения. Следовательно, любая длина в пределах максимума, установленного для APDU (до 65535 включительно), может быть закодирована тремя байтами.

Примечание — Стандарты серии ГОСТ Р ИСО/МЭК 7816 (ИСО/МЭК 7816) не используют неопределенную длину, устанавливаемую базовыми правилами кодирования АСН.1 (см. ГОСТ Р ИСО/МЭК 8825).

Г.4 Поле значения

В настоящем стандарте поле значения некоторых простых информационных объектов BER-TLV содержит ноль, один или большее число информационных объектов SIMPLE-TLV.

Поле значения любого другого простого информационного объекта BER-TLV содержит ноль, один или большее число элементов данных, задаваемых в описании информационного объекта.

Поле значения каждого составного информационного объекта BER-TLV содержит ноль, один или большее число информационных объектов BER-TLV.

ПРИЛОЖЕНИЕ Д
(справочное)

Примеры профилей карт

Д.1 Введение

Настоящее приложение определяет ряд профилей карт, которые могут служить ориентирами для разработчиков приложений при выборе команд для использования в приложениях. Эти профили могут также помочь при определении возможностей, которые желательно предусмотреть в карте. Профили карт могут сочетаться.

Д.2 Профиль М

В картах этого профиля предусматривают, как минимум, следующие возможности и команды.

Структуры файлов:

- прозрачная структура;
- линейная структура с записями фиксированной длины.

Команды:

- СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ и ОБНОВИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ с:

байтом P1, где бит b8 = 0,

длинами до 256 байтов включительно;

- СЧИТАТЬ ЗАПИСЬ(И) и ОБНОВИТЬ ЗАПИСЬ с байтом:

P2, где биты b8—b4 = 0,

P2, где бит b3 = 1,

P2, где последовательность битов b3—b2—b1 = 000, 001, 010 или 011, и байтом P1 = 0;

- ВЫБРАТЬ ФАЙЛ с последовательностью байтов P1—P2 = '0000';

- ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ с последовательностью байтов P1—P2 = '0001' или '0002';
- ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ с последовательностью байтов P1—P2 = '0000'.

Д.3 Профиль N

Этот профиль является таким же, как профиль M, плюс дополнительная опция P1 = '04' в команде ВЫБРАТЬ ФАЙЛ.

Д.4 Профиль O

В картах этого профиля предусматривают, как минимум, следующие возможности и команды.
Структуры файлов:
- прозрачная структура;
- линейная структура с записями фиксированной длины;
- линейная структура с записями переменной длины;
- циклическая структура с записями фиксированной длины.

Команды:
- СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ, ВВЕСТИ ДВОИЧНОЕ ЗНАЧЕНИЕ и ОБНОВИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ с:

байтом P1, где бит b8 = 0,
длинами до 256 байтов включительно;
- СЧИТАТЬ ЗАПИСЬ(И), ВВЕСТИ ЗАПИСЬ и ОБНОВИТЬ ЗАПИСЬ с байтом:

P2, где биты b8—b4 = 0,
P2, где бит b3 = 1,
P2, где последовательность битов b3—b2—b1 = 000, 001, 010 или 011, и байтом P1 = 0;
- ПРИСОЕДИНИТЬ ЗАПИСЬ с последовательностью байтов P1—P2 = '0000';
- ВЫБРАТЬ ФАЙЛ с байтом:

P1 = '00', '01', '02', '03', '04', '08' или '09',
P2 = '00';
- ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ с последовательностью байтов P1—P2 = '0001' или '0002';
- ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ с последовательностью байтов P1—P2 = '0000';
- ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ с последовательностью байтов P1—P2 = '0000';
- СОЗДАТЬ ЗАДАЧУ с последовательностью байтов P1—P2 = '0000'.

Д.5 Профиль P

В картах этого профиля предусматривают, как минимум, следующие возможности и команды.
Структуры файлов — прозрачная структура.
Байты предыстории:
- данные об услугах, предоставляемых картой (равны '3188');
- исходные данные доступа (равны '4164').

Команды:
- СЧИТАТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ и ОБНОВИТЬ ДВОИЧНОЕ ЗНАЧЕНИЕ с:

байтом P1, где бит b8 = 0,
длинами до 64 байтов включительно;
- ВЫБРАТЬ ФАЙЛ с последовательностью байтов P1—P2 = '0400';
- ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ с последовательностью байтов P1—P2 = '0001' или '0002';
- ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ с последовательностью байтов P1—P2 = '0000'.

Д.6 Профиль Q

В картах этого профиля предусматривают, как минимум, следующие возможности и команды.
Байты предыстории:
- исходные данные доступа (равны '45' плюс команда ИЗВЛЕЧЬ);
- функциональные возможности карты (равны '7180').

Безопасный обмен сообщениями.
Команды:
- ИЗВЛЕЧЬ ДАННЫЕ и ПОМЕСТИТЬ ДАННЫЕ с тегом в байтах P1, P2;
- ВЫБРАТЬ ФАЙЛ с последовательностью байтов P1—P2 = '0401', '0402' или '0403';
- ВЫПОЛНИТЬ ВЕРИФИКАЦИЮ с байтом P1 = '00';
- ВЫПОЛНИТЬ ВНУТРЕННЮЮ АУТЕНТИФИКАЦИЮ;
- ВЫПОЛНИТЬ ВНЕШНЮЮ АУТЕНТИФИКАЦИЮ;
- СОЗДАТЬ ЗАДАЧУ.

ПРИЛОЖЕНИЕ Е
(справочное)

Использование безопасного обмена сообщениями

Е.1 Сокращения

В настоящем приложении применяют следующие сокращения:
CC — криптографическая контрольная сумма (cryptographic checksum);
CG — криптограмма (cryptogram);
CH — заголовок команды (CLA, INS, P1, P2) (command header);
CR — управляющая ссылка (control reference);
FR — ссылка на файл (file reference);
KR — ссылка на ключ (key reference);
L — длина (length);
LE — значение L_e в незащищенной команде (один или два байта, кодирующих положительное целое число без знака; нулевое значение означает максимум);
PB — байты незначащей информации (байт со значением '80', за которым следуют от 0 до k—1 байтов со значением '00', где k — длина блока) (padding bytes);
PI — байт индикатора заполнения незначащей информацией (padding indicator byte);
PV — незашифрованное значение (plain value);
RD — описатель ответа (response descriptor);
T — тег (tag);
‖ — сцепление.

Е.2 Криптографическая контрольная сумма

В соответствии с 5.7 использование криптографических контрольных сумм (см. 5.6.3.1) представлено для четырех случаев, определяемых в таблице 4 и на рисунке 4. В приведенных примерах значение L_{CC} равняется четырем. CLA* указывает на использование безопасного обмена сообщениями, т.е. в байте CLA, который равен '0X', '8X', '9X' или 'AX' в соответствии с таблицей 9, бит b4 равен единице.

Случай 1. Нет данных, нет данных.

Незащищенная пара команда—ответ следующая:

Заголовок команды	Тело команды
CLA INS P1 P2	Пустое

Тело ответа	Завершитель ответа
Пустое	SW1 SW2

Случай 1а. Защита состояния не применяется.

Защищенный командный APDU следующий:

Заголовок команды	Тело команды
CLA* INS P1 P2	Новое поле L_c (один байт, равный '06') ‖ Новое поле данных (шесть байтов)

Новое поле данных — один информационный объект
T_{CC} ‖ L_{CC} ‖CC
Данные, охватываемые CC (бит b3=1 в байте CLA*), — один блок
CH ‖ PB
Защищенный ответный APDU следующий:

Тело ответа	Завершитель ответа
Пустое	Новые SW1 SW2

Случай 1б. Защита состояния применяется.

Защищенный командный APDU следующий:

Заголовок команды	Тело команды
CLA* INS P1 P2	Новое поле L_c (один байт, равный '06') ‖ Новое поле данных (шесть байтов) ‖ Новое поле L_e (один байт, равный '00')

Новое поле данных — один информационный объект

T_{CC} ‖ L_{CC} ‖ CC

Данные, охватываемые CC (бит b3=1 в байте CLA*), — один блок

CH ‖ PB

Защищенный ответный APDU следующий:

Тело ответа	Завершитель ответа
Новое поле данных	Новые SW1 SW2

Новое поле данных — два информационных объекта:

T_{SW} (бит b1=1) ‖ L_{SW} ‖ SW (новые SW1, SW2) ‖

T_{CC} ‖ L_{CC} ‖ CC

Данные, охватываемые CC, — один блок

T_{SW} (бит b1=1) ‖ L_{SW} ‖ SW ‖ PB

Случай 2. Нет данных, данные.

Незащищенная пара команда—ответ следующая:

Заголовок команды	Тело команды
CLA INS P1 P2	Поле L_e

Тело ответа	Завершитель ответа
Поле данных	SW1 SW2

Защищенный командный APDU следующий:

Заголовок команды	Тело команды
CLA* INS P1 P2	Новое поле L_c ‖ Новое поле данных ‖ Новое поле L_e (один или два байта, равные '00')

Новое поле данных — два информационных объекта:

T_{LE} (бит b1=1) ‖ L_{LE} ‖ LE ‖

T_{CC} ‖ L_{CC} ‖ CC

Данные, охватываемые CC:

- один блок (если бит b3=0 в байте CLA*)

T_{LE} (бит b1=1) ‖ L_{LE} ‖ LE ‖ PB ;

- два блока (если бит b3=1 в байте CLA*):

CH ‖ PB ‖

T_{LE} (бит b1=1) ‖ L_{LE} ‖ LE ‖ PB

Защищенный ответный APDU следующий:

Тело ответа	Завершитель ответа
Новое поле данных	Новые SW1 SW2

Новое поле данных — три информационных объекта:

T_{PV} (бит b1=1) || L_{PV} || PV ||

[T_{SW} (бит b1=1) || L_{SW} || SW (новые SW1, SW2)] ||

T_{CC} || L_{CC} || CC

Данные, охватываемые CC, — один или более блоков

T_{PV} (бит b1=1) || L_{PV} || PV || [T_{SW} (бит b1=1) || L_{SW} | SW] || PB

Случай 3. Данные, нет данных.

Незащищенная пара команда—ответ следующая:

Заголовок команды	Тело команды
CLA INS P1 P2	Поле L_c | Поле данных

Тело ответа	Завершитель ответа
Пустое	SW1 SW2

Случай 3а. Защита состояния не применяется.

Защищенный командный APDU следующий:

Заголовок команды	Тело команды
CLA* INS P1 P2	Новое поле L_c | Новое поле данных

Новое поле данных — два информационных объекта:

T_{PV} (бит b1=1) || L_{PV} || PV ||

T_{CC} || L_{CC} || CC

Данные, охватываемые CC:

- один или более блоков (если бит b3=0 в байте CLA*)

T_{PV} (бит b1=1) || L_{PV} || PV || PB;

- два или более блоков (если бит b3=1 в байте CLA*):

CH | PB |

T_{PV} (бит b1=1) || L_{PV} || PV || PB

Защищенный ответный APDU следующий:

Тело ответа	Завершитель ответа
Пустое	Новые SW1 SW2

Случай 3б. Защита состояния применяется.

Защищенный командный APDU следующий:

Заголовок команды	Тело команды
CLA* INS P1 P2	Новое поле L_c || Новое поле данных || Новое поле L_e (один или два байта, равные '00')

Новое поле данных — два информационных объекта:

T_{PV} (бит b1=1) || L_{PV} || PV ||

T_{CC} || L_{CC} || CC

Данные, охватываемые CC:

- один или более блоков (если бит b3=0 в байте CLA*)

T_{PV} (бит b1=1) || L_{PV} || PV || PB;

- два или более блоков (если бит b3=1 в байте CLA*):

CH | PB |

T_{PV} (бит b1=1) || L_{PV} || PV || PB

Защищенный ответный APDU следующий:

Тело ответа	Завершитель ответа
Новое поле данных	Новые SW1 SW2

Новое поле данных — два информационных объекта:

T_{SW} (бит b1=1) || L_{SW} || SW (новые SW1, SW2)] ||

T_{CC} || L_{CC} || CC

Данные, охватываемые CC, — один блок

T_{SW} (бит b1=1) || L_{SW} || SW] || PB

Случай 4. Данные, данные.

Незащищенная пара команда—ответ следующая:

Заголовок команды	Тело команды
CLA INS P1 P2	Поле L_c | Поле данных | Поле L_e

Тело ответа	Завершитель ответа
Поле данных	SW1 SW2

Защищенный командный APDU следующий:

Заголовок команды	Тело команды
CLA* INS P1 P2	Новое поле L_c | Новое поле данных | Новое поле L_e (один или два байта, равные '00')

Новое поле данных — три информационных объекта:

T_{PV} (бит b1=1) || L_{PV} || PV |

T_{LE} (бит b1=1) || L_{LE} || LE ||

T_{CC} || L_{CC} || CC

Данные, охватываемые CC:

- один или более блоков (если бит b3=0 в байте CLA*)

T_{PV} (бит b1=1) || L_{PV} || PV | T_{LE} (бит b1=1) || L_{LE} || LE | PB;

- два или более блоков (если бит b3=1 в байте CLA*):

CH || PB ||

T_{PV} (бит b1=1) || L_{PV} || PV | T_{LE} (бит b1=1) || L_{LE} || LE || PB

Защищенный ответный APDU следующий:

Тело ответа	Завершитель ответа
Новое поле данных	Новые SW1 SW2

Новое поле данных — три информационных объекта:

T_{PV} (бит b1=1) || L_{PV} || PV |

[T_{SW} (бит b1=1) || L_{SW} || SW (новые SW1, SW2)] ||

T_{CC} || L_{CC} || CC

Данные, охватываемые CC, — один или более блоков

T_{PV} (бит b1=1) || L_{PV} || PV || [T_{SW} (бит b1=1) || L_{SW} || SW] || PB

Е.3 Криптограммы

Использование криптограмм с заполнением блоков данных незначащей информацией и без заполнения (см. 5.6.4) показано в полях данных (как командного, так и ответного APDU). Вместо информационных объектов с незашифрованным значением, фигурирующих в предыдущих примерах, использованы информационные объекты для конфиденциальности следующим образом.

Случай а. Незашифрованные данные, не закодированные информационными объектами BER-TLV.

Поле данных

$T_{PI\ CG} \| L_{PI\ CG} \| PI \| CG$

Данные, переносимые CG, — один или более блоков:

не BER-TLV закодированные данные

и байты незначащей информации в соответствии с PI.

Случай б. Незашифрованные данные, закодированные информационными объектами BER-TLV.

Поле данных

$T_{CG} \| L_{CG} \| CG$

Данные, переносимые CG, — строка скрываемых байтов:

информационные объекты BER-TLV (незначащая

информация, зависящая от алгоритма и режима его работы).

Е.4 Управляющие ссылки

Показано использование управляющих ссылок (см. 5.6.5.1).

Поле данных команды

$T_{CR} \| L_{CR} \| CR$,

где $CR = T_{FR} \| L_{FR} \| FR \| T_{KR} \| L_{KR} \| KR$

Е.5 Описатель ответа

Показано использование описателя ответа (см. 5.6.5.2).

Поле данных команды

$T_{RD} \| L_{RD} \| RD$,

где $RD = T_{PV} \| '00' \| T_{CC} \| '00'$

Поле данных ответа

$T_{PV} \| L_{PV} \| PV \| T_{CC} \| L_{CC} \| CC$

Е.6 Команда КОНВЕРТ

Показано использование команды КОНВЕРТ (см. 7.2).

Поле данных команды

$T_{PI\ CG} \| L_{PI\ CG} \| PI \| CG$

Данные, переносимые CG:

командный APDU, начинающийся с CH,

и байты незначащей информации в соответствии с PI.

Поле данных ответа

$T_{PI\ CG} \| L_{PI\ CG} \| PI \| CG$

Данные, переносимые CG:

ответный APDU

и байты незначащей информации в соответствии с PI.

УДК 336.77:002:006.354 ОКС 35.240.15 Э46 ОКП 40 8470

Ключевые слова: обработка данных, обмен информацией, идентификационные карты, IC-карты, сообщения, способы защиты, аутентификация

Редактор *В.П. Огурцов*
Технический редактор *Н.С. Гришанова*
Корректор *В.И. Варенцова*
Компьютерная верстка *С.В. Рябовой*

Изд. лиц. № 02354 от 14.07.2000. Сдано в набор 04.08.2004. Подписано в печать 03.09.2004. Усл.печ.л. 8,84. Уч.-изд.л. 8,70.
Тираж 244 экз. С 3709. Зак. 771.

ИПК Издательство стандартов. 107076 Москва, Колодезный пер., 14.
e-mail: info@standards.ru
Набрано в Издательстве на ПЭВМ
Отпечатано в филиале ИПК Издательство стандартов — тип. "Московский печатник", 105062 Москва, Лялин пер., 6.
Плр № 080102

gost.e-ecolog.ru