Advanced Card Systems Ltd.
Card & Reader Technologies

ACR1581U

USB デュアルインターフェイス リーダライタ

リファレンスマニュアル　V1.07

事前に告知なく変更される場合があります

info@acs.com.hk
www.acs.com.hk

目次

1.0. 紹介 6

2.0. 特性 7

3.0. ACR1581U の構造 9

3.1. リーダーの機能アーキテクチャ図 9

3.2. PC/SC ドライバが ICC、PICC および SAM 間の通信 9

4.0. ハードウェアのデザイン 11

4.1. USB 11

4.1.1. 通信パラメ―タ― 11

4.1.2. エンドポイント 11

4.2. 接触式スマートカードインターフェース 11

4.2.1. スマートカード電源 VCC（C1） 11

4.2.2. カードタイプのセレクション 11

4.2.3. マイクロコントローラベースカードのためのインターフェース 12

4.3. 非接触スマートカードインターフェース 12

4.3.1. 搬送波周波数 12

4.3.2. ポーリング 12

4.4. ユーザーインターフェース 12

4.4.1. ブザーと LED 12

5.0. ハードウェアデザイン 14

5.1. PCSC API 14

5.1.1. SCardEstablishContext 14

5.1.2. SCardListReaders 15

5.1.3. SCardConnect 16

5.1.4. SCardControl 17

5.1.5. SCardTransmit 19

5.1.6. SCardDisconnect 21

5.1.7. APDU の流れ 22

5.1.8. 直接なコマンドの流れ 23

5.2. 接触式スマートカードプロトコル 24

5.2.1. メモリカード– 1/2/4/8/16 kb I2C カード 24

5.2.2. メモリカード– 32/64/128/256/512/1024 kb I2C カード 27

5.2.3. メモリカード – ATMEL AT88SC153 30

5.2.4. メモリカード – ATMEL AT88SC1608 36

5.2.5. メモリカード – SLE4418/SLE4428/SLE5518/SLE5528 42

5.2.6. メモリカード – SLE4432/SLE4442/SLE5532/SLE5542 49

5.2.7. メモリカード– SLE4406/SLE4436/SLE5536/SLE6636 56
5.2.8. メモリカード – SLE4404 61
5.2.9. ACOS6-SAM カードコマンド 76

5.3. 非接触スマートカード プロトコル 93
5.3.1. ATR の生成 93
5.3.2. APDU、私有（Psedo）APDU およびカード固有（Native）コマンド 97
5.3.3. PICC の PCSC 私有（Pseudo）APDU（独自の拡張機能付き） 97
5.3.4. 透過（Pass Though）コマンド 108
5.3.5. PCSC 2.0 パート 3 サポートできる APDU コマンド（V2.02 及びその以降のバージョン） 114
5.3.6. PICC の専属の私有（Pseudo）APDU 129
5.3.7. PCSC 準拠のタグをアクセスする（ISO 14443-4） 131
5.3.8. サポート PICC ATR 134

6.0. Escape コマンド 137

6.1. PICCEscape コマンド 137
6.1.1. RF 制御（RF Control）[E0 00 00 25 01 …] 137
6.1.2. PCD/PICC 状態取得（Get PCD/PICC Status）[E0 00 00 25 00] 137
6.1.3. ポーリング/ATR オプションの取得（Get Polling/ATR Option）[E0 00 00 23 00] 138
6.1.4. ポーリング/ATR オプションの設定（Set Polling/ATR Option）[E0 00 00 23 01 …] 138
6.1.5. PICC ポーリングタイプ取得（Get PICCPolling Type）[E0 00 01 20 00] 139
6.1.6. PICC ポーリングタイプ設定（Set PICC Polling Type）[E0 00 01 20 02 …] 140
6.1.7. 自動 PPS 取得（Get Auto PPS ）[E0 00 00 24 00] 141
6.1.8. 自動 PPS 設定（Set Auto PPS ）[E0 00 00 24 01…] 141
6.1.9. PICC タイプ取得（Read PICC Type）[E0 00 00 35 00] 142
6.1.10. RF パワー設定取得（Get RF Power Setting）[E0 00 00 50 00] 143
6.1.11. RF パワー設定（Set RF Power Setting）[E0 00 00 50 01 …] 143
6.1.12. 選択的保留設定取得（Get Selective Suspend Setting）[E0 00 00 E5 00] 144
6.1.13. 選択的保留設定設置（Set Selective Suspend Setting）[E0 00 00 E5 01…] 144
6.1.14. PICC– HID キーボードの Escape コマンド 145
6.1.15. PICC-カードシミュレーションの Escape コマンド 153
6.1.16. PICC-検出モードの Escape コマンド 163

6.2. ICCEscape コマンド 165
6.2.1. 専用モード取得（Get Exclusive Mode ）[E0 00 00 2B 00] 165
6.2.2. 専用モード設定（Set Exclusive Mode ）[E0 00 00 2B01 …] 165
6.2.3. カード電源設置取得（Get Card Power Config）[E0 00 00 0B 00] 166
6.2.4. カード電源配置設定（Set Card Power Config）[E0 00 00 0B 01…] 166

6.3. 周辺設備制御と他の Escape コマンド 167
6.3.1. ファームウェアバージョン取得（Get Firmware Version）[E0 00 00 18 …] 167
6.3.2. シリアルナンバー取得（Get Serial Number ）[E0 00 00 33 00] 167
6.3.3. USB 記述子内の S/N を設定する（Set S/N in USB Descriptor）[E0 00 00 F0] 167

6.3.4. ブザー制御の設定-単発（Set Buzzer Control - Single Time）[E0 00 00 28 01 …] 168
6.3.5. ブザー制御の設定-重複（Set Buzzer Control - Repeatable）[E0 00 00 28 03 …] 169
6.3.6. LED 状態取得（Get LED Status）[E0 00 00 29 00] 169
6.3.7. LED 制御設定（Set LED Control）[E0 00 00 29 01 …] 170
6.3.8. UI 操作取得（Get UI Behaviour）[E0 00 00 21 00] 170
6.3.9. UI 操作設定（Set UI Behaviour）[E0 00 00 21 01 …] 171

附録 A. *NDEF* メッセージ *172*

附録 B. *ACR1281U* との互換性 *173*

附録 B.1. 認証キーのダンロード（Load Authentication Keys） 173

附録 B.2. MIFARE Classic ((1K/4K))カードに対しての認証（Authentication for MIFARE Classic ((1K/4K） 175

附録 B.3. 手動 PICC ポーリング（Manual PICC Polling） 176

附録 B.4. PICC 操作パラメ―タ―を読取る（Read PICC Operating Parameter） 177

附録 B.5. PICC 操作パラメ―タ―設定（Set PICC Operating Parameter） 177

附録 B.6. カード挿入カウンタ初期化（Initialize Cards Insertion Counter） 178

附録 B.7. カード挿入カウンタを読み取る（Read Cards Insertion Counter） 178

附録 B.8. カード挿入カウンタ更新（Update CardInsertion Counter） 179

附録 B.9. 互換性と移行説明 180

図示一覧表

図 1 ：ACR1581U リーダーライターの機能ブロック図 9
図 2 ：ACR1581U の構造 10
図 3 ：ACR1581U の APDU 流れ 22
図 4 ：ACR1581U 直接コマンドの流れ 23
図 5 ：ACR1581U 透過セッションフローチャート 114

チャート一覧表

表 1 ： USB インターフェース配線 11
表 2 ：ブザーと LED インジケータ 13
表 3 ：識別子ビット値変更 76
表 4 ：MIFARE Classic 1K カードのメモリマップ 100

表 5 ：MIFARE Classic 4K カードのメモリマップ 101
表 6 ：MIFARE Ultralight カードのメモリマップ 102
表 7 ：NFC フォーラムタイプ 2 ラベルのメモリマップ（2000 バイト） 154
表 8 ：FeliCa カードのメモリマップ（160 バイト） 155

1.0. 紹介

ACR1581DualBoost III は ACR1281U の成功経験を引き継いて、ACS DualBoost シリーズリーダライタの第 3 世代製品である。デュアルインタフェースリーダライタとして、各種接触式と非接触式のスマートカードを読み書きでき、USB CCID 類ドライバ及び USB インタフェースを通じてコンピュータと接続し、コンピュータアプリケーションからカードコマンドを受け取ることができる。従来の ISO 7816 MCU カード、MIFARE®カードおよび ISO 14443 の A クラスと B クラスの非接触カードをサポートする以外、FeliCa カードと ISO 15693 カードにも対応している。

コンピュータとカードの中間デバイスとして、ACR1581U は PC からのコマンドを実行して、非接触タグ、MCU、SAM カードおよび周辺デバイス（LED またはブザー）と通信する。その 3 種類のインタフェース（PICC インタフェース、ICC インタフェース、SAM インタフェース）はすべて PC/SC 規格に適合しており、接触式インタフェースは ISO 7816 規格で定義された APDU コマンドを使用している。接触式 MCU カードの動作については、関連カードのドキュメントおよび PC/SC 規格を参照してください。

この API ファイルでは、非接触インタフェースと接触式メモリカードをサポートするために PC/SC APDU コマンドを実行する方法、および ACR1581U の周辺機器を制御する方法について詳細に説明する。

2.0. 特性

- USB フルスピード·インターフェース
- CCID準拠
- スマートカードリーダー：
  - 非接触インターフェース：
    - 読み書き速度が 26 kbps（ISO 15693 カード）および 848 kbps（ISO 14443 カード）
    - 内蔵アンテナは非接触タグの読み書きに使用され、スマートカードの読み取り距離は 70 mm に達する（タグの種類によって異なる）
    - ISO 15693 カードサポート
    - ISO 14443 パート 4 のタイプ A・B のカードと MIFARE シリーズサポート
    - 衝突防止機能内蔵
    - 拡張の APDU サポート（最大 64 KB）
  - 接触式インターフェース
    - ISO 7816 クラス A 、B、C の（ 5V、3V および 1.8V ）カードサポート
    - 共通の特権カード（CAC）サポート
    - 個人認証カード（PIV）サポート
    - T = 0 または T = 1 プロトコルのマイクロプロセッサー·カードをサポート
    - プロトコルとパラメータの選択サポート（PPS ）
    - 短絡保護保有
    - 拡張 APDU 対応（T=1：最大 64 K バイト、T=0：最大 512+10 バイト）
  - SAM インターフェース：
    - 1つ SAMカードスロット
    - ISO7816クラスA のSAMカードサポート
- アプリケーション プログラミング インターフェース：
  - PC/SC サポート
  - CT- API サポート（ PC/SC 上のレイヤーによるパッケージ）
- 内蔵されている周辺機器：
  - 2×ユーザー制御可能な LED（青と緑）
  - ユーザー制御可能なブザー
- USBファームウェアのアップグレード機能
- Android™ 3.1と以降のバージョンサポート[1]

[1] *ACS 定義された Android ライブラリ使用*

- 以下の規格に準拠：
  - ISO 14443
  - ISO 15693
  - ISO 7816
  - PC/SC
  - CCID
  - CE
  - UKCA
  - FCC
  - RoHS
  - REACH
  - Microsoft® WHQL

3.0. ACR1581U の構造

3.1. リーダーの機能アーキテクチャ図

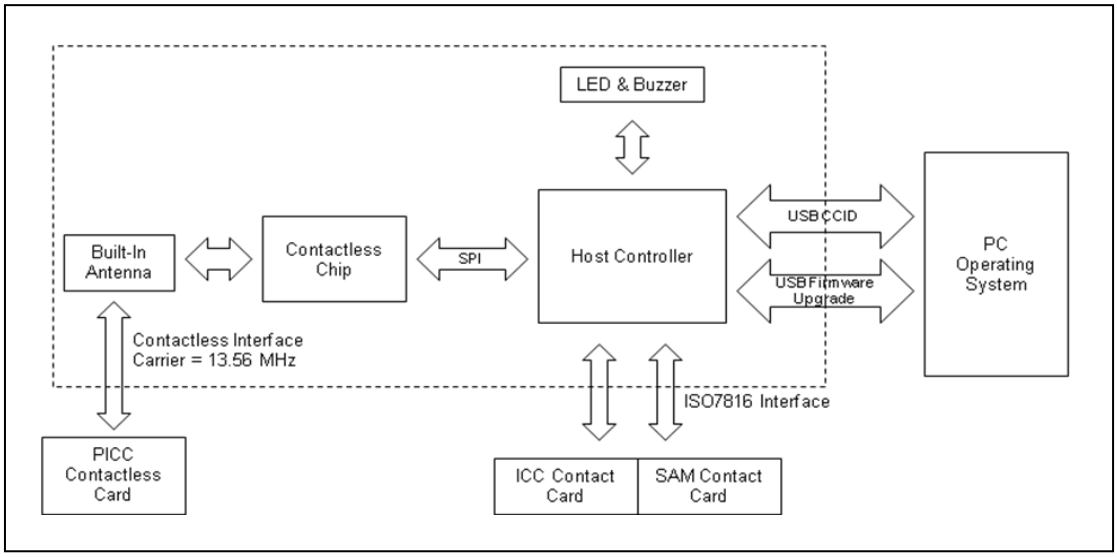

図 1：ACR1581U リーダーライターの機能ブロック図

3.2. PC/SC ドライバが ICC、PICC および SAM 間の通信

ACR1581U は CCID プロトコルを使用して PC と通信する。ICC、PICC と SAM との通信も PC/SC 規格に準拠している。

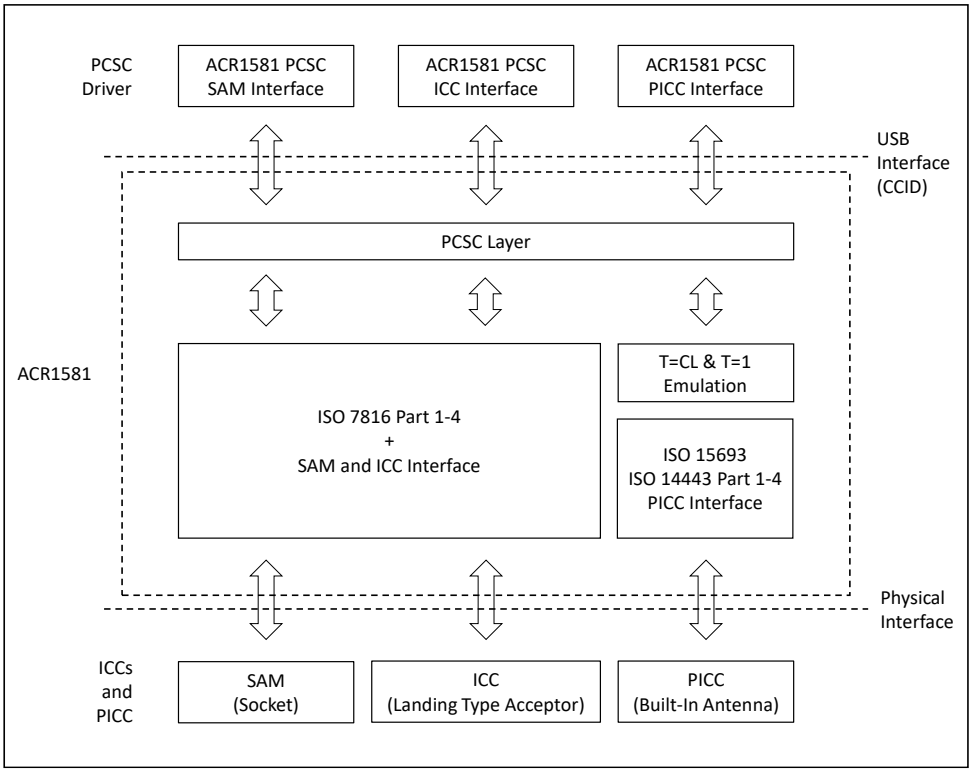

図 2：ACR1581U の構造

4.0. ハードウェアのデザイン

4.1. USB

ACR1581U が USB 規格に準拠した USB インターフェースを介して PC と接続する。

4.1.1. 通信パラメーター

ACR1581U は USB 2.0 規格の要求に従って USB インタフェースを通じてコンピュータと接続を確立する。USB 全速モードをサポートし、レートは 12 Mbps である。

ピン	信号	機能
1	V_{BUS}	カードに+5 V の電源を供給
2	D-	ACR1581U-C1 と PC は差動信号でデータを転送する。
3	D+	ACR1581U-C1 と PC は差動信号でデータを転送する。
4	GND	参考用の電圧レベル

表 1： USB インターフェース配線

注： *ACR1581U が USB インターフェースを介して、正常に動作させるには、まずドライバプログラムをインストールする必要がある。*

4.1.2. エンドポイント

ACR1581U 下記のエンドポイントを介して、ホスト PC と通信する：

Control Endpoint – 設置と制御

Bulk-OUT – ホスト PC から ACR1581U にコマンドを送信する（パケットサイズ 64 バイト）

Bulk-IN – ACR1581U からホスト PC に応答を送信する（パケットサイズ 64 バイト）

Interrupt-IN – ACR1581U からホスト PC にカード状態のメッセージを送信する（パケットサイズ 8 バイト）

4.2. 接触式スマートカードインターフェース

ACR1581U は挿入されたカードの間のインターフェースが ISO 7816-3 仕様プロトコルに準拠して、ACR1581U の実用的な機能性を向上するために、いくつかの制限または向上が行われている。

4.2.1. スマートカード電源 VCC（C1）

挿入されたカードの消費電流は 60mA よりも高くてはならない。

4.2.2. カードタイプのセレクション

制御 PC は、挿入されたカードをアクティする前に、ACR1581U に正しいコマンドを送信してカードタイプを選択する必要がある。メモリカードと MCU ベースカードも含めている。

MCU ベースカードに対して、リーダーが T=0 または T=1 中から望ましいプロトコルを選びます。しかしながら、挿入されたカードは両方のプロトコルタイプをサポートできる場合は、リーダーが PPS を通じて、このセレクションを受け入れられて、実行する。MCU カードは 1 つだけのプロトコルタイプ（T=0 または T=1）をサポートする時に、アプリケーションがどのプロトコルをがデータ選ぶことと関係なく、リーダーは自動的にこのプロトコルタイプを選択する。

4.2.3. マイクロコントローラベースカードのためのインターフェース

マイクロコントローラベースカードは C1（VCC）、C2（RST)、C3（CLK）、C5（GND）および C7（I/O）これらのコンタクトだけ使用する。5 MHz の周波数が CLK 信号（C3）に適用する。

4.3. 非接触スマートカードインターフェース

ACR1581U は非接触カードとの間のインタフェースが ISO 14443 規格に準拠しており、ACR1581U の実用的な機能を強化するために、いくつかの制限または向上が行われている。

4.3.1. 搬送波周波数

ACR1581U-C1 の搬送波周波数は 13.56MHz である。

4.3.2. ポーリング

ACR1581U-C1 は自動に作業場に入る非接触カードを検出する。この機能が ISO 14443-4 の A と B タイプのカード、MIFARE カードをサポートします。

4.4. ユーザーインターフェース

4.4.1. ブザーと LED

ACR1581U には、接触式と非接触インタフェースの状態を示すための LED インジケータと単音ブザーが搭載されている。ブルーの LED は PICC の状態を示し、緑の LED は ICC の状態を示す。

リーダー状態	ブザー	グリーン (ICC)	青色 LED （PICC）
1. リーダー挿入	1 回鳴らす	● >> ● >> ●	
2. スタンバイ（非接触カードポーリング、ICC カードもしくは PICC カードは存在しない）	OFF	OFF	●
3. スタンバイ（ポーリングなし、ICC カードもしくは PICC カードは存在しない）	OFF	OFF	OFF
4. 非接触式カードかざす	1 回鳴らす	ICC 状態に基づく	●
5. 非接触式カード存在	OFF	ICC 状態に基づく	●

リーダー状態	ブザー	グリーン (ICC)	青色 LED (PICC)
6. 非接触式カード取り外す	OFF	ICC 状態に基づく	スタンバイ/ ICC 状態に基づく
7. 非接触式カード通信中	OFF	ICC 状態に基づく	速く点滅する
8. 接続式カードを挿入して	1 回鳴らす	●	OFF/ PICC 状態に基づく
9. 接触カードあり	OFF	●	OFF/ PICC 状態に基づく
10. 接触カードない	OFF	OFF	●
11. 接触カード通信中	OFF	速く点滅する	PICC 状態に基づく

表 2：ブザーと LED インジケータ

5.0. ハードウェアデザイン

5.1. PCSC API

このセクションでは、アプリケーションプログラミング用の PCSC API について説明します。これらの API の詳細について、Microsoft MSDN ライブラリまたは PCSC ワークグループ仕様サイトを参照してください。

5.1.1. SCardEstablishContext

SCardEstablishContext 関数はデータベース操作を実行するリソースマネージャのコンテキストを確立するのためです。

参照してください：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379479%28v=vs.85%29.aspx

他の PCSC アクションを実行する前に、この関数を実行してください。

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT hContext;
int retCode;
void main ()
{
   // To establish the resource manager context and assign it to
 “hContext”
   retCode = SCardEstablishContext(SCARD_SCOPE_USER,
                  NULL,
                  NULL,
                  &hContext);
   if (retCode != SCARD_S_SUCCESS)
   {
      // Establishing resource manager context failed
   }
   else
   {
      // Establishing resource manager context successful
      // Further PCSC operation can be performed
   }
}
```

例：

5.1.2. SCardListReaders

SCardListReaders 関数を使用して、システム内に指定さリーダーライターグループコレクション内のリーダーライターリストを取得します（重複項目排除）。

呼び出し側がリーダーライターグループのリストを提供して、関数が指定されたグループ内のリーダーライターの名前リストを返す。認識できないグループの名前は無視されます。この関数が現在システムに利用できるグループ中のリーダーライターだけ返されます。

参照してください：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379793%28v=vs.85%29.aspx

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT hContext; // Resource manager context
int retCode;
char readerName [256]; // List reader name

void main ()
{
   To establish the resource manager context and assign to “hContext”
   retCode = SCardEstablishContext(SCARD_SCOPE_USER,
                  NULL,
                  NULL,
                  &hContext);
   if (retCode != SCARD_S_SUCCESS)
   {
      // Establishing resource manager context failed
   }
   else
   {
      // Establishing resource manager context successful
      // List the available reader which can be used in the system
      retCode = SCardListReaders (hContext,
                  NULL,
                  readerName,
                  &size);
      if (retCode != SCARD_S_SUCCESS)
      {
         // Listing reader fail
      }
      if (readerName == NULL)
      {
         // No reader available
      }
      else
      {
         // Reader listed
      }
   }
}
```

例：

5.1.3. SCardConnect

SCardConnect 関数は特定のエクスプローラコンテキストを使用して、アプリケーションと特定のリーダーライターに含まれるスマートカードとの間の接続を確立します。特定のリーダライターの中にカードがない場合、エラーメッセージが返されます。

参照してください：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379473%28v=vs.85%29.aspx

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT       hContext;          // Resource manager context
SCARDHANDLE        hCard;             // Card context handle
unsigned long      dwActProtocol;     // Establish active protocol
int                retCode;
char               readerName [256]; // List reader name
char               rName [256];       // Reader name for connection

void main ()
{
   …
      if (readerName == NULL)
      {
         // No reader available
      }
      else
      {
         // Reader listed
      rName = "ACS ACR1581 1S Dual Reader PICC 0"; // Depends on what
                                                   reader be used
                                                   // Should connect to
                                                   PICC interface
         retCode = SCardConnect(hContext,
               rName,
               SCARD_SHARE_SHARED,
               SCARD_PROTOCOL_T0,
               &hCard,
               &dwActProtocol);
         if (retCode != SCARD_S_SUCCESS)
         {
            // Connection failed (May be because of incorrect reader
         name, or no card was detected)
         }
         else
         {
            // Connection successful
         }
      }
   }
```

例：

5.1.4. SCardControl

SCarControl 関数は、リーダーライターへの直接制御を提供し、SCarConnect 関数が成功して呼び出られた後、SCarDisconnect 関数が成功して呼び出られる前にいつでも SCarControl 関数を呼び出すことができます。リーダーライターの状態への影響は制御コードに依存します。

参照してください：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379474%28v=vs.85%29.aspx

注釈：*Escape* コマンドのはこの *API* で送信する必要がある。

例：

```
#define SCARD_SCOPE_USER    0

#define EscapeCommand 0x310000 + 3500*4
   SCARDCONTEXT          hContext;          // Resource manager context
   SCARDHANDLE           hCard;             // Card context handle
   unsigned long         dwActProtocol;     // Established active protocol
   int                   retCode;
   char                  readerName [256]; // Lists reader name
   char                  rName [256];       // Reader name for connection
   BYTE                  SendBuff[262],     // APDU command buffer
                         RecvBuff[262];     // APDU response buffer
   BYTE                  FWVersion [20],    // For storing firmware
                                               version message
   BYTE                  ResponseData[50]; // For storing card response
   DWORD                 SendLen,           // APDU command length
                         RecvLen;           // APDU response length
void main ()
   {
      …
      rName = "ACS ACR1581 1S Dual Reader PICC 0";  // Depends on what
                                                      reader will be used
                                                      // Should connect to
                                                      PICC interface

         retCode = SCardConnect(hContext,
            rName,
            SCARD_SHARE_DIRECT,
            SCARD_PROTOCOL_T0| SCARD_PROTOCOL_T1,
            &hCard,
            &dwActProtocol);
         if (retCode != SCARD_S_SUCCESS)
         {
            // Connection failed (may be because of incorrect reader
         name, or no card was detected)
         }
         else
         {
            // Connection successful
            RecvLen = 262;
            // Get firmware version
            SendBuff[0] = 0xE0;
            SendBuff[1] = 0x00;
            SendBuff[2] = 0x00;
            SendBuff[3] = 0x18;
            SendBuff[4] = 0x00;
```

```
            SendLen = 5;
            retCode = SCardControl ( hCard,
                      EscapeCommand,
                      SendBuff,
                      SendLen,
                      RecvBuff,
                      RecvLen,
                      &RecvLen);
            if (retCode != SCARD_S_SUCCESS)
            {
               // APDU sending failed
               return;
            }
            else
            {
               // APDU sending successful
               // The RecvBuff stores the firmware version message.
               for (int i=0;i< RecvLen-5;i++)
               {
                  FWVersion[i] = RecvBuff [5+i];
               }
            }
            // Connection successful
            RecvLen = 262;

            // Turn Green LED on, turn Red LED off
            SendBuff[0] = 0xE0;
            SendBuff[1] = 0x00;
            SendBuff[2] = 0x00;
            SendBuff[3] = 0x29;
            SendBuff[4] = 0x01;
            SendBuff[5] = 0x02;  // Green LED On, Red LED off
            SendLen = 6;
            retCode = SCardControl ( hCard,
                      EscapeCommand,
                      SendBuff,
                      SendLen,
                      RecvBuff,
                      RecvLen,
                      &RecvLen);
            if (retCode != SCARD_S_SUCCESS)
            {
               // APDU sending failed
               return;
            }
            else
            {
               // APDU sending success
            }
```

5.1.5. SCardTransmit

SCardTransmit 関数はサービス請求をスマートカードに送信するために、またはスマートカードから返されるデータを受信するために使われます。

参照してください：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379804%28v=vs.85%29.aspx

注： この *API* で *APDU* コマンド（即ち：接続が確立されたカードに送信するコマンド、***PICC の PCSC 私有（Pseudo）APDU（独自の拡張機能付き）*** と ***PCSC 2.0 パート 3 の例***

1. 透明セッション開始する

   コマンド：**FF C2 00 00 02 81 00**

   応答：**C0 03 00 90 00 90 00**

2. アンテナフィールドをオフにする

   コマンド：**FF C2 00 00 02 83 00**
   応答：**C0 03 00 90 00 90 00**

3. アンテナフィールドをオンにする

   コマンド：**FF C2 00 00 02 84 00**

   応答：**C0 03 00 90 00 90 00**

4. アクティベーISO 14443-4A

   コマンド： **FF C2 00 02 04 8F 02 00 04**

   応答： **C0 03 01 64 01 90 00** （カードがない場合）
   C0 03 00 90 00 5F 51 [Len] [ATR] 90 00

5. 0AH に PCB を設定し、送信データで CRC、パリティ、プロトコルプロローグを有効にします。

   コマンド：**FF C2 00 01 0A 90 02 00 00 FF 6E 03 07 01 0A**

   応答：**C0 03 00 90 00 90 00**

6. カードに APDU「80B2000008」を送信し、応答を取得します。

   コマンド：**FF C2 00 01 0E 5F 46 04 40 42 0F 00 95 05 80 B2 00 00 08**

   応答：**C0 03 00 90 00 92 01 00 96 02 00 00 97 0C [カードの応答] 90 00**

7. 透明セッションを終了する。

   コマンド：**FF C2 00 00 02 82 00**

   応答：**C0 03 00 90 00 90 00**

PICC の専属の私有（Pseudo）APDU）を送信する。

例：

```
#define SCARD_SCOPE_USER      0

SCARDCONTEXT         hContext;          // Resource manager context
SCARDHANDLE          hCard;             // Card context handle
unsigned long        dwActProtocol;     // Established active protocol
int                  retCode;
char                 readerName [256]; // List reader name
char                 rName [256];       // Reader name for connect
BYTE                 SendBuff[262],     // APDU command buffer
                     RecvBuff[262];     // APDU response buffer
BYTE                 CardID [8],        // For storing the FeliCa IDM/
                                        MIFARE UID
BYTE                 ResponseData[50]; // For storing card response
DWORD                SendLen,           // APDU command length
                     RecvLen;           // APDU response length
SCARD_IO_REQUEST     ioRequest;

void main ()
{
   ...
   rName = "ACS ACR1581 1S Dual Reader PICC 0"; // Depends on what
                                                  reader should be used
                                                  // Should connect to PICC
                                                  interface
      retCode = SCardConnect(hContext,
                     rName,
                     SCARD_SHARE_SHARED,
                     SCARD_PROTOCOL_T0,
                     &hCard,
                     &dwActProtocol);
      if (retCode != SCARD_S_SUCCESS)
      {
         // Connection failed (May be because of incorrect reader
         name, or no card was detected)
      }
      else
      {
         // Connection successful
         ioRequest.dwProtocol = dwActProtocol;
         ioRequest.cbPciLength = sizeof(SCARD_IO_REQUEST);
         RecvLen = 262;
```

```
// Get MIFARE UID/ FeliCa IDM
SendBuff[0] = 0xFF;
SendBuff[1] = 0xCA;
SendBuff[2] = 0x00;
SendBuff[3] = 0x00;
SendBuff[4] = 0x00;
SendLen = 5;
   retCode = SCardTransmit( hCard,
                  &ioRequest,
                  SendBuff,
                  SendLen,
                  NULL,
                  RecvBuff,
                  &RecvLen);

if (retCode != SCARD_S_SUCCESS)
{
   // APDU sending failed
   return;
}
else
{
   // APDU sending successful
   // The RecvBuff stores the IDM for FeliCa / the UID for
   MIFARE.
   // Copy the content for further FeliCa access
   for (int i=0;i< RecvLen-2;i++)
   {
      CardID [i] = RecvBuff[i];
   }
}
```

5.1.6. SCardDisconnect

SCardDisconnect 関数は前に確立されたアプリケーションとターゲットリーダー間の接続を終了するためです。

参照してください：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379475%28v=vs.85%29.aspx

この関数が PCSC 操作をエンドするために使用されます。

例：

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT          hContext;          // Resource manager context
SCARDHANDLE        hCard;             // Card context handle
unsigned long      dwActProtocol;     // Established active protocol
int                retCode;

void main ()
{
   …
         // Connection successful
      …
      retCode = SCardDisconnect(hCard,SCARD_RESET_CARD);
      if (retCode != SCARD_S_SUCCESS)
      {
         // Disconnection failed
      }
      else
      {
         // Disconnection successful
      }
   }
}
```

5.1.7. APDU の流れ

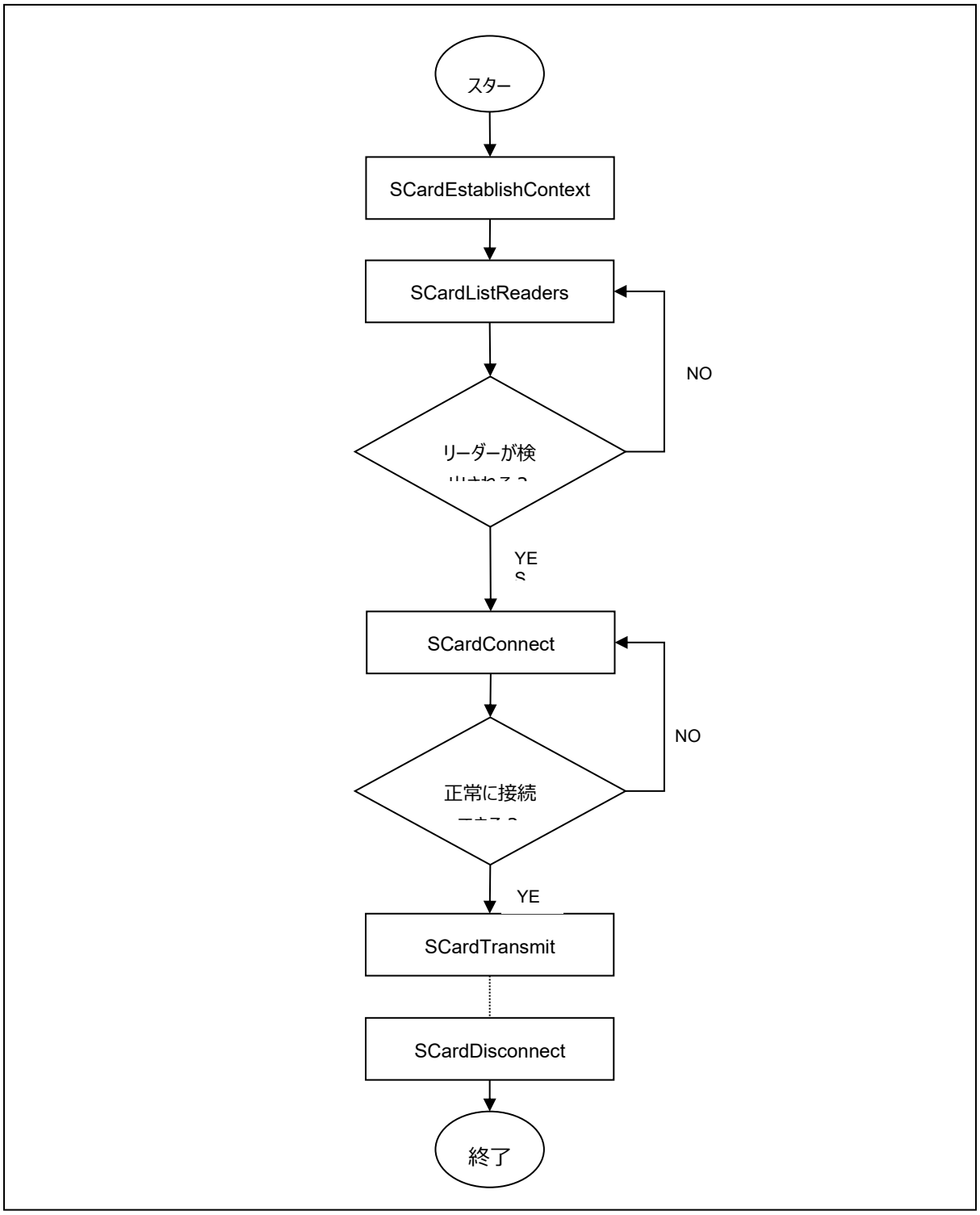

図 3：ACR1581U の APDU 流れ

5.1.8. 直接なコマンドの流れ

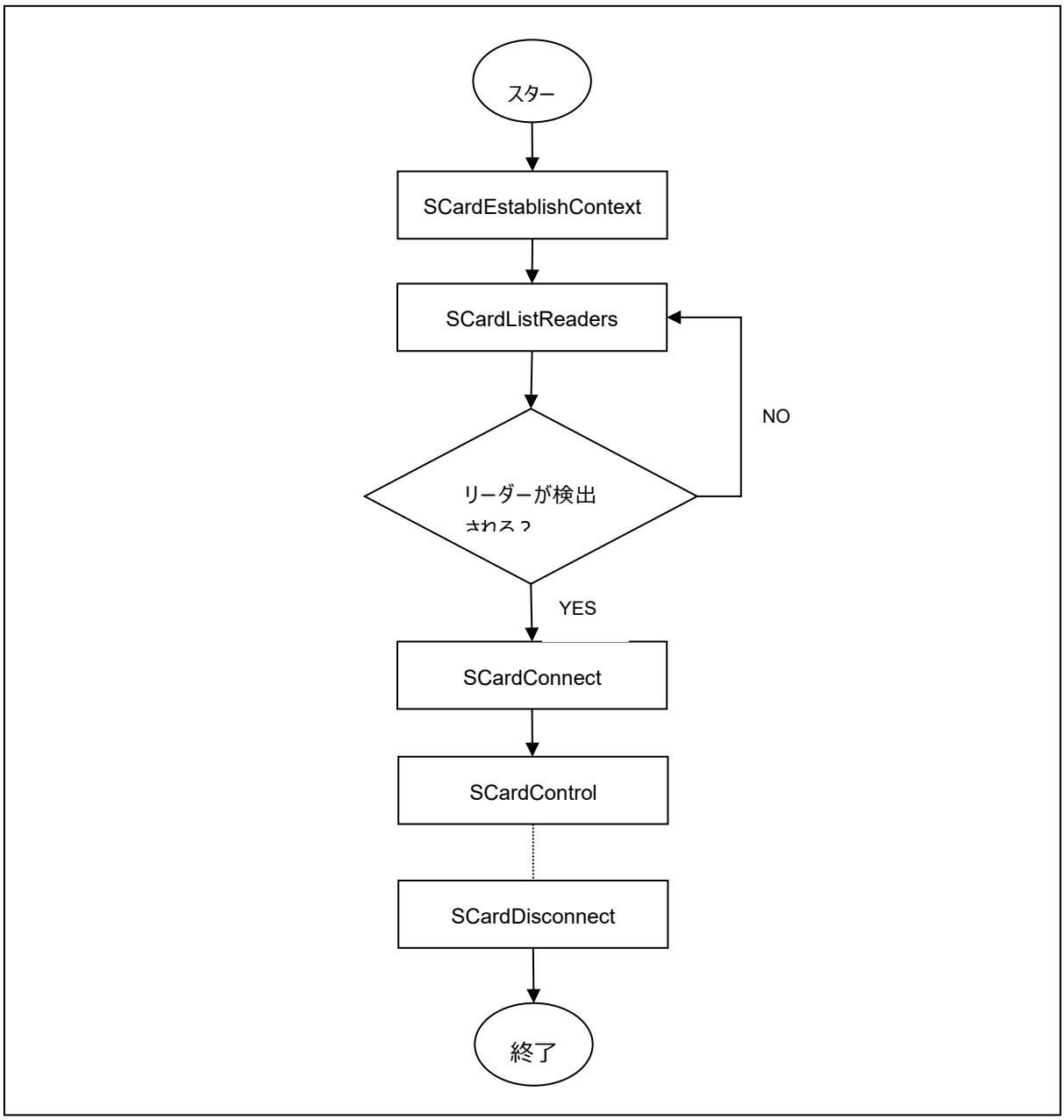

図 4：ACR1581U 直接コマンドの流れ

5.2. 接触式スマートカードプロトコル

5.2.1. メモリカード– 1/2/4/8/16 kb I2C カード

5.2.1.1. カードタイプ選択（Select Card Type）

このコマンドはカードリーダーに挿入されて、選択されたカードにパワーダウン/アップを実行し、同時にリセットを実行する時に使われる。

コマンド

コマンド	CLA	INS	P1	P2	Lc	カードタイプ
Select Card Type	FFh	A4h	00h	00h	01h	01h

応答コード

結果	SW1	SW2	意味
成功	90h	00h	操作が成功に完了しました。

5.2.1.2. ページサイズ選択（Select Page Size）

このコマンドで読み取られるページサイズを選択する。デフォルト値は 8 バイトの書き込みページ。カードが取り出されて、リーダーの電源がオフになっている時に、デフォルト値にリセットされる。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Select Page Size	FFh	01h	00h	00h	01h	ページサイズ

応答コード

結果	SW1	SW2	意味
成功	90h	00h	操作が成功に完了しました。

ページサイズ：1 バイト

状態	説明
03h	8バイトの書き込みページ
04h	16バイトの書き込みページ
05h	32バイトの書き込みページ

状態	説明
06h	64バイトの書き込みページ
07h	128バイトの書き込みページ

5.2.1.3. メモリカード読み取る（Read Memory Card）

このコマンドは、指定したアドレス位置からメモリカードの内容を読み込む時に使われる。

コマンド

コマンド	CLA	INS	P1	P2	Le
Read Memory Card	FFh	B0h	メモリアドレス		長さ

応答コード

結果	SW1	SW2	意味
成功	90h	00h	操作が成功に完了しました。

5.2.1.4. メモリカード書き込む（Write Memory Card）

このコマンドは、指定したアドレス位置からメモリカードに書き込む時に使われる。

コマンド

コマンド	CLA	INS	P1	P2	Le	コマンドデータイン
Write Memory Card	FFh	D0h	メモリアドレス		長さ	データ

応答コード

結果	SW1	SW2	意味
成功	90h	00h	操作が成功に完了しました。

5.2.2. メモリカード– 32/64/128/256/512/1024 kb I2C カード

5.2.2.1. Select Card Type

このコマンドはカードリーダーに挿入されて、選択されたカードにパワーダウン/アップを実行し、同時にリセットを実行する時に使われる。

コマンド

コマンド	CLA	INS	P1	P2	Lc	カードタイプ
Select Card Type	FFh	A4h	00h	00h	01h	02h

応答コード

結果	SW1	SW2	意味
成功	90h	00h	操作が成功に完了しました。

5.2.2.2. ページサイズ選択（Select Page Size）

このコマンドで読み取られるページサイズを選択する。デフォルト値は 8 バイトの書き込みページ。カードが取り出されて、リーダーの電源がオフになっている時に、デフォルト値にリセットされる。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Select Page Size	FFh	01h	00h	00h	01h	ページサイズ

応答コード

結果	SW1	SW2	意味
成功	90h	00h	操作が成功に完了しました。

ページサイズ：1 バイト

状態	説明
03h	8バイトの書き込みページ
04h	16バイトの書き込みページ
05h	32バイトの書き込みページ
06h	64バイトの書き込みページ

状態	説明
07h	128バイトの書き込みページ

5.2.2.3. メモリカード読み取る（Read Memory Card）

このコマンドは、指定したアドレス位置からメモリカードの内容を読み込む時に使われる。

コマンド

コマンド	CLA	INS	P1	P2	Le
Read Memory Card	FFh		メモリアドレス		長さ

応答コード

結果	SW1	SW2	意味
成功	90h	00h	操作が成功に完了しました。

注釈： ***INS*** *B0h = 32, 64, 128, 256, 512 kb I2C カード用*

*1011 000*b；その中 *は 17 ビットのアドレスの MSB = 1024 kb I2C カード用*

5.2.2.4. メモリカード書き込む（Write Memory Card）

このコマンドは、指定したアドレス位置からメモリカードに書き込む時に使われる。

コマンド

コマンド	CLA	INS	P1	P2	Le	コマンドデータイン
Write Memory Card	FFh		メモリアドレス		長さ	データ

応答コード

結果	SW1	SW2	意味
成功	90h	00h	操作が成功に完了しました。

注釈：*INS* *B0h = 32, 64, 128, 256, 512 kb I2C カード用*

*1011 000*b；その中 *は 17 ビットのアドレスの MSB = 1024 kb I2C カード用*

5.2.3. メモリカード – ATMEL AT88SC153

5.2.3.1. カードタイプ選択（Select card type）

このコマンドは選択されて、カードリーダーに挿入されたカードにパワーダウン/アップを実行する。同時にリセットを実行する時に使われる。また 8 バイトページの書込みサイズを選択しする。

コマンド

疑似 APDU						
コマンド	CLA	INS	P1	P2	Lc	カードタイプ
Select Card Type	FFh	A4h	00h	00h	01h	03h

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.3.2. メモリカード読み取る（Read memory card）

このコマンドは、指定したアドレス位置からメモリカードの内容を読み込む時に使われる。

コマンド

疑似 APDU					
コマンド	CLA	INS	P1	バイトアドレス	MEM_L
Read Memory Card	FFh		00h		

その中：

INS （1 バイト）

パーティション 00b 読み取る、INS=B0 h

パーティション 01b 読み取る、INS=B1h

パーティション 10b 読み取る、INS=B2h

パーティション 11b 読み取る、INS=B3h

ヒューズマーク読み取る、INS=B4h

バイトアドレス （1 バイト）

メモリカードのメモリアドレス位置。

MEM_L （1 バイト）

メモリカードから読み出したいデータの長さ

応答

応答	バイト 1	…	…	バイト N	SW1	SW2
結果						

その中：

バイト(1…N) メモリカードから読み出されたデータ

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.3.3. メモリカード書き込む（Write memory card）

このコマンド、指定したアドレス位置からメモリカードに書き込む時に使われる。

コマンド

疑似 APDU									
コマンド	CLA	INS	P1	バイトアドレス	MEM_L	バイト 1	…	…	バイト N
Write Memory Card	FFh		00h						

その中：

INS （1 バイト）

パーティション 00b 読み取る、INS=D0h

パーティション 01b 読み取る、INS=D1h

パーティション 10b 読み取る、INS=D2h

パーティション 11b 読み取る、INS=D3h

ヒューズマーク読み取る、INS=B4h

バイトアドレス （1 バイト）

メモリカードのメモリアドレス位置。

MEM_L （1 バイト）

メモリカードに書き込みたいデータの長さ

バイト(1…N) メモリカードに書き込みたいデータ

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.3.4. パスワード検証（Verify password）

このコマンドは、ユーザーが入力した PIN がメモリカードのパスワードと同じであるかどうかを確認するために使用される。

コマンド

疑似 APDU									
コマンド	CLA	INS	P1	P2	Lc	RP	PW (0)	PW (1)	PW (2)
Verify Password	FFh	20h	00h		03h				

その中：

PW(0),PW(1),PW(2) メモリカードに送信したいパスワード

P2 （1 バイト）

= 0000 00r pb

ここで、「rp」2 ビットの識別は比較したいパスワードである

r = 0 ：パスワードを書く

r = 1 ：パスワードを読み取る

p=パスワードセット番号

r p = 01b：安全なパスワード

応答

応答	データ出力	
結果	SW1	ErrorCnt

その中：

SW1 = 90h

ErrorCnt （1 バイト）

=エラーカウンタ

FFh は検証が正しいことを示している。00 h はパスワードがロックされていることを示している（最大再試行回数を超過した）。他の値は現在の認証が失敗したことを示している。

5.2.3.5. 認証初期化（Initialize authentication）

このコマンドは、カード認定を初期化するために使用される。

コマンド

疑似 APDU									
コマンド	CLA	INS	P1	P2	Lc	Q(0)	Q(1)	…	Q(7)
Initialize Authentication	FFh	84h	00h	00h	08h				

その中：

Q (0…7) （8 バイト）

=ホスト乱数

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.3.6. 認証検証（Verify authentication）

このコマンドは、カード認証を検証するために使用される。

コマンド

疑似 APDU									
コマンド	CLA	INS	P1	P2	Lc	Ch (0)	Ch (1)	…	Ch (7)
Verify Authentication	FFh	82h	00h	00h	08h				

その中：

Ch (0…7) （8 バイト）

=ホスト挑戦数

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.4. メモリカード – ATMEL AT88SC1608

5.2.4.1. カードタイプ選択（Select card type）

このコマンドは選択されて、カードリーダーに挿入されたカードにパワーダウン/アップを実行する。同時にリセットを実行する時に使われる。また 16 バイトページの書込みサイズを選択しする。

コマンド

疑似 APDU						
コマンド	CLA	INS	P1	P2	Lc	カードタイプ
Select Card Type	FFh	A4h	00h	00h	01h	04h

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.4.2. メモリカード読み取る（Read memory card）

このコマンドは、指定したアドレス位置からメモリカードの内容を読み込む時に使われる。

コマンド

疑似 APDU					
コマンド	CLA	INS	ゾーンアドレス	バイトアドレス	MEM_L
Read Memory Card	FFh				

その中：

INS （1 バイト）

ユーザーゾーンを読み取る、INS=B0h

構成ゾーンまたはヒューズマーク読み取る、INS=B1h

ゾーンアドレス （1 バイト）

= 00000 A10 A9 A8b、ここで A10 はパーティションアドレスの MSB である

** ヒューズマーク読み取ると関係ない

バイトアドレス （1 バイト）

= A7 A6 A5 A4 A3 A2 A1 A0b はメモリカードのメモリアドレス位置である

ヒューズマーク読み取る時、バイトアドレス = 1000 0000b

MEM_L （1 バイト）

メモリカードから読み出したいデータの長さ

応答

応答	バイト 1	…	…	バイト N	SW1	SW2
結果						

その中：

バイト(1…N) メモリカードから読み出されたデータ

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.4.3. メモリカード書き込む（Write tomemory card）

このコマンド、指定したアドレス位置からメモリカードに書き込む時に使われる。

コマンド

疑似 APDU									
コマンド	CLA	INS	ゾーンアドレス	バイトアドレス	MEM_L	バイト 1	…	…	バイト N
Write Memory Card	FFh								

その中：

INS （1 バイト）

ユーザーゾーンを読み取る、**INS=D0h**

構成ゾーンまたはヒューズマーク読み取る、**INS=D1h**

ゾーンアドレス （1 バイト）

= 00000 A10 A9 A8b、ここで A10 はパーティションアドレスの MSB である

** ヒューズマーク読み取ると関係ない

バイトアドレス （1 バイト）

= A7 A6 A5 A4 A3 A2 A1 A0b はメモリカードのメモリアドレス位置である

ヒューズマーク読み取る時、バイトアドレス = 1000 0000b

MEM_L （1 バイト）

メモリカードに書き込みたいデータの長さ

バイト(1…N) メモリカードに書き込みたいデータ

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.4.4. パスワード検証（Verify password）

このコマンドは、ユーザーが入力した PIN がメモリカードのパスワードと同じであるかどうかを確認するために使用される。

コマンド

疑似 APDU									
コマンド	CLA	INS	P1	P2	Lc	RP	PW (0)	PW (1)	PW (2)
Verify Password	FFh	20h	00h	00h	04h				

その中：

PW(0),PW(1),PW(2) メモリカードに送信したいパスワード

RP （1 バイト）

= 0000 r p2 p1 p0b

ここで、“r p2 p1 p0”2 ビットの識別は比較したいパスワードである

r = 0 ：パスワードを書く込む

r = 1 ：パスワードを読み取る

p2 p1 p0 =パスワードセット番号

r p2 p1 p0 = 0111b：安全なパスワード。

応答

応答	データ出力	
結果	SW1	ErrorCnt

その中：

SW1 = 90h

ErrorCnt （1 バイト）

=エラーカウンタ

FFh は検証が正しいことを示している。00 h はパスワードがロックされていることを示している（最大再試行回数を超過した）。他の値は現在の認証が失敗したことを示している。

5.2.4.5. 認証初期化（Initialize authentication）

このコマンドは、カード認定を初期化するために使用される。

コマンド

疑似 APDU									
コマンド	CLA	INS	P1	P2	Lc	Q(0)	Q(1)	…	Q(7)
Initialize Authentication	FFh	84h	00h	00h	08h				

その中：

Q (0…7) （8 バイト）

=ホスト乱数

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.4.6. 認証検証（Verify authentication）

このコマンドは、カード認証を検証するために使用される。

コマンド

疑似 APDU									
コマンド	CLA	INS	P1	P2	Lc	Ch (0)	Ch (1)	…	Ch (7)
Verify Authentication	FFh	82h	00h	00h	08h				

その中：

Ch (0…7) （8 バイト）

=ホスト挑戦数

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.5. メモリカード – SLE4418/SLE4428/SLE5518/SLE5528

5.2.5.1. カードタイプ選択（Select card type）

このコマンドはカードリーダーに挿入されて、選択されたカードにパワーダウン/アップを実行し、同時にリセットを実行する時に使われます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	カードタイプ
Select Card Type	FFh	A4h	00h	00h	01h	05h

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.5.2. メモリカード読み取る（Read memory card）

このコマンドは、指定したアドレス位置からメモリカードの内容を読み込む時に使われる。

コマンド

コマンド	CLA	INS	バイトアドレス		MEM_L
			MSB	LSB	
Read Memory Card	FFh	B0h			

その中：

MSB バイトアドレス （1 バイト）

= 0000 00 A9 A8b はメモリカードのメモリアドレスである

LSB バイトアドレス （1 バイト）

= A7 A6 A5 A4 A3 A2 A1 A0b はメモリカードのメモリアドレス位置である

MEM_L （1 バイト）

メモリカードから読み出したいデータの長さ

応答

応答	バイト 1	…	…	バイト N	SW1	SW2
結果						

その中：

バイト(1…N) メモリカードから読み出されたデータ

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.5.3. 入力エラーカウンタ読み取る（Read presentation error counter memory card）（SLE4428 と SLE5528 のみ）

このコマンドがプレゼンテーションエラーカウンタを読み取る時に使われる。

コマンド

コマンド	CLA	INS	P1	P2	MEM_L
Read Presentation Error Counter	FFh	B1h	00h	00h	03h

応答

応答	ErrCnt	仮想数 1	仮想数 2	SW1	SW2
結果					

その中：

ErrCnt （1 バイト）

パスワード入力エラーカウンタの値

FFh=検証が正しい

00H=パスワードがロックされている（最大再試行回数を超過した）

他の値は認証が失敗したことを示す。

仮想値 1、仮想値 2 （2 バイト）

カードから読み取られた仮想データ

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.5.4. プロテクションビット読み取る（Read protection bit）

このコマンドは保護ビットを読み取る時に使われる。

コマンド

コマンド	CLA	INS	バイトアドレス		MEM_L
			MSB	LSB	
Read Protection Bit	FFh	B2h			

その中：

MSB バイトアドレス （1 バイト）

メモリカードのメモリアドレス位置

= 0000 00 A9 A8b

LSB バイトアドレス （1 バイト）

メモリカードのメモリアドレス位置

= A7 A6 A5 A4 A3 A2 A1 A0b

MEM_L （1 バイト）

カードから読み出された保護ビットの長さは 8 ビットの倍数で、最大は 32 である。

MEM_L = 1 + INT((ビット数 – 1)/8)

例えば、メモリ 0010H から始まりの 8 保護ビットを読み取るために、下記の pseudo-APDU を発行する必要がある：

FF B1 00 10 01h

応答

応答	PROT 1	…	…	PROT L	SW1	SW2
結果						

その中：

PROT(1..L) 保護ビットが含めているバイト

SW1 SW2 = 90 00h （操作が成功に完了であれば）

PROT バイト中で、保護ビットは以下のように並べている：

PROT 1								PROT 2								....							
P8	P7	P6	P5	P4	P3	P2	P1	P16	P15	P14	P13	P12	P11	P10	P9	..	..	..	..	..	..	P18	P17

その中：

Px は応答データの BYTE *x* の保護ビットである：

0=バイトの書き込みが保護されている

1=バイトは書き込むことができる

5.2.5.5. メモリカード書き込む（Write memory card）

このコマンド、指定されたアドレスにメモリカードを書き込む時に使われる。

コマンド

コマンド	CLA	INS	バイトアドレス		MEM_L	バイト 1	…	…	バイト N
			MSB	LSB					
Write Memory Card	FFh	D0h							

その中：

MSB バイトアドレス （1 バイト）

= 0000 00 A9 A8b はメモリカードのメモリアドレスである

LSB バイトアドレス （1 バイト）

= A7 A6 A5 A4 A3 A2 A1 A0b はメモリカードのメモリアドレス位置である

MEM_L （1 バイト）

メモリカードに書き込みたいデータの長さ

バイト(1…N) メモリカードに書き込みたいデータ

5.2.5.6. 保護メモリカード書き込む（Write tomemory card）

このコマンドは、コマンドで指定されたバイトごとを特定のアドレスに格納されたバイトと比較し、データが一致すると、対応する保護ドビットが不可逆的に「0」に設定される。

コマンド

コマンド	CLA	INS	バイトアドレス		MEM_L	バイト 1	…	…	バイト N
			MSB	LSB					
Write Protection Memory Card	FFh	D1h							

その中：

MSB バイトアドレス （1 バイト）

= 0000 00 A9 A8b はメモリカードのメモリアドレスである

LSB バイトアドレス （1 バイト）

= A7 A6 A5 A4 A3 A2 A1 A0b はメモリカードのメモリアドレス位置である

MEM_L （1 バイト）

=メモリカードに書き込みたいデータの長さ

バイト(1…N)

=カード内のバイトアドレスから始まるデータと比較するバイト値。BYTE 1 とバイトアドレス中のデータを比べる；BYTE N とバイトアドレス + N -1 のデータが比べる。

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.5.7. メモリカードパスワード提示（Present code memory card）（SLE44428 と SLE5528 のみ）

このコマンドは、SLE4428 カードと SLE5528 カードの書き込みを有効にするためにメモリカードにパスワードを提示するために使用される。以下のアクションが実行される：

1. パスワード入力エラーカウンタにビット『1』を検索して、“0”に変更する。
2. 指定されたパスワードをカードに提示する。
3. プレゼンテーションエラーカウンタを消去する。

コマンド

コマンド	CLA	INS	P1	P2	MEM_L	パスワード	
						バイト 1	バイト 2
Present Code Memory Card	FFh	20h	00h	00h	02h		

その中：

パスワード （3 バイト）

パスワード（PIN）

応答

応答	データ出力	
結果	90h	ErrorCnt

その中：

ErrorCnt （1 バイト）

エラーカウンタ

FFh = 検証が正しい。

00H = パスワードがロックされている（最大再試行回数を超過した）。

他の値は認証が失敗したことを示す。

5.2.6. メモリカード – SLE4432/SLE4442/SLE5532/SLE5542

5.2.6.1. カードタイプ選択（Select card type）

このコマンドはカードリーダーに挿入されて、選択されたカードにパワーダウン/アップを実行し、同時にリセットを実行する時に使われます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	カードタイプ
Select Card Type	FFh	A4h	00h	00h	01h	06h

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.6.2. メモリカード読み取る（Read memory card）

このコマンドは、指定したアドレス位置からメモリカードの内容を読み込む時に使われる。

コマンド

コマンド	CLA	INS	P1	バイトアドレス	MEM_L
Read Memory Card	FFh	B0h	00h		

その中：

バイトアドレス （1 バイト）

= A7 A6 A5 A4 A3 A2 A1 A0b はメモリカードのメモリアドレス位置である

MEM_L （1 バイト）

メモリカードから読み出したいデータの長さ

応答

応答	バイト 1	…	…	バイト N	PROT1	PROT2	PROT3	PROT4	SW1	SW2
結果										

その中：

バイト(1…N) メモリカードから読み出されたデータ

PROT(1…4) 保護ビットが含めているバイト

SW1 SW2 = 90 00h （操作が成功に完了であれば）

PROT バイト中で、保護ビットは以下のように並べている：

PROT 1								PROT 2								....							
P8	P7	P6	P5	P4	P3	P2	P1	P16	P15	P14	P13	P12	P11	P10	P9	..	..	..	..	..	..	P18	P17

その中：

Px は応答データの BYTE *x* の保護ビットである：

0=バイトの書き込みが保護されている

1=バイトは書き込むことができる

5.2.6.3. 入力エラーカウンタ読み取る（Read presentation error counter memory card）（SLE4442 と SLE5542 のみ）

このコマンドがプレゼンテーションエラーカウンタを読み取る時に使われる。

コマンド

コマンド	CLA	INS	P1	P2	MEM_L
Read Presentation Error Counter	FFh	B1h	00h	00h	04h

応答

応答	ErrCnt	仮想数 1	仮想数 2	仮想数 3	SW1	SW2
結果						

その中：

ErrCnt （1 バイト）

パスワード入力エラーカウンタの値

07h = 検証が正しい

00H = パスワードがロックされている（最大再試行回数を超過した）

他の値は認証が失敗したことを示す。

仮想値 1、仮想値 1、仮想値 2 (3 バイト)

カードから読み取られた仮想データ

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.6.4. プロテクションビット読み取る（Read Protection Bit）

このコマンドは始めの 32 バイトの保護ビットを読み取る時に使われる。

コマンド

コマンド	CLA	INS	P1	P2	MEM_L
Read Protection Bit	FFh	B2h	00h	00h	04h

応答

応答	PROT 1	PROT 2	PROT 3	PROT 4	SW1	SW2
結果						

その中：

PROT (1..4) 保護ビットが含めているバイト

SW1 SW2 = 90 00h （操作が成功に完了であれば）

PROT バイト中で、保護ビットは以下のように並べている：

PROT 1								PROT 2								....							
P8	P7	P6	P5	P4	P3	P2	P1	P16	P15	P14	P13	P12	P11	P10	P9	..	..	..	..	..	..	P18	P17

その中：

Px 応答データの BYTE の保護ビット：

0 = バイトの書き込みが保護されている

1 = バイトは書き込むことができる

5.2.6.5. メモリカード書き込む（Write memory card）

このコマンド、指定されたアドレスにメモリカードを書き込む時に使われる。

コマンド

コマンド	CLA	INS	P1	バイトアドレス	MEM_L	バイト 1	…	…	バイト N
Write Memory Card	FFh	D0h	00h						

その中：

バイトアドレス （1 バイト）

= A7 A6 A5 A4 A3 A2 A1 A0b はメモリカードのメモリアドレス位置である

MEM_L （1 バイト）

メモリカードに書き込みたいデータの長さ

バイト(1…N) メモリカードに書き込みたいデータ

応答

応答	データ出力	
結果	SW1	SW2

その中：SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.6.6. 保護メモリカード書き込む（Write tomemory card）

このコマンドは、コマンドで指定されたバイトごとを特定のアドレスに格納されたバイトと比較し、データが一致すると、対応する保護ドビットが不可逆的に「0」に設定される。

コマンド

コマンド	CLA	INS	P1	バイトアドレス	MEM_L	バイト 1	…	…	バイト N
Write Protection Memory Card	FFh	D1h	00h						

その中：

バイトアドレス （1 バイト）

= 000A4 A3 A2 A1b (00h – 1Fh)は、メモリカードの保護メモリアドレス位置である

MEM_L （1 バイト）

メモリカードに書き込みたいデータの長さ

バイト(1…N)　カード内のバイトアドレスから始まるデータと比較するバイト値。バイト 1 とバイトアドレス中のデータを比べる；バイト N とバイトアドレス + N -1 のデータが比べる。

応答

応答	データ出力	
結果	SW1	SW2

その中：SW1 SW2　= 90 00h （操作が成功に完了であれば）

5.2.6.7. メモリカードパスワード提示（Present code memory card）（SLE4442 と SLE5542 のみ）

SLE4442 と SLE5542 に書き込む操作を有効にするために、メモリカードにパスワードを提示する時に、このコマンドを使用する。以下のアクションが実行される：

1. パスワード入力エラーカウンタにビット『1』を検索して、“0”に変更する。
2. 指定されたパスワードをカードに提示する。
3. プレゼンテーションエラーカウンタを消去する。

コマンド

コマンド	CLA	INS	P1	P2	MEM_L	パスワード		
						バイト 1	バイト 2	バイト 3
Present Code Memory Card	FFh	20h	00h	00h	03h			

その中：

パスワード　（3 バイト）

パスワード（PIN）

応答

応答	データ出力	
結果	SW1	ErrorCnt

その中：

ErrorCnt　（1 バイト）

エラーカウンタ

07h = 検証が正しい

00H = パスワードがロックされている（最大再試行回数を超過した）

他の値は認証が失敗したことを示す。

5.2.6.8. メモリカードパスワード変更（Change code memory card）（SLE4442 と SLE5542 のみ）

指定されたデータを新しいパスワードとして、カードに書き入れる時に、このコマンドを使用する。PRESENT_CODE コマンドでカードに現在のパスワードを提出してから、このコマンドを実行する。

コマンド

コマンド	CLA	INS	P1	P2	MEM_L	パスワード		
						バイト 1	バイト 2	バイト 3
Change Code Memory Card	FFh	D2h	00h	01h	03h			

その中：

パスワード （3 バイト）

パスワード（PIN）

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.7. メモリカード– SLE4406/SLE4436/SLE5536/SLE6636

5.2.7.1. カードタイプ選択（Select card type）

このコマンドはカードリーダーに挿入されて、選択されたカードにパワーダウン/アップを実行し、同時にリセットを実行する時に使われます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	カードタイプ
Select Card Type	FFh	A4h	00h	00h	01h	07h

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.7.2. メモリカード読み取る（Read Memory Card）

このコマンドは、指定したアドレス位置からメモリカードの内容を読み込む時に使われる。

コマンド

コマンド	CLA	INS	P1	バイトアドレス	MEM_L
Read Memory Card	FFh	B0h	00h		

その中：

バイトアドレス （1 バイト）

メモリカードのメモリアドレス位置

MEM_L （1 バイト）

メモリカードから読み出したいデータの長さ

応答

応答	バイト 1	…	…	バイト N	SW1	SW2
結果						

その中：

バイト(1…N) メモリカードから読み出されたデータ

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.7.3. 1 バイトを書き込む（Write one byte memory card）

このコマンドは、挿入されたカードの指定アドレスに 1 バイトを書き込むために使用される。バイトは LSB を初めにカードに書かれている。カードアドレス 0 のビットが 0 バイトの LSB とみなされる。

このようなカードには、コマンドデータフィールド内のフラグによって区別される 4 つの異なる書き込みモードがある。

a. Write

マンドで指定されたバイトの値が指定されたアドレスに書き込まれて、カードに個人化情報とカウンタ値を書き入れる。

b. Write with carry

コマンドで指定されたバイトの値が指定されたアドレスに書き込まれて、コマンドは次の下位カウンタステージを消去するためにカードに送信される。このモードはカードにカウンタ値を更新するだけ使用できる。

c. Write with backup enabled (SLE4436、SLE5536 および SLE6636 のみ)

マンドで指定されたバイトの値が指定されたアドレスに書き込まれて、カードに個人化情報とカウンタ値を書き入れます。バックアップビットを有効にして、カード裂けが発生すると、データの損失を防止することができる。

d. Write with carry and backup enabled (SLE4436、SLE5536 および SLE6636 のみ)

コマンドで指定されたバイトの値が指定されたアドレスに書き込まれて、コマンドは次の下位カウンタステージを消去するためにカードに送信される。このモードはカードにカウンタ値を更新するだけ使用できます 。バックアップビットを有効にして、カード裂けが発生すると、データの損失を防止することができます。

以下のモードで、指定されたアドレスのバイトは書き込みの操作を実行する前に消去されないので、メモリビットが‘1’を‘0’にプログラムしかできません。
SLE4436 カードと SLE5536 カード中で利用可能なのバックアップモードは、書き込み動作中に有効または無効にすることができる。

コマンド

コマンド	CLA	INS	P1	バイトアドレス	MEM_L	モード	バイト
Read Memory Card	FFh	D0h	00h		02h		

その中：

バイトアドレス （1 バイト）
メモリカードのメモリアドレス位置

モード （1 バイト）
書き込みモードとバックアップオプションを指定する
00h = Write
01h = Write with carry
02h = Write with backup enabled (SLE4436、SLE553 および SLE6636 のみ)
03h = Write with carry and with backup enabled (SLE4436、SLE5536 および SLE6636 のみ)

バイト （1 バイト）
カードに書き込みたいバイトの値

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.7.4. メモリカードパスワード提示（Present code memory card）

カードの個人化モードを有効にするために、メモリカードにシークレットコードを提出する時に、このコマンドを使用します。以下のアクションが実行される：

1. パスワード入力エラーカウンタにビット『1』を検索して、“0”に変更する。
2. 指定されたパスワードをカードに提示する。

コマンド

コマンド	CLA	INS	P1	P2	MEM_L	パスワード			
						Addr	バイト 1	バイト 2	バイト 3
Present Code Memory Card	FFh	20h	00h	00h	04h	09h			

その中：

Addr （1 バイト）

入力エラーカウンタのビットアドレス

パスワード （3 バイト）

パスワード（PIN）

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.7.5. メモリカード検証（Authenticate memory card）（SLE4436、SLE5536 及び SLE6636 のみ）

このコマンドがカードから認証証明書を読み取る時に使われる。以下のアクションが実行される：

1. コマンドで指定されたカード中の Key 1 または Key 2 を選択する。
2. コマンドに指定された乱数をカードに送信する。
3. カードによって計算した認証データの各ビットに対して指定された CLK パルスの数を生成する
4. カードから 16 ビットの認証データを読み出す。
5. カードを通常動作モードにリセットする

認証プロセスが 2 つステップで実行される：ステップ 1 はカードに認証証明書を送信する。ステップ 2 は、カードによって計算した 2 バイトの認証データを取り戻す。

ステップ 1 ：カードに認証証明書を送信する

コマンド

コマンド	CLA	INS	P1	P2	MEM_L	パスワード				
						キー	CLK_CNT	バイト 1	…	バイト 6
Send Authentication Certificate	FFh	84h	00h	00h	08h					

その中：

キー （1 バイト。）

認証証明書を計算するためのキー

00h = Key 1、暗号ブロック連鎖ない

01h = Key 2、暗号ブロック連鎖ない

80h = Key 1、暗号ブロック連鎖あり（SLL5536 と SLE6636 のみ）

81h = Key 2、暗号ブロック連鎖あり (SLE5536 と SLE6636 のみ)

CLK_CNT （1 バイト）

カードに供給されるクロックパルスの数。カードは、認証証明書の各ビットを計算するためにパルスを使用する。標準値は 160（A0h）です。

バイト (1...6) カードの乱数データ。

応答

応答	SW1	SW2
結果	61h	02h

ステップ 2：認証データ取得（Get Response）。

コマンド

コマンド	CLA	INS	P1	P2	MEM_L
Get Authentication Data	FFh	C0h	00h	00h	02h

応答

応答	Cert		SW1	SW2
結果				

その中：

Cert （2 バイト）

カードによって計算された 16 ビットの認証データ。バイト 1 の LSB はカードから読み出された一番目の認証ビット。

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.8. メモリカード – SLE4404

5.2.8.1. カードタイプ選択（Select card type）

このコマンドはカードリーダーに挿入されて、選択されたカードにパワーダウン/アップを実行し、同時にリセットを実行する時に使われます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	カードタイプ
Select Card Type	FFh	A4h	00h	00h	01h	08h

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.8.2. メモリカード読み取る（Read memory card）

このコマンドは、指定したアドレス位置からメモリカードの内容を読み込む時に使われる。

コマンド

コマンド	CLA	INS	P1	バイトアドレス	MEM_L
Read Memory Card	FFh	B0h	00h		

その中：

バイトアドレス （1 バイト）

メモリカードのメモリアドレス位置

MEM_L （1 バイト）

メモリカードから読み出したいデータの長さ

応答

応答	バイト 1	…	…	バイト N	SW1	SW2
結果						

その中：

バイト(1…N) メモリカードから読み出されたデータ

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.8.3. メモリカード書き込む（Write memory card）

このコマンド、指定されたアドレスにメモリカードを書き込む時に使われる。このバイトは LSB を初めにカードに書かれています。カードアドレス 0 のビットが 0 バイトの LSB とみなされます。

指定されたアドレスのバイトは書き込みの操作を実行する前に消去されないので、メモリビットが‘1’を‘0’にプログラムしかできません。

コマンド

コマンド	CLA	INS	P1	バイトアドレス	MEM_L	バイト 1	…	…	バイト N
Write Memory Card	FFh	D0h	00h						

その中：

バイトアドレス （1 バイト）

メモリカードのメモリアドレス位置

MEM_L （1 バイト）

メモリカードに書き込みたいデータの長さ

バイト(1…N) メモリカードに書き込みたいデータ

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.8.4. 一時メモリ消去（Erase scratch pad memory card）

挿入されたカードのスクラッチパッド中のデータを消去する時に、このコマンドを使う。スクラッチパッドメモリ内のすべてのメモリビットが“1”にプログラムされる。

コマンド

コマンド	CLA	INS	P1	バイトアドレス	MEM_L
Erase Scratch Pad	FFh	D2h	00h		00h

その中：

バイトアドレス　　（1 バイト）

一時メモリ装置のメモリバイトアドレス位置。標準値は 02h です

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2　　= 90 00h （操作が成功に完了であれば）

5.2.8.5. ユーザーパスワード検証（Verify user code）

挿入されているカードにユーザーシークレットコードを提出する時、このコマンドが使用される。｜（2 バイト）このパスワードにより、ユーザーはカードのメモリにアクセスできる。

以下のアクションが実行される：

1. 指定されたパスワードをカードに提出する。
2. パスワード入力エラーカウンタにビット『1』を検索して、“0”に変更する。
3. プレゼンテーションエラーカウンタを消去する。提出したシークレットコードが成功に認証されて、ユーザーエラーカウンターが消去ことができる。

コマンド

コマンド	CLA	INS	エラーカウンタ LEN	バイトアドレス	MEM_L	パスワード	
						バイト 1	バイト 2
Verify User Code	FFh	20h	04h	08h	02h		

その中：

エラーカウンタ LEN （1 バイト）

入力エラーカウンタの長さ、単位はビットです。

バイトアドレス （1 バイト）

カード中のキーのバイトアドレス

パスワード （1 バイト）

ユーザーパスワード

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

= 63 00h（残りの再試行回数が 0 の場合）

注釈：*SW1 SW2 = 90 00h* を受信してから、*Verify_User_Code* が正しいかどうかをチェックするために、ユーザーのエラーカウンターを再度読み取るはずです。ユーザーのエラーカウンターが全部消去されて、*'FFh'*に等しいされている場合、前回の検証が成功したのを示す。

5.2.8.6. メモリパスワード検証（Verify memory code）

挿入されているカードにメモリコードを提出する時、このコマンドが使用される。｜（4 バイト）メモリコードはユーザコードとユーザメモリの再ロードを許可するために使用される。

以下のアクションが実行される：

1. 指定されたパスワードをカードに提出する。
2. プレゼンテーションエラーカウンタにビット'1'を検索して、'0'に変更する。
3. プレゼンテーションエラーカウンタを消去する。

   注釈：エラーカウンターメモリ中のデータが消去されてはいけない。

コマンド

コマンド	CLA	INS	エラーカウンタ LEN	バイトアドレス	MEM_L	パスワード			
						バイト 1	バイト 2	バイト 3	バイト 4
Verify Memory Code	FFh	20h	40h	28h	04h				

その中：

エラーカウンタ LEN （1 バイト）

入力エラーカウンタの長さ、単位はビットです。

バイトアドレス （1 バイト）

カード中のキーのバイトアドレス

パスワード （4 バイト）

ストレージパスワード

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

= 63 00h（残りの再試行回数が 0 の場合）

注釈：*SW1 SW2 = 90 00h* を受信してから、*Verify Memory Code* が正しいかどうかをチェックするために、ユーザーのエラーカウンターを再度読み取るはずです。アプリケーションゾーンのデータが全部消去されて、*'FFh'*に等しいされている場合、前回の検証が成功したことを示す。

メモリカード – AT88SC101/AT88SC102/AT88SC1003

5.2.8.7. カードタイプ選択（Select card type）

このコマンドは選択されて、カードリーダーに挿入されたカードにパワーダウン/アップを実行する。同時にリセットを実行する時に使われる。

コマンド

疑似 APDU						
コマンド	CLA	INS	P1	P2	Lc	カードタイプ
Select Card Type	FFh	A4h	00h	00h	01h	09h

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

5.2.8.8. メモリカード読み取る（Read Memory Card）

このコマンドは、指定したアドレス位置からメモリカードの内容を読み込む時に使われる。

コマンド

疑似 APDU					
コマンド	CLA	INS	P1	バイトアドレス	MEM_L
Read Memory Card	FFh	B0h	00h		

その中：

バイトアドレス （1 バイト）

メモリカードのメモリアドレス位置。

MEM_L （1 バイト）

メモリカードから読み出したいデータの長さ

応答

応答	バイト 1	…	…	バイト N	SW1	SW2
結果						

その中：

バイト(1…N)	メモリカードから読み出されたデータ
SW1 SW2	= 90 00h （操作が成功に完了であれば）

5.2.8.9. メモリカード書き込む（Write Memory Card）

このコマンドは、挿入されたカードの指定されたアドレスにデータを書き込むに使用される。バイトは LSB を初めにカードに書かれている。カードアドレス 0 のビットが 0 バイトの LSB とみなされる。

指定されたアドレスのバイトは書き込みの操作を実行する前に消去されないので、メモリビットが'1'を'0'にプログラムしかできません。

コマンド

疑似 APDU									
コマンド	CLA	INS	P1	バイトアドレス	MEM_L	バイト 1	…	…	バイト N
Write Memory Card	FFh	D0h	00h						

その中：

バイトアドレス （1 バイト）

メモリカードのメモリアドレス位置。

MEM_L （1 バイト）

メモリカードに書き込みたいデータの長さ

バイト(1…N) 書き込み待ちカードのバイト。

応答

応答	データ出力	
結果	SW1	SW2

その中：**SW1 SW2** = 90 00h （操作が成功に完了であれば）

5.2.8.10. 非アプリケーション領域消去（Erase non-application zone）

このコマンドが非アプリケーション領域ににストアーされているデータを消去する時に使われる。EEPROM メモリが 16 ビットのワードで構造される。独立な 1 ビットのワードを消去しても、ERASE 操作が全てのワードを消去できる。したがって、メモリ中の任意のビットに消去を実行すると、そのメモリのすべての 16 ビットをクリアして、"1"の状態になる。

エラーカウンタまたはアプリケーションゾーンのデータを消去する時、以下のコマンドを参照してください：

- 指定の Erase Application Zone With Erase コマンド
- 指定の Erase Application Zone With Write and Erase コマンド
- 指定の Verify Security Code コマンド

コマンド

疑似 APDU					
コマンド	CLA	INS	P1	バイトアドレス	MEM_L
Erase Non-Application Zone	FFh	D2h	00h		00h

その中：

バイトアドレス　（1 バイト）

消去したいワードのメモリバイトアドレスの位置

応答

応答	データ出力	
結果	SW1	SW2

その中：**SW1 SW2** = 90 00h （操作が成功に完了であれば）

5.2.8.11. アプリケーションゾーン消去（Erase Application Zone with Erase）

このコマンドは以下の状況に適用される：

- AT88SC101：アプリケーションゾーンのデータを消去して、EC 機能が無効になる。
- AT88SC102：アプリケーションゾーン 1 のデータを消去する。
- AT88SC102：アプリケーションゾーン 2 のデータを消去して、EC2 機能が無効になる。
- AT88SC1003：アプリケーションゾーン 1 のデータを消去する。
- AT88SC1003：アプリケーションゾーン 2 のデータを消去して、EC2 機能が無効になる。
- AT88SC1003：アプリケーションゾーン 3 のデータを消去する。

このコマンドで以下の操作を実行する：

1. 指定されたシークレットコードをカードに提出する。
2. プレゼンテーションエラーカウンタを消去する。提出したコードが成功に認証されて、アプリケーションゾーンのデータが消去することができる。

コマンド

疑似 APDU										
コマンド	CLA	INS	エラーカウンタ LEN	バイトアドレス	MEM_L	パスワード				
						バイト 1	バイト 2	…	…	バイト N
Erase Application Zone with Erase	FFh	20h	00h							

その中：

エラーカウンタ LEN （1 バイト）
= 入力エラーカウンタの長さ、単位はビットです。ずっと 00h であるべき。

バイトアドレス （1 バイト）
= カード中でアプリケーションゾーンのキーのバイトアドレス。精確な数値が下のチャートを参照してください。

MEM_L （1 バイト）
= 『消去』キーの長さ。精確な数値が下のチャートを参照してください。

パスワード**(1…N)** = “消去”キー

状況	バイトアドレス	LEN
AT88SC101：アプリケーションゾーンを消去して、EC 機能が無効になる。	96h	04h
AT88SC102：アプリケーションゾーン 1 を消去する	56h	06h
AT88SC102：アプリケーションゾーン 2 を消去して、EC2 機能が無効になる。	9Ch	04h
AT88SC1003：アプリケーションゾーン 1 を消去する	36h	06h

状況	バイトアドレス	LEN
AT88SC1003：アプリケーションゾーン 2 を消去して、EC2 機能が無効になる。	5Ch	04h
AT88SC1003：アプリケーションゾーン 3 を消去する	C0h	06h

応答

応答	データ出力	
結果	SW1	SW2

その中：**SW1 SW2** = 90 00h （操作が成功に完了であれば）

注： *SW1 SW2 = 90 00h* を受信してから、コマンド *Erase Application Zone with Erase* が正しいかどうかをチェックするために、アプリケーションゾーンのデータを再度読み取るはずです。アプリケーションゾーンのデータが全部消去されて、*"FFh"* に等しいされている場合、前回の検証が成功したことを示す。

5.2.8.12. アプリケーションゾーン消去（書き込みと消去）（Erase Application Zone with Write andErase）

このコマンドは以下の状況に適用される：

- AT88SC101：アプリケーションゾーンのデータを消去して、EC 機能が有効になる。
- AT88SC102：アプリケーションゾーン 2 のデータを消去して、EC2 機能が有効になる。
- AT88SC1003：アプリケーションゾーン 2 のデータを消去して、EC2 機能が有効になる。

EC または EC2 機能が有効になってから（すなわち：ECEN または EC2EN ヒューズが変更されていなくて、“1”の状態）、以下の操作を実行する：

1. 指定されたコードをカードに提出する。
2. プレゼンテーションエラーカウンタにビット‘1’を検索して、‘0’に変更する。
3. プレゼンテーションエラーカウンタを消去する。提出したコードが成功に認証されて、アプリケーションゾーンのデータが消去することができる。

コマンド

疑似 APDU									
コマンド	CLA	INS	エラーカウンタ LEN	バイトアドレス	MEM_L	パスワード			
						バイト 1	バイト 2	バイト 3	バイト 4
Erase Application Zone with Write and Erase	FFh	20h	80h		04h				

その中：

エラーカウンタ LEN （1 バイト）

= 入力エラーカウンタの長さ、単位はビットです。ずっと 80h であるべき。

バイトアドレス （1 バイト）

= カード中でアプリケーションゾーンのキーのバイトアドレス。精確な数値が下のチャートを参照してください。

パスワード （4 バイト）

= 『消去』キー

状況	バイトアドレス
AT88SC101	96h
AT88SC102	9Ch
AT88SC1003	5Ch

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

= 63 00（残りの再試行回数が 0 の場合）

注： *SW1 SW2 = 90 00* を受信してから、コマンド *Erase Application Zone with Write and Erase* が正しいかどうかをチェックするために、アプリケーションゾーンのデータを再度読み取るはずです。アプリケーションゾーンのデータが全部消去されて、*"FFh"* に等しいされている場合、前回の検証が成功した。

5.2.8.13. 安全パスワード検証（Verify Security Code）

挿入されているカードにセキュリティコードを提出する時、このコマンドが使用される。｜（2 バイト）カードのメモリがアクセスできるように、セキュリティコードはそれが意図されている。

以下のアクションが実行される：

1. 指定されたコードをカードに提出する。
2. プレゼンテーションエラーカウンタにビット‘1’を検索して、‘0’に変更する。
3. プレゼンテーションエラーカウンタを消去する。提出したコードが成功に認証されて、セキュリティコードの試みるカウンターが消去ことができる。

コマンド

疑似 APDU							
コマンド	CLA	INS	エラーカウンタ LEN	バイトアドレス	MEM_L	パスワード	
						バイト 1	バイト 2
Verify Security Code	FFh	20h	08h	0Ah	02h		

その中：

エラーカウンタ LEN （1 バイト）

= 入力エラーカウンタの長さ、単位はビットです。

バイトアドレス （1 バイト）

=カード中のキーのバイトアドレス。

パスワード （2 バイト）

= セキュリティコード

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

= 63 00（残りの再試行回数が 0 の場合）

注釈：*SW1SW2 = 90 00h* を受信してから、*Verify User Code* が正しいかどうかをチェックするために、セキュリティコードの試行カウンター（*SCAC*）を再度読み取るはずです。*SCAC* が消去されて、*'FFh'*に等しいされている場合、前回の検証が成功した。

5.2.8.14. 識別子ビット変更（Blow Fuse）

このコマンドは挿入されているカードの識別子ビットを変更する時に使われる。ヒューズは EC_EN の識別子ビット、EC2EN の識別子ビット、メーカーの識別子ビットまたは発行者の識別子ビット可能です。

注釈：識別子ビットを変更することは不可逆過程です。

コマンド

疑似 APDU									
コマンド	CLA	INS	エラーカウンタ LEN	バイトアドレス	MEM_L	パスワード			
						識別子ビットアドレス（高）	識別子ビットアドレス（低）	FUS Pin 状態	RST Pin 状態
Blown Fuse	FFh	05h	00h	00h	04h			01h	00h 01h

その中：

識別子ビットアドレス （2 バイト）

=識別子ビットのビットマスク精確な数値が下のチャートを参照してください。

FUS Pin 状態 （1 バイト）

= FUS pin の状態、ずっと 01h であるべき。

RST Pin 状態 （1 バイト）

= RST pin の状態、精確な数値が下のチャートを参照してください。

		識別子ビットアドレス（高）	識別子ビットアドレス（低）	RST Pin 状態
AT88SC101	メーカーの識別子ビット	05h	80h	01h
	EC2EN の識別子ビット	05h	C9h	01h
	発行者の識別子ビット	05h	E0h	01h
AT88SC102	メーカーの識別子ビット	05h	B0h	01h
	EC2EN の識別子ビット	05h	F9h	01h
	発行者の識別子ビット	06h	10h	01h
AT88SC1003	メーカーの識別子ビット	03h	F8h	00h
	EC2EN の識別子ビット	03h	FCh	00h
	発行者の識別子ビット	03h	E0h	00h

表 3：識別子ビット値変更

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了であれば）

= 63 00（残りの再試行回数が 0 の場合）

5.2.9. ACOS6-SAM カードコマンド

このセクションでは、SAM 専用のコマンドを紹介する。CCID ホストは、PCSC API 内の SCardTransmit（）を使用してカード固有コマンド（Card Native Command）または APDU をリーダライタに送信することができる。

注釈：*ACOS6-SAM* カードは *ACR1581U* リーダライタのセット *SDK* にのみ付属しており、標準の小売版リーダライタには *ACOS6-SAM* カードは含まれていません。*ACOS6-SAM*** コマンドのすべての情報と応用例をご了解したい場合、*ACS* 販売代表に連絡し、*ACOS6-SAM* リファレンスマニュアルを入手してください。

5.2.9.1. キー生成（Generate Key）

このコマンドは、顧客カードのシリアル番号などの偏差データを用いて分散鍵を生成し、ACOS 3/6 または他のカードに導入することで、顧客カード発行の目的を満たす。

APDU	説明	
CLA	80h	
INS	88h	
P1	00h 01h 02h	8 バイトのキーを生成する 16 バイトのキーを生成する 24 バイトのキーを生成する
P2	導出鍵を生成するためのマスター鍵の鍵インデックス	
P3	08h	
データ	データ入力：	

特定の応答ステータスバイト

SW1 SW2	説明
69 86h	DF 未選択
6A 86h	P1 もしくは P2 無効

SW1 SW2	説明
67 00h	P3 が正しくない、08h でなければならない
6A 83h	指定された鍵レコードが EF 2 には見つかりません
69 81h	無効な EF2（レコードサイズ、ファイルタイプなど）
6A 88h	EF2 が見つかりません
62 83h	現在の DF はロックされている; EF2 がロックされている
69 83h	使用カウンタはゼロです。
69 82h	セキュリティ条件が満たされていない
6A 87h	指定されたマスターキーは 3-DES 暗号化サポートしない
61 08h	コマンドが完了、結果を得るために GET REPONSE を送信する

5.2.9.2. キーデータ分散化（またはロード）（Diversify (or Load) Key Data））

このコマンドは、鍵分散と鍵ロードによって SAM カードに暗号化操作を実行する準備をさせます。コマンドデータ入力として、シリアル番号と CBC 初期ベクトルを取ります。

APDU	説明								
CLA	80h								
INS	72h								
P1	b7	b6	b5	b4	b3	b2	b1	b0	説明
	-	0	0	0	0	0	0	1	パスワード(Sc)
	-	0	0	0	0	0	1	0	アカウントキー(K_{ACCT})
	-	0	0	0	0	0	1	1	エンドキー
	-	0	0	0	0	1	0	0	カードキー
	-	0	0	0	0	1	0	1	キーを一括に暗号化する（分散しない）
	-	0	0	0	0	1	1	0	初期ベクトル
	0	-	-	-	-	-	-	-	16 バイトのキー
	1	-	-	-	-	-	-	-	24 バイトのキー
P2	マスターキーの索引： Bit7: 1 =現在 EF2 中のローカルキー； 0 =グローバルキー EF2 Bit6-Bit5: 00b - RFU Bit4-Bit0: キーインデックス								
P3	P1 = 1～4 の場合、P3 = 8/16（アルゴリズムが AES の場合、P3 = 8/16） P1 = 5 の場合、P3 = 0 P1 = 6 の場合、 P3 = 8（マスターキーの Algo は DES / 3DES / 3KDES） P3 = 16（マスターキーの Algo は AES）								
データ	P1 = 1-4 の場合、クライアントカードのシリアル番号（algo が AES の場合、データはクライアントカードのシリアル番号またはクライアントカードのシリアル番号に「0000000000000000」が付加されます） P1 = 5 の場合、コマンドデータはありません。 P1 = 6 の場合、DES / 3DES / 3KDES / AES CBC の初期ベクトル。								

特定の応答ステータスバイト

SW1 SW2	説明
69 86h	DF 未選択
6A 86h	間違った P1、P1 は 1～6 でなければなりません
67 00h	間違った P3、P3 は 8（または 0）
62 83h	現在の DF がロックされているか、または EF2 がロックされている
69 82h	セキュリティ条件が満たされていない
6A 88h	EF2 が見つかりません
6A 83h	EF2 に指定マスターキーが見つかりません
69 81h	無効な EF2（FDB、MRL など、一貫性がない）
6A 87h	指定された KEY 認証できない
69 83h	参照されたキーがロックされています
90 00h	ターゲットキーが生成され、SAM メモリに準備されている

5.2.9.3. 暗号化（Encrypt）

このコマンドは、DES または 3DES を使用して次のいずれかの方法でデータを暗号化します。

1. ACOS3/6、DESFire®、DESFire® EV1/EV2/Light または MIFARE Plus カードとの相互認証手順によって作成されたセッションキー。
2. 分散化した鍵（パスワード）。
3. キーを一括に暗号化する。
4. セッション鍵で分散化したパスワードを暗号化します。
5. 非 SM コマンドを指定して、ACOS3 セキュア・メッセージング・コマンドを作成します。

APDU	説明
CLA	80h
INS	74h

APDU	説明								
	b7	b6	b5	b4	b3	b2	b1	b0	説明
	-	0	0	0	0	0	0	-	ECB モード
	-	0	0	0	0	0	1	-	CBC モード
	-	0	0	0	0	1	0	-	小売 MAC モード
	-	0	0	0	0	1	1	-	MAC モード
	-	0	0	0	1	0	0	-	ACOS3 SM コマンド用意する
	-	1	0	0	1	0	1	-	MIFARE DESFire 暗号化
P1	-	1	0	0	1	1	0	-	MIFARE DESFire EV1/EV2/Light 暗号化
	-	0	0	0	1	1	1	-	CMAC
	-	0	1	0	0	0	0		MIFARE Plus コマンド
	-	0	1	0	0	0	1		MIFARE Plus 応答
	0	-	-	-	-	-	-	0	3DES
	0	-	-	-	-	-	-	1	DES
	1	-	-	-	-	-	-	0	3K DES
	1	-	-	-	-	-	-	1	AES
	-	-	-	-	-	-	-	-	他のすべての値 - RFU
P2	P 2 は、Load Key 機能を使用して SAM セットに分散された鍵を表す。 1 - セッションキーKs でデータを暗号化する 2 - 多様なキーSc でデータを暗号化する 3 - 一括暗号鍵でデータを暗号化する 0 - ENC（Sc、Ks）を返す P1.b3 = 1 または b5 = 1 の場合、P2 は 1 でなければなりません P2 = 0h の場合、P1 は 0 または 1 のいずれかになります								
P3	P3 < 128 P1 のビット 3 が 1 に等しくなく、P1 のビット 5 が 1 に等しくない場合 - P2 = 1-3 の場合、8（DES / 3DES / 3KDES）の倍数または 16 バイト（AES）の 128 バイト - P2 = 0 の場合、0								

APDU	説明
データ	プレーンテキスト P2 b6 = 1 の場合、データのフォーマットが次のようになります： • プレインテキストデータの長さ • DESFire カードのコマンドとヘッダーの長さ • DESFire カードのコマンドとヘッダー • プレーンテキスト P1 = A1h、の場合、暗号化は Plus コマンド用です • MFP コマンドが値操作コマンドの場合、データのフォーマットは Command Code（1 バイト）＋BlockNum（2/4 バイト）＋Value（4 バイト）でなければなりません。 • MFP Command が Proximity Check の場合、データフォーマットは Command Code（1 バイト）+ PPS1（1 バイト）でなければなりません。 • MFP コマンドが読み取りの場合、データフォーマットは Command code(1 バイト）+BlockNum（2 バイト） • MFP コマンドが書き込みである場合、データフォーマットは Command code（1 バイト）+ BlockNum（2 バイト）+plaintext P1 = A3h、 • ICC によって返されたデータ（SC コードは含まず、RMAC が存在する場合は RMAC を含まない） 以前 Authenticate EV 2 First/Authenticate EV 2 NonFirst コマンドで認証された場合、P1 は CDh または 8 Fh でなければなりません。 • P1=CDh の場合、暗号化操作は DESFire EV 2/Light コマンドに使用される。データのフォーマットは： 平文データ • P1=8Fh の場合、CMAC は DESFire EV 2/Light コマンドに使用される。データのフォーマットは： ➢ EV2 コマンド：完全コマンド（CMAC データを含まない） ➢ EV 2 応答：EV 2 からの完全応答 ➢ Light コマンド：完全コマンド（CMAC データと CLA/P 1/P 2/Lc/Le バイトを含まない）。 ➢ Light 応答：Light からの SW 2 と応答データ。 EV 2/Light カードがステータスコードと平文データのみを返す場合は、データは 00 h であり、SAM カードに送信される必要がある。

特定の応答ステータスバイト

SW1 SW2	説明
69 86h	DF 未選択
6A 86h	P1 もしくは P2 無効
67 00h	P3 正しくない
6A 83h	ACOS ターゲットキーが準備されていません（キーを生成するために Diversify を使用してください）
61 XX	暗号化が行われ、結果を得るために GET RESPONSE を使用する

5.2.9.4. 復号化（Decrypt）

このコマンドは、DES または 3DES または AES を使用して次のいずれかを使用してデータを復号化するために使用されます。

1. ACOS3/6、MIFARE DESFire、MIFARE DESFire EV1//またはMIFARE Plusカードとの相互認証手順によって作成されたセッションキー。
2. 分散化した鍵（パスワード）。
3. キーを一括に暗号化する。
4. セッション鍵で多様化したパスワードを解読化します。
5. ACOS3の安全なメッセージングの応答を確認し、解読する。

ACOS3 の安全なメッセージングの応答を確認し、解読する。

APDU	説明
CLA	80h
INS	76h

APDU	説明								
	b7	b6	b5	b4	b3	b2	b1	b0	説明
	-	0	0	0	0	0	0	-	ECB モード
	-	0	0	0	0	0	1	-	CBC モード
	-	0	0	0	1	0	0	-	ACOS3 の安全なメッセージングの応答を確認し、解読する。
	-	1	0	0	1	0	1	-	MIFARE DESFire で復号化
P1	-	1	0	0	1	1	0	-	MIFARE DESFire EV1 で復号化
	-	0	1	0	0	1	0	-	MIFARE Plus 復号化
	0	-	-	-	-	-	-	0	3DES
	0	-	-	-	-	-	-	1	DES
	1	-	-	-	-	-	-	0	3K DES
	1	-	-	-	-	-	-	1	AES
	0	0	0	0	-	-	-	-	他のすべての値 - RFU
P2	P 2 は、Load Key 機能を使用して SAM セットに分散された鍵を表す。 1 - セッションキーKs でデータを復号化する 2 - 多様なキーSc でデータを復号化する 3 - 一括暗号鍵でデータを復号化する 0 - DEC（Sc、Ks）を返す								
P3	P3 < 128 P1 = A5h の場合、P3 = 16/32/48 P1 のビット 3 が 1 に等しくない場合 - P2 = 1-3 の場合、8（DES / 3DES / 3KDES）の倍数または 16 バイト（AES）の 128 バイト - P2 = 0 の場合、0								

APDU	説明
データ	暗号文 P1 = A5h の場合、データが暗号化されたテキスト P2 b6 = 1 の場合、データのフォーマットが次のようになります： • Plain テキストデータの長さ（不明の場合は 00 を使用） • DESFire カードのコマンドとデータヘッダーの長さ • DESFire カードのコマンドとカードのデータヘッダー • 暗号化されたテキスト 以前 Authenticate EV2 First/Authenticate EV2 NonFirst コマンドで認証された場合、かつ EV2/Light カードは暗号化された及び付加 CMAC データを返す場合、P1 は CDh Fh でなければなりません。データのフォーマットは： • EV 2 応答：EV 2 からの完全応答 • Light 応答：Light からの SW 2 と応答データ。

特定の応答ステータスバイト

SW1 SW2	説明
69 86h	DF 未選択
6A 86h	P1 もしくは P2 無効
67 00h	P3 正しくない
6A 83h	ACOS ターゲットキーが準備されていません（キーを生成するために Diversify を使用してください）
61 XX	解読化が行われ、結果を得るために GET RESPONSE を使用する

5.2.9.5. 認証準備（Prepare Authentication)

このコマンドは、ACOS3/6 カード、または MIFARE Ultralight C/MIFARE DESFire カード/MIFARE DESFire EV1/EV2/Light カード/MIFARE Plus カードに対する SAM カード（端末として）の正当性を検証するために使用される。

APDU	説明
CLA	80h
INS	78h
P1	00h – 3DES 01h – DES

APDU	説明
	02h – 3KDES (MIFARE DESFire EV1/ACOS3) 03h – AES (MIFARE DESFire EV1/MIFARE Plus/ACOS3) 80h – 3DES (MIFARE DESFire 認証のみ) 81h – DES (MIFARE DESFire 認証のみ) 他 – RFU
P2	0h - ACOS3 / 6 が認証を返すことを確認する 01h - MIFARE 超軽量 C / DESFire 認証（多様化）ターミナルキー 05h - MIFARE Ultralight C / DESFire は一括暗号鍵で認証します 02h - MIFARE Plus 認証 SL1 から SL3 の最初の認証 03h - MIFARE Plus 認証。SL1 から SL2 への認証。 04h - MIFARE Plus 認証。SL2 から SL3 への認証に続きます。 06h – MIFARE DESfire EV2/Light AuthenticateEV2First/AuthenticateEV2NonFirst/Mifare Plus EV1 セキュアメッセージ送信
P3	8 – (P1 = 00h, 01h, 02h, 80h, 81h) 16 – (P1 = 03h)
データ	カードチャレンジデータ

特定の応答ステータスバイト

SW1 SW2	説明
69 86h	DF 未選択
6A 86h	P1 もしくは P2 無効
67 00h	P3 が正しくない、08h でなければならない
6A 83h	ACOS キー（KT または KC）が準備できていない（Diversify でこのキーを生成してください）
69 82h	セキュリティ条件が満たされていない
61 10h	コマンドが完了、結果を得るために GET REPONSE を送信する

5.2.9.6. 認証検証（Verify Authentication）

このコマンドは、ACOS 3/6、MIFARE Ultralight C、MIFARE DESFire/MIFARE DESFire EV1 または MIFARE Plus カードが端末に対する正当性を検証するために使用され、内部でプロセスキーKs も生成されます。

APDU	説明
CLA	80h
INS	7Ah
P1	00h – 3DES (P2 = 0) 01h – DES (P2 = 0) 02h – 3KDES (P2 = 0，ACOS3) 03h – AES (P2 = 0，ACOS3) 他 – RFU
P2	00h - ACOS3 /6 認証で返されたメッセージを検証する 01h - MIFARE Ultralight C®/ DESFire®/ DESFire® EV1 認証状況を確認する 02h – MIFARE Plus 認証リターン情報、または MIFARE DESFire EV2/Light の AuthenticateEV2 First/AuthenticateEV2 NonFirst リターン情報を調べる
P3	08h - （P2 = 0、P2 = 1、セッションキーは DES / 3DES） 16h - （P2 = 1、、セッションキーは 3KDES / AES 使用） 16h - （P2 = 02、MIFARE Plus がリターンデータ ek（RndA '）） 32h - （P2 = 02、MIFARE Plus がリターンデータ ek（TI + PICCcap2 + PCDcap2））
データ	ACOS 3/6：DES (K_S, RND_T) MIFARE DESFire/ DESFire EV1/EV2/Light リターンデータ：ek(RndA') MIFARE Plus がリターンデータ ek（RndA '）または ek（TI + PICCcap2 + PCDcap2）

特定の応答ステータスバイト

SW1 SW2	説明
69 86h	DF 未選択
6A 86h	P1 もしくは P2 無効
67 00h	P3 が正しくない、08h でなければならない
6A 83h	ACOS-SAM セッションキーまたは RNDT は準備ができていません。PREPARE AUTHENTICATION を使用してこれらのキーを作成します。
69 82h	データ正しくない
90 00h	データ正しくて ACOS 相互認証成功

5.2.9.7. ACOS クエリアカウントチェック（Verify ACOS Inquire Account）

このコマンドは、ACOS3 / 6 カードの Inquire Account purse コマンドを確認するために使用します。それは、ACOS3 / 6 によって返された MAC チェックサムが、SAM の多様化された鍵で正しいことを検証する。

APDU	説明								
CLA	80h								
INS	7Ch								
P1	b7	b6	b5	b4	b3	b2	b1	b0	説明
	-	0	0	0	0	-	0	-	ACOS INQ_AUT 無効
	-	0	0	0	0	-	1	-	ACOS INQ_AUT 有効
	-	0	0	0	0	0	-	-	ACOS INQ_ACC_MAC 無効
	-	0	0	0	0	1	-	-	ACOS INQ_ACC_MAC 無効
	0	-	-	-	-	-	-	0	3DES
	0	-	-	-	-	-	-	1	DES
	1	-	-	-	-	-	-	0	3K DES（ACOS3 のみ）
	1	-	-	-	-	-	-	1	AES（ACOS3 のみ）
P2	0h								
P3	1Dh								
データ	クライアントの ACOS カードの INQUIRE ACCOUNT によって返されるデータブロック。								

特定の応答ステータスバイト

SW1 SW2	説明
69 86h	DF 未選択
6A 86h	P1 もしくは P2 無効
67 00h	P3 正しくない
6A 83h	ACOS キー K_S または K_{ACCT} は準備ができていません。K_{ACCT} 生成するには DIVERSIFY コマンドを使用します。該当する場合は、「Prepare Authentication」を介して K_S を生成します。
6F 00h	**データブロックの MAC が正しくありません**
90 00h	**データブロックの MAC が正しい**

5.2.9.8. ACOS アカウント取引準備（Prepare ACOS Account Transaction）

ACOS3 / 6 クレジット/デビットコマンドを作成するには、ACOS3 / 6 が検証するために MAC を計算する必要があります。

APDU	説明								
CLA	80h								
INS	7Eh								
P1	b7	b6	b5	b4	b3	b2	b1	b0	説明
	-	0	0	0	0	0	0	-	ACOS TRNS_AUT 無効
	-	0	0	0	0	0	1	-	ACOS TRNS_AUT 有効
	0	-	-	-	-	-	-	0	3DES
	0	-	-	-	-	-	-	1	DES
	1	-	-	-	-	-	-	0	3K DES（ACOS3 のみ）
	1	-	-	-	-	-	-	1	AES（ACOS3 のみ）
P2	E2h: クレジット E6h: デビット								
P3	0Dh								
データ	データブロック								

特定の応答ステータスバイト

SW1 SW2	説明
69 86h	DF 未選択
6A 86h	P1 もしくは P2 無効
67 00h	P3 が正しくない、必ず 0Dh
6A 83h	ACOS キーK_S または K_{ACCT} は準備ができていません。K_{ACCT} 生成するには DIVERSIFY コマンドを使用します。該当する場合は、「Prepare Authentication」を介して K_S を生成します。
61 0Bh	コマンドが完了、結果を得るために GET REPONSE を送信する

5.2.9.9. デビット証明書認証（Verify Debit Certificate）

ACOS3 / 6 の場合、DEBIT コマンドに P1 = 1 が指定されている場合は、借方の証明書が返されます。デビット証明書は、ACOS3 の応答をこのコマンドの結果と比較することによって確認できます。

APDU	説明								
CLA	80h								
INS	70h								
P1	b7	b6	b5	b4	b3	b2	b1	b0	説明
	-	0	0	0	0	0	0	-	ACOS TRNS_AUT 無効
	-	0	0	0	0	0	1	-	ACOS TRNS_AUT 有効
	0	-	-	-	-	-	-	0	3DES
	0	-	-	-	-	-	-	1	DES
	1	-	-	-	-	-	-	0	3K DES（ACOS3 のみ）
	1	-	-	-	-	-	-	1	AES（ACOS3 のみ）
P2	0h								
P3	14h								
データ	データブロック								

特定の応答ステータスバイト

SW1 SW2	説明
69 86h	DF 未選択
6A 86h	P1 もしくは P2 無効
67 00h	P3 が正しくない、14h でなければならない
6A 83h	ACOS キー K_S または K_{ACCT} は準備ができていません。K_{ACCT} 生成するには DIVERSIFY コマンドを使用します。該当する場合は、「Prepare Authentication」を介して K_S を生成します。
69 82h	セキュリティ条件が満たされていない
6F 00h	DEBIT CERTIFICATE 無効
90 00h	成功，DEBIT CERTIFICATE 有効

5.2.9.10. キー取得（Get Key）

鍵を取るコマンドにより、鍵を現在の SAM の鍵ファイル（SFI=02 h）から別の ACOS 6/ACOS 6-SAM カードに安全に注入することができ、このプロセスは、鍵分散を介さなくても実現できます。これを使用すると、注入される鍵が暗号化およびメッセージ認証コードによって保護されます。

またこのコマンドが分散化により、現在の SAM のキーファイル（SFI = 02h）から ACOS7/10、MIFARE DESFire、MIFARE DESFire EV1/EV2/Light または MIFARE Plus カードに安全に注入することができる。これを使用すると、注入される鍵が暗号化およびメッセージ認証コードによって保護されます。

カードヘッダーブロック（ACOS6-SAM リファレンスマニュアルのセクション 4.2）の特殊機能フラグ（キー注入専用フラグ）のビット 7 が設定され、キーファイルが有効になっている場合、キーのロードまたは変更には Get キーを使用する必要があります。Bit 7 が設定されると、鍵ファイルがアクティブになると、Read Record コマンドの使用はいずれの場合も無効になります。

このコマンドを実行する前に、**相互認証**の相互認証プロセス（ACOS6-SAM リファレンスマニュアルのセクション 6.3）または MIFARE Plus/MIFARE DESFire 相互認証プロセスで、ターゲットカードとのセッション鍵がすでに確立されています。

注：*GET KEY* コマンドはキーデータのみを取得できます。

APDU	説明			
CLA	80h			
INS	CAh			
P1	ACOS カードがキーを取得、リセットする用			
	00h	応答データは MSAM のキーです		
	01h	応答データは 16 バイトの Diversify キーです		
	02h	応答データは 24 バイトの Diversify キーです		
	03h	応答データは MIFARE Plus カードの Change Key コマンドです		
	DESFire カードのキー取得変更キー、DESFire / DESFire EV1 Change キーの応答データ			
		カードタイプ	キー番号の認証とキー番号の変更*	キー長さ
	80h	MIFARE DESFire	MIFARE®DESFire の中では異なる	16 バイト
	81h	MIFARE DESFire EV1	MIFARE DESFire EV1 カードの中では異なる	16 バイト
	82h	MIFARE DESFire EV1	MIFARE DESFire EV1 カードの中では異なる	24 バイト
	88h	MIFARE DESFire	MIFARE DESFire の中では同じ	16 バイト
	89h	MIFARE DESFire EV1	MIFARE DESFire EV1 カードの中では同じ	16 バイト

APDU	説明
	8Ah　MIFARE DESFire EV1　MIFARE DESFire EV1 カードの中では同じ　24 バイト
P2	SAM のキーID（変更のための新しいキー）
P3	P1 = 00h，P3 = 08h P1 = 02h，P3 = 10h P1 = 03h，P3 = 0h P1 = 80/81/82/88/89/8Ah：P3 = 0Bh もしくは 0Dh
データ	P1 = 00h の場合、コマンドデータは RNDTarget P1 = 01 / 02h の場合、コマンドデータは RND_{Target} + ターゲットカードのシリアル（またはロット）番号です P1 = 03h の場合 P1 = 03h - ターゲットカードのシリアル番号（8 バイト） - ライトコマンド（A0 または A1）（1 バイト） - BNr（2 バイト） P1 = 80/81/82/88/89/8Ah： - ターゲットカードのシリアル番号（8 バイト） - 元の鍵 ID（元の鍵が格納されている SAM カードの鍵、00 = DESFire のデフォルト鍵 - カード） - 鍵番号（DESFire カード鍵番号） - Key Version（DESFire カードのキーバージョン、使用されない場合、値= 00） -コマンド：C6（オプション） -KeySetNo（オプション）

*このリストは、リストされているカードが違う Change Key と Authenticate Key を持っているか、または 2 つの鍵が同じ値を使用しているかを示しています。

特定の応答ステータスバイト

SW1 SW2	説明
69 85h	SAM セッションキーが準備完了していません
62 83h	現在の DF がブロックされているか、またはターゲット EF がブロックされている
69 86h	DF 未選択
69 81h	キーファイルのファイルタイプが間違っています。内部線形変数ファイル

SW1 SW2	説明
69 82h	ターゲットファイルのヘッダブロックのチェックサムが正しくないか、セキュリティ条件が満たされていません
6A 86h	P1 もしくは P2 無効
67 00h	P3 正しくない
6A 83h	ターゲットキーが準備されていないか、キーの長さが 16 未満です
61 1Ch	コマンド成功、GET RESPONS で結果を取得

5.3. 非接触スマートカード プロトコル

5.3.1. ATR の生成

リーダーが PICC を検出すると、PICC を識別するために、ATR が PCSC ドライバに送されます。

5.3.1.1. ATR フォーマット（ISO14443-3-3 PICC に適用）

バイト	数値	標記	説明
0	3Bh	最初のヘッダー	
1	8Nh	T0	ハイハーフバイト8は、以降にTA1、TB1、TC1が存在せず、TD1のみ存在することを示します。 ローハーフバイト N は歴史的なバイトの数を示します（HistByte 0 - HistByte N-1)
2	80h	TD1	ハイハーフバイト8は、TA2、TB2とTC2が存在せず、TD2のみ存在することを示します。 ローハーフバイ 0 はプロトコルタイプが T=0 を示す
3	01h	TD2	ハイハーフバイト0は、TA3、TB3、TC3およびTD3が全部存在しないのを示します。 ローハーフバイ 1 はプロトコルタイプが T=1 を示す
4 ~ 3+N	80h	T1	カテゴリインジケータバイトは、80のステータスインジケータが任意のCOMPACT-TLVデータオブジェクトに存在するかもしれない意味です
	4Fh	Tk	アプリケーション識別子にはインジケータが存在している
	0Ch		長さ
	RID		登録されたアプリケーションプロバイダ識別子 （RID）# A0 00 00 03 06
	SS		基準のバイト
	C0 ..C1h		カードネームバイト
	00 00 00 00h	RFU	RFU # 00 00 00 00
4+N	UU	TCK	T0からTkまでのすべてのバイトの排他的論理和

例：

MIFARE Classic 1KカードのATR = {3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6Ah}

その中：

長さ (YY) = 0Ch

RID = {{A0 00 00 03 06h} (PC/SCワークグループ)

基準 (SS) = 03h(ISO 14443A、3パート)

カードネーム(C0 ..C1) = [00 01h} (MIFARE Classic 1K)

基準 （SS） = 03h：ISO 14443A、3パート

= 11h：FeliCa

カードネーム (C0 ..C1)：

00 01：MIFARE Classic 1K
00 02：MIFARE Classic 4K
00 03：MIFARE Ultralight®
00 26：MIFARE Mini®
00 3A：MIFARE Ultralight® C
00 36：MIFARE Plus® SL1 2K
00 37：MIFARE Plus® SL1 4K
00 38：MIFARE Plus® SL2 2K
00 39：MIFARE Plus® SL2 4K
00 30：Topaz和Jewel
00 3B：FeliCa
FF 28：JCOP 30
FF [SAK]：定義されていないタグ

5.3.1.2. ATR フォーマット（ISO14443-4-3 PICC に適用）

<table>
<tr><th>バイト</th><th>数値</th><th>標記</th><th>説明</th></tr>
<tr><td>0</td><td>3Bh</td><td>最初のヘッダー</td><td></td></tr>
<tr><td>1</td><td>8Nh</td><td>T0</td><td>ハイハーフバイト8は、以降にTA1、TB1、TC1が存在せず、TD1のみ存在することを示します。
ローハーフバイト N は歴史的なバイトの数を示します（HistByte 0 - HistByte N-1)</td></tr>
<tr><td>2</td><td>80h</td><td>TD1</td><td>ハイハーフバイト8は、TA2、TB2とTC2が存在せず、TD2のみ存在することを示します。
ローハーフバイ 0 はプロトコルタイプが T=0 を示す</td></tr>
<tr><td>3</td><td>01h</td><td>TD2</td><td>ハイハーフバイト0は、TA3、TB3、TC3およびTD3が全部存在しないのを示します。
ローハーフバイ 1 はプロトコルタイプが T=1 を示す</td></tr>
<tr><td rowspan="2">4 ~ 3+N</td><td>XX</td><td>T1</td><td rowspan="2">歴史バイト：

ISO 14443-A：
ATS応答の歴史的なバイト。ISO 14443-4基準を参照してください。

ISO 14443-B：
<table>
<tr><th>バイト 1~4</th><th>バイト 5~7</th><th>バイト8</th></tr>
<tr><td>ATQBのアプリケーションデータ</td><td>ATQBからのプロトコル情報バイト</td><td>ハイハーフバイト =ATTRIBコマンドのMBLI；ローハーフバイ（RFU）=0</td></tr>
</table></td></tr>
<tr><td>XX</td><td>Tk</td></tr>
<tr><td>4+N</td><td>UU</td><td>TCK</td><td>T0からTkまでのすべてのバイトの排他的論理和</td></tr>
</table>

例1：
MIFARE® DESFire®のATR = {3B 81 80 01 80 80h} // 6 バイトのATR

注釈：*APDU"FF CA 01 00 00h"*を使用して、*ISO 14443A-4*の*PICC*に準拠しているまたは*ISO 14443B-4*の*PICC*に準拠しているを区別します。可能な場合、完全な*ATS*を取得します。*ISO 14443A-3*または*ISO 14443B-3/4*の*PICC*に準拠する場合、*ATS*が返される。

APDU コマンド = FF CA 01 00 00h
APDU 応答 = 06 75 77 81 02 80 90 00h
ATS = {06 75 77 81 02 80h}

例2：

EZ-LinkのATR = {3B 88 80 01 1C 2D 94 11 F7 71 85 00 BEh}

ATQBのアプリケーションデータ = 1C 2D 94 11h

ATQBのプロトコル情報 = F7 71 85h

ATTRIB の MBLI =00h

5.3.2. APDU、私有（Psedo）APDU およびカード固有（Native）コマンド

CCID ホストが PCSC API 中の SCardTransmit（）に対応する CCID メッセージ PC _ to _ RDR _ XfrBlock を使用して、カードリーダーにカードアドミニストレーションコマンドまたは APDU を送信することができる。PICC では、カードが ISO 14443 第 4 部分プロトコルまたは Innovation プロトコルをサポートしている場合、リーダ/ライタはプロトコルフレームにコマンド/APDU をパッケージ化してカードに直接送信し、コマンド/APDU を解析しません。カードが両方のプロトコルをサポートしていない場合、CCID ホストにメッセージ「6 A 81」が返されます。

注：Microsoft Window はスマートカードプラグアンドプレイに対応しているため、Microsoft Window はカードが提示された時にカードに APDU 命令を送信することがあります。この操作により、カードをリセットしない限り、DESFire カードが ISO APDU モードになり、固有コマンドを受信できなくなります。通常、Microsoft Window はカードが無反応状態になってから 10 秒後にカードをリセットします（PC _ to _ RDR _ IccPowerOff 経由）。

5.3.3. PICC の PCSC 私有（Pseudo）APDU（独自の拡張機能付き）

非接触カードへの間接アクセスには、以下の私有 APDU を使用する。CCID ホストが PCSC API 中の SCardTransmit（）に対応する CCID メッセージ PC _ to _ RDR _ XfrBlock を使用して、カードリーダーにこれらの APDU を送信することができる。私有 APDU を受信すると、リーダライタは低レベルのコマンドを解読し、カードに送信します。カードがこれらの低レベルコマンドを処理した後、リーダライタはカードレスポンスを収集し、レスポンスを作成して CCID ホストに返信する。

5.3.3.1. データ取得（Get Data）[FFCA…]

このコマンドは、シーケンス番号、プロトコルパラメータなど、アクティブ化中に取得したデータを読み込むために使用します。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get Data	FFh	CAh	下記の表を確認ください		00h（全長）

コマンドパラメーター

P1	P2	意味
00h	00h	カードの UID/PUPI/SN を取得する
01h	00h	A タイプ 4 パートの ATS を取得
02h	00h	下記のカードタイプ関連データを取得し、転送順序： A タイプ：2 バイト ATQA/ATVA + 4/7/10 バイト UID + 1 バイト最後の SAK。

P1	P2	意味
		B タイプ：12 バイト ATQB
80h	00h	下記のカードタイプ関連データを取得し、転送順序： A タイプ：2 バイト ATQA/ATVA + 4/7/10 バイト UID + 1/2/3 バイト SAK。 B タイプ：12 バイト ATQB FeliCa：17 バイト ATQ (+ 6 バイト ATTR、アクティブ化されている場合) SRI：8 バイト UID + 1 バイトチップ ID。 ISO15693：1 バイト DSFID + 8 バイト UID CTS：4 バイト SN + 2 バイト ATQT Innovatron：4 バイト SN + 1 バイトタプアドレス。

応答

応答	データ出力		
結果	データ	SW1	SW2

応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	6X XXh	失敗

例：

“接続された PICC”のシリアルナンバーを取得します

```
UINT8 GET_UID[5] = {FF, CA, 00, 00, 00};
```

“接続された ISO 14443-A PICC”の ATS を取得します

```
UINT8 GET_ATS[5] = {FF, CA, 01, 00, 00};
```

5.3.3.2. キーロード（Load Key）[FF 82 …]

このコマンドは、キーバッファ番号で指定された内部キーバッファにキーデータをロードするために使用します。鍵バッファは失いやすいストレージに属し、認証に使用されるコンテンツが含まれています。このコマンドはカードデータ転送をしません。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Load Authentication Keys	FFh	82h	00h	キーバッファ番号（0-1）	キー長さ	キー

キー長さ/データ

カードタイプ	キー長さ(Lc)	キーデータ（転送/保存順番）
MIFARE Standard MIFARE Plus SL1	06h	6 バイト Crypto1 Key A/B。
MIFARE Plus SL1 MIFARE Plus SL2	16h	6 バイト Crypto1 Key A/B + 16 バイト AES Key。
MIFARE Plus SL2	06h	6 バイト暗号化 Crypto1 Key A/B。
MIFARE Ultralight C MIFARE DESFire	10h	16 バイト 2K3DES Key。

応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	6X XXh	失敗

例：

// 失いやすいキーのメモリに 00h キーをロードする {FF FF FF FF FF FFh}。

APDU = {FF 82 00 00 06 FF FF FF FF FF FFh}

5.3.3.3. 認証（Authenticate）[FF 86 00 00 05 …]

このコマンドは、カードに認証プロセスを実行するために使用します。認証が成功すると、ユーザーは保護されたブロック/ページにアクセスできます。コマンドを送信する前に、ユーザーは Load Key コマンドを使用して正しいキーデータを指定した**キーバッファ番号**バッファにロードする必要があります。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Authenticate	FFh	86h	00h	00h	05h	下記の表を確認ください

コマンドデータ

バイト 0	バイト 1	バイト 2	バイト 3	バイト 4
01h	00h (RFU)	アドレス	キーのタイプ	キーバッファ番号

アドレスとキーのタイプ

カードタイプ	アドレス	キーのタイプ
MIFARE Standard MIFARE Plus SL1 MIFARE Plus SL2	00h~FFh：ブロック 0~255	60h：Crypto1 Key A 61h：Crypto1 Key B
MIFARE UltraLightC	00h (RFU)	80h：2K3DES
MIFARE DESFire	00h~0Eh：DESFire キー番号 0~14	0Ah：2K3DES

応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	6X XXh	失敗

セクター (16 個のセクター，各セクターには 4 個の連続的なブロックが含めている)	データブロック (3 つのブロック，各には 16 バイト)	トレーラーブロック (1 つのブロック，16 バイト)
セクター0	00h – 02h	03h
セクター1	04h – 06h	07h
..	..	..
..	..	..
セクター14	38h – 0Ah	3Bh
セクター15	3Ch – 3Eh	3Fh

表 4：MIFARE Classic 1K カードのメモリマップ

セクター (32 個のセクター，各セクターには 4 個の連続的なブロックが含めている)	データブロック (3 つのブロック，各には 16 バイト)	トレーラーブロック (1 つのブロック，16 バイト)	
セクター0	00h ~ 02h	03h	2 KB
セクター1	04h ~ 06h	07h	
..			
..			
セクター30	78h ~ 7Ah	7Bh	
セクター31	7Ch ~ 7Eh	7Fh	

セクター (8 個のセクター，各セクターには 16 個の連続的なブロックが含めている)	データブロック (15 つのブロック，各には 16 バイト)	トレーラーブロック (1 つのブロック，16 バイト)	
セクター32	80h ~ 8Eh	8Fh	2 KB
セクター33	90h ~ 9Eh	9Fh	
..			
..			
セクター38	E0h ~ EEh	EFh	
セクター39	F0h ~ FEh	FFh	

表 5：MIFARE Classic 4K カードのメモリマップ

バイトナンバー	0	1	2	3	ページ
シリアルナンバー	SN0	SN1	SN2	BCC0	0
シリアルナンバー	SN3	SN4	SN5	SN6	1
内部/ロック	BCC1	Internal	Lock0	Lock1	2
OTP	OPT0	OPT1	OTP2	OTP3	3
データ読み取り/書き込み	Data0	Data1	Data2	Data3	4
データ読み取り/書き込み	Data4	Data5	Data6	Data7	5
データ読み取り/書き込み	Data8	Data9	Data10	Data11	6
データ読み取り/書き込み	Data12	Data13	Data14	Data15	7
データ読み取り/書き込み	Data16	Data17	Data18	Data19	8
データ読み取り/書き込み	Data20	Data21	Data22	Data23	9
データ読み取り/書き込み	Data24	Data25	Data26	Data27	10
データ読み取り/書き込み	Data28	Data29	Data30	Data31	11
データ読み取り/書き込み	Data32	Data33	Data34	Data35	12
データ読み取り/書き込み	Data36	Data37	Data38	Data39	13
データ読み取り/書き込み	Data40	Data41	Data42	Data43	14
データ読み取り/書き込み	Data44	Data45	Data46	Data47	15

512 ビット
または
64 バイト

表 6：MIFARE Ultralight カードのメモリマップ

例：

// {TYPE A，キーナンバーの 00h}によって、ブロック 04h を認証します。PC/SC V2.07

APDU = {FF 86 00 00 05 01 00 04 60 00h}

注：*MIFARE Ultralight* のメモリは自由にアクセスできます。認証はいりません。

5.3.3.4. バイナリブロック読み取り（Read Binary Blocks）[FF B0...]

このコマンドは、指定されたブロック/ページアドレスの場所から指定されたバイトのデータを PICC から読み出すために使用します。カードの種類によっては、このコマンドを呼び出す前に、ユーザーはまず認証を行い、これらのブロック/ページへのアクセス権を取得する必要がある場合があります。

コマンド：

コマンド	CLA	INS	P1	P2	Le
Read Binary Blocks	FFh	B0h	モードとアドレス		更新していないバイト

P1/P2（モードとアドレス）

カードタイプ	P1[7:4] モード	P1[3:0] + P2[7:0] 開始アドレス（MSB が前）
MIFARE Standard MIFARE Plus SL1 MIFARE Plus SL2	0h：テールブロックをスキップ 8h：テールブロックを含む	000h~0FFh：ブロック 0~255
MIFARE Ultralight MIFARE Ultralight C	0h（保留）	000h~02Fh：ページ 0~47
SRIX4K/SRT512	0h（保留）	000h~07Fh：ブロック 0~127 0FFh：システムゾーン
PicoPass	0h（保留）	000h~0FFh：ブロック 0~255
ISO15693	0h（保留）	000h~7FFh：ブロック 0~2047
Topaz/NFC Type-1 タブ	0h（保留）	000h~7FFh：バイトアドレス
CTS	0h（保留）	000h~01Fh：ブロック 0~31

Le（読み取り待ちのバイト数）

タイプ	バイト 0	バイト 1	バイト 2
短い	00h：256 バイトを読み取る 01h~FFh：1~255 バイトを読み取る	--	
長い	00h	0000h：65536 バイトを読み取る 0001h~~FFFFh：1~65535 バイトを読み取る	

応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	6X XXh	失敗

例：

// バイナリブロック 04h から 16 バイトを読み取る（MIFARE Classic　1K または 4K）

APDU = FF B0 00 04 10h

// バイナリブロック 80h から 240 バイトを読み出す（MIFARE Classic 4K）

// ブロック 80 からブロック 8Eh まで（15 個ブロック）

APDU = FF B0 00 80 F0h

5.3.3.5.　バイナリブロック更新（Update Binary Blocks）[FF D6 …]

このコマンドは、指定されたブロック/ページアドレスの場所から指定されたバイトのデータを PICC に書き込むために使用します。カードの種類によっては、このコマンドを呼び出す前に、ユーザーはまず認証を行い、これらのブロック/ページへのアクセス権を取得する必要がある場合があります。

カード内のブロック/ページへのデータの書き込みは、カードのセキュリティ設定（例えば MIFARE カードの末尾ブロック）を変更する可能性があるので、特に注意してください。誤ったデータを書き込むか、操作が失敗すると、カードがロックされる可能性があります。カードロックされるリスクを軽減するために、セキュリティブロック/ページに関わる場合には、1 つの APDU コマンドを使用して複数のブロック/ページにデータを書き込むことはお勧めしません。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Update Binary Blocks	FFh	D6h	モードとアドレス		書き込み待ちバイト	データバイト

P1/P2（モードとアドレス）と Write Size 一致（ブロック/ページ容量）

カードタイプ	P1[7:4] モード	P1[3:0] + P2[7:0] 開始アドレス（MSB が前）	ブロック/ページの容量（バイト）
MIFARE Standard MIFARE Plus SL1 MIFARE Plus SL2	0h：テールブロックをスキップ 8h：テールブロックを含む	000h~0FFh：ブロック 0~255	16
MIFARE Ultralight MIFARE UltraLightC	0h（保留）	000h~02Fh：ページ 0~47	4
SRIX4K/SRT512	0h（保留）	SRIX4K/SRT512	4
PicoPass	0h（保留）	PicoPass	8

カードタイプ	P1[7:4] モード	P1[3:0] + P2[7:0] 開始アドレス（MSB が前）	ブロック/ページの容量 （バイト）
ISO15693	0h（保留）	000h~7FFh：ブロック 0~2047	ISO15693
Topaz/NFC Type-1 タブ	0h：消去を含む	000h~7FFh：バイトアドレス	1(アドレス 78h)また 8(その他)
	8h：消去を含まない		
CTS	0h（保留）	000h~01Fh：ブロック 0~31	CTS

Lc（書き込み待ちバイト）

タイプ	バイト 0	バイト 1	バイト 2
短い	01h~FFh：1~255 バイトバイト書き込み待ち	--	
長い	00h	0001h~FFFFh：1~65535 バイトバイト書き込み待ち	

応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	6X XXh	失敗

例：

// MIFARE Classic 1K/4K カード中のバイナリブロック 04h のデータを{00 01 ..0Fh}に更新します

APDU = {FF D6 00 04 10 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0Fh}

//MIFARE Ultralight 中のバイナリブロック 04 h を{00 01 02 03}に更新する

APDU = {FF D6 00 04 04 00 01 02 03h}

5.3.4. 透過（Pass Though）コマンド

5.3.4.1. ISO14443-3 タブアクセス

この節では、ISO 14443-3 規格に準拠したコマンドを送信するために透過コマンド構造を使用する方法について説明する。ファームウェアに対する要求は次の通りです：

- ACR1581U-C　FW 1.09.00 及びその以降

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
ISO14443-3 コマンド	FFh	00h	00h	00h	データ長さ	データ

応答

応答	データ出力		
結果	データ	SW1	SW2

応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	6X XXh	失敗

5.3.4.1.1. ISO14443-3 固有コマンドを透過コマンドに転換

本節は ISO14443-3 固有コマンドを透過コマンドに転換する手順を説明する。

例：

Mifare Ultralight AES バージョン号を取得

60

透過フォーマット

FF 00 00 00 01 60

5.3.4.2. ISO15693 タグのアクセス

この節では、ISO15693 規格に準拠したコマンドを送信するために透過コマンド構造を使用する方法について説明する。ファームウェアに対する要求は次の通りです：

- ACR1581U-C FW 1.08.00 及びその以降

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン		Le
ISO15693 コマンド	FFh	FBh	00h	フラグ	データ長さ＋ 1	コマンドコード	データ	--/00h

その中：

フラグ： 00h（デフォルト）：フラグを 22h に設定する。この場合、データフィールドに UID を含める必要はありません。

注意：このオプションはカスタム（Custom）またはプライベート（Proprietary）コマンドとは併用できません。

その他：この値をフラグの値として使用する

応答

応答	データ出力		
結果	データ	SW1	SW2

応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	6X XXh	失敗

5.3.4.2.1. ISO15693 コマンドを透過コマンドに転換

本節 ISO15693 コマンドを透過コマンドに転換する手順を説明します。なお、透過コマンドモードは 26 kbps レートのコマンドのみをサポートしている。53 kbps のコマンドを使用する場合は、*Error! Reference source not found.*を参照してください。

例 1：

沈黙保つ（強制コマンド）

22 02 XX XX XX XX XX XX XX XX

その中：

22　フラグビット

02　沈黙保つコマンドコード

XX XX XX XX XX XX XX XX　　8 バイト UID

透過フォーマット

FF FB 00 22 09 02 XX XX XX XX XX XX XX XX

または

FF FB 00 00 01 02

例 2：

非アドレス指定で複数のデータブロックを読み込む（オプションコマンド）

02 23 00 01

その中：

02　フラグビット

23　複数ブロックの読み込みコマンドコード

00　1 つ目のブロックの番号

01　2 個ブロックを読み取る

透過フォーマット

FF FB 00 02 03 23 00 01

例 3：

選択した状態で乱数を取得する（NXP SLIX 2 カスタムコマンド）

12 B2 04

その中：

12　フラグビット

B2　ランダム数を取得コマンドコード

04　　NXP 製造メーカーコード

透過フォーマット

FF FB 00 12 02 B2 04

5.3.4.2.2. サポートコマンドリスト

以下の表に ISO15693-3 標準で定義されたすべてのコマンドが記載されている。『N/A』でマークされているコマンドは未テストの意味です。これらのコマンドに関するご質問がございましたら、メールで info@acs.com.hk までご連絡ください。注意：ISO15693 カードはすべてのオプションコマンドをサポートしない場合があります。具体的なサポート内容については、対応カードの技術仕様書をご参照ください。

コマンドコード	タイプ	機能	サポート
01h	強制	点検（インベントリ）	YES
02h	強制	沈黙保つ（Stay Quiet）	YES
20h	オプショナル	単一ブロックを読み取る（Read Single Block）	YES
21h	オプショナル	単一ブロックを書き込み（Write Single Block）	YES
22h	オプショナル	ロックブロック（Lock Block）	YES
23h	オプショナル	複数のブロックを読み取る（Read Multiple Blocks）	YES
24h	オプショナル	複数のブロックを書き込み（Write Multiple Blocks）	YES
25h	オプショナル	選択（Select）	YES
26h	オプショナル	準備完了にリセット（Reset to Ready）	YES
27h	オプショナル	AFI 書き込み（Write AFI ）	YES
28h	オプショナル	AFI ロック（Lock AFI）	YES
29h	オプショナル	DSFID 書き込み（Write DSFID ）	YES
2Ah	オプショナル	DSFID ロック（Lock DSFID）	YES
2Bh	オプショナル	システム情報読み取り（Get System Information）	YES
2Ch	オプショナル	複数のブロックのセキュリティ状態を取得（Get Multiple Block Security Status）	YES
2Dh	オプショナル	迅速に複数のブロックを読み取る（Fast Read Multiple Blocks）	NO
30h	オプショナル	拡張された単一ブロックの読み取り（Extended Read Single Block）	YES
31h	オプショナル	拡張された単一ブロックの書き込み（ExtendedWrite Single Block）	YES
32h	オプショナル	拡張されたロックブロック（Extended Lock Block）	YES
33h	オプショナル	拡張された複数のブロック読み取り（Extended Read Multiple Blocks）	YES

コマンドコード	タイプ	機能	サポート
34h	オプショナル	拡張された複数のブロック書き込み（Extended Write Multiple Blocks）	YES
35h	オプショナル	認証（Authenticate）	YES
36h	オプショナル	キー更新（KeyUpdate）	N/A
37h	オプショナル	認証通信暗号化フォーマットインジケータ（AuthComm Crypto Format Indicator）	N/A
38h	オプショナル	安全通信暗号化フォーマットインジケータ（SecureComm Crypto Format Indicator）	N/A
39h	オプショナル	チャレンジ（Challenge）	NO
3Ah	オプショナル	バッファの読み込み（ReadBuffer）	YES
3Bh	オプショナル	拡張されたシステム情報読み取り（Extended Get System Information）	YES
3Ch	オプショナル	拡張された複数のブロックのセキュリティ状態を取得（Extended Get Multiple Block Security Status）	YES
3Dh	オプショナル	迅速に拡張された複数のブロック読み取り（Fast Extended Read Multiple Blocks）	NO
A0-DFh	カスタマイズ	IC 製造メーカー関係データ（IC Mfg Dependent）	N/A
E0-FFh	専用	IC 製造メーカー関係データ（IC Mfg Dependent）	N/A

5.3.4.3. FeliCa タグのアクセス

FeliCa タグをアクセスするコマンドは、PCSC タグと MIFARE タグをアクセスするコマンドと違います。このコマンドは FeliCa 基準に準拠して、ヘッダが追加されています。

FeliCa コマンドのフォーマット

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
FeliCa コマンド	FFh	00h	00h	00h	データの長さ	FeliCa コマンド（長さバイトで始まる）

FeliCa の応答データフォーマット（データ+2 バイト）

応答	データ出力
結果	応答データ

例のメモリブロックデータの読み取り

1. FeliCa を接続する。

   ATR = 3B 8F 80 01 80 4F 0C A0 00 00 03 06 11 00 3B 00 00 00 00 42h

   その中：11 00 3Bh = FeliCa

2. FeliCa IDM の読み取り。

   コマンド = FF CA 00 00 00h

   応答 = [IDM (8 バイト)] 90 00h

   例：FeliCa IDM = 01 01 06 01 CB 09 57 03h

3. FeliCa コマンドアクセス。

   例：メモリブロックデータの「読み取り」

   コマンド = FF 00 00 00 10 10 06 01 01 06 01 CB 09 57 03 01 09 01 01 80 00h

   その中：

   Felica コマンド = 10 06 01 01 06 01 CB 09 57 03 01 09 01 01 80 00h

   IDM = 01 01 06 01 CB 09 57 03h

   応答 = メモリブロックデータ

5.3.5. PCSC 2.0 パート 3 サポートできる APDU コマンド（V2.02 及びその以降のバージョン）

PCSC 2.0 パート 3 のコマンドが透過的にアプリケーションからデータを非接触非タグへ渡し、アプリケーションとプロトコルに透過的に受信したデータを返し、同時にプロトコルを切り替えるために使用されています。

5.3.5.1. PCSC 2.0 第 3 部流れ

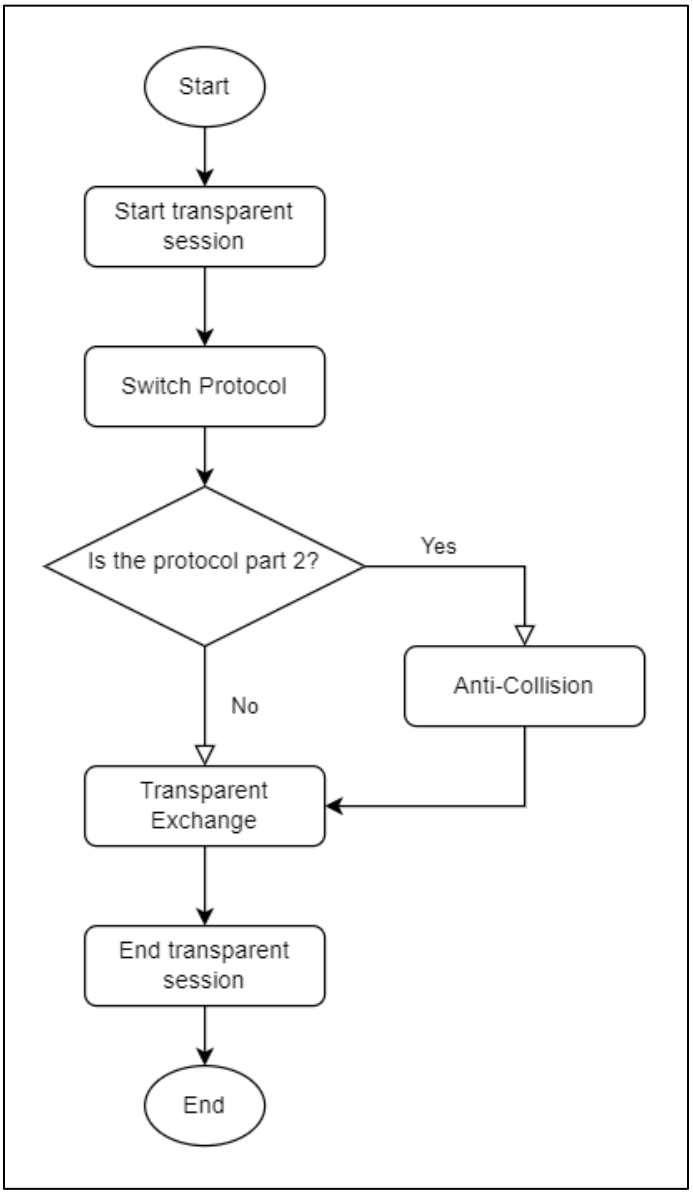

図 5：ACR1581U 透過セッションフローチャート

5.3.5.2. コマンドと応答の APDU フォーマット

コマンドのフォーマット

CLA	INS	P1	P2	Lc	コマンドデータイン
FFh	C2h	00h	機能	データ長さ	データ[データの長さ]

その中：機能 （1 バイト）：

00h = セッション管理
01h = 透明インタラクティブ
02h = プロトコルの切り替え
他 = RFU

応答フォーマット

データ出力	SW1	SW2
BER-TLV 符号化されたデータフィールド		

すべてのコマンドは、レスポンスデータフィールド（利用可能な場合）と一緒に SW1 と SW2 を返します。SW1 と SW2 は ISO7816 準拠以下の C0 データオブジェクトの SW1 SW2 も使用する必要があります。

C0 データ要素のフォーマット

タグ	長さ（1 バイト）	SW2
C0h	03h	エラーステータス

エラーステータスの説明

エラーステータス	説明
XX SW1 SW2	XX = APDU 内エラーデータオブジェクトの番号 00 = APDU の一般的なエラー 01 = 一番目のデータオブジェクト内にエラーがある 02 = 二番目のデータオブジェクト内にエラーがある
00 90 00h	エラーは発生していません
XX 62 82h	データオブジェクトの XX 警告、要求された情報は存在していません
XX 63 00h	情報なし
XX 63 01h	実行は他のデータオブジェクトの障害のため停止しました
XX 6A 81h	データオブジェクトはサポートされていません XX
XX 67 00h	予期しない長さのデータオブジェクト XX
XX 6A 80h	予期しない値のデータオブジェクト XX
XX 64 00h	データオブジェクト XX 実行エラー（IFD から応答ない）

エラーステータス	説明
XX 64 01h	データオブジェクト XX 実行エラー（ICC から応答ない）
XX 6F 00h	データオブジェクト XX は正確な診断なしで失敗しました

最後の 2 バイトは、エラーについての説明で、一番目のバイトの数値が誤ったデータオブジェクトの XX の番号を表します。ISO7816 に基づいて、SW1 SW2 の値が許可されています。

C-APDU データフィールドには複数のデータオブジェクトがあって、1 つのデータオブジェクトが失敗した場合、他のデータオブジェクトが失敗したデータオブジェクトに依存しない場合、IFD は次のデータオブジェクトを処理することができます。

5.3.5.3. セッション管理（Manage Session）[FF C2 00 00 ...]

このコマンドでユーザーがセッションを開始し、ポーリング機能を無効にして、後続の通信を行うことができます。通信が完了したら、ユーザーはすぐにセッションを終了する必要があります。

正しく使用されていない場合、リーダ/ライタがカードの存在を検出できず、論理的/物理的にリーダ/ライタ接続を切断しない限り、自動的にリカバリできないことに注意してください。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン	Le
Manage Session	FFh	C2h	00h	00h	Cmd データの長さ	Cmd TLV	--/00h

応答コード

応答データ	SW1 SW2	意味
--	90 00h	操作が成功に完了しました。
Rsp TLV	90 00h	Le = 0x00：Cmd TLV の一つが失敗したエラーの詳細については、Rsp TLV を参照してください。
--	6X XXh	Le = --：Cmd TLV の一つが失敗した

Cmd TLV

Cmd	意味
Start Session: 81 00h	セッションを開始し、ポーリングを無効にします。
RF Off: 83 00h	RF をオフにする
Timer: 5F 46 04h [TIME]	次の RF On/Off TLV 前のスリープ時間を設定する ［TIME］：4 バイトの値（MSB の前）、範囲は 1000～100000 us。実際のスリープ時間は、最も近い 1000 us に四捨五入されます。
RF On: 84 00h	RF をオンにする
End Session: 82 00h	セッションを終えて、ポーリングを有効にします。

Rsp TLV

Rsp	意味
TLV Error: C0 03 NN 6X XXh	NN 目のコマンド TLV エラー。

5.3.5.3.1. セッションデータオブジェクトを開始する（Start Session Data Object）

このコマンドは、透過的なセッションを開始するために使用されています。セッションが開始されると、セッションが終了されるまで、自動ポーリングが無効になります。

セッションデータオブジェクトを開始する

タグ	長さ（1 バイト）	数値
81h	00h	-

5.3.5.3.2. セッションデータオブジェクトを終了する（End Session Data Object）

このコマンドは、透過的なセッションを終了するために使用されています。セッションが開始される前に自動ポーリング状態にリセットされます。

セッションデータオブジェクトを終了する

タグ	長さ（1 バイト）	数値
82h	00h	-

5.3.5.3.3. RF データオブジェクトをオフにする（Turn Off the RF Data Object）

このコマンドはアンテナフィールドをオフにする時に使われます。

RF データオブジェクトをオフにする

タグ	長さ（1 バイト）	数値
83h	00h	-

5.3.5.3.4. RF データオブジェクトをオンにする（Turn O n the RF Data Object）

このコマンドはアンテナフィールドをオンにする時に使われます。

RF データオブジェクトをオンにする

タグ	長さ（1 バイト）	数値
84h	00h	-

5.3.5.3.5. タイマーデータオブジェクト（Timer Data Object）

このコマンドは、1 µs の単位で 32 ビットのタイマーデータオブジェクトを作成するために使用されます。

例： RF データオブジェクトを閉じると、RF データオブジェクトを開く間に 5000µs のタイマーデータオブジェクトがある場合、リーダライタは RF フィールドを約 5000µs 閉じてから、RF フィールドを再開する。

タイマーデータオブジェクト

タグ	長さ（1 バイト）	数値
5F 46h	04h	タイマー（4 バイト）

5.3.5.4. 透明交換（Transparent Exchange）[FF C2 00 01 …]

このコマンドにより、ユーザーはカードに任意のビットまたはバイトを送信/受信することができ、オプションで ISO 14443 パート 4 などの様々なリンクおよびトランスポート層、およびいくつかのリンク層冗長性（CRC およびパリティ）を構成することができます。ユーザは、任意のカード固有の生データをこのプライベート APDU に埋め込み、カードに送信することができます。

注意する必要があるのは、このコマンドがカードのサポートの内部処理プロセスに干渉する可能性があり、ドライバ/ファームウェアに通知せずにカードの状態を変更する可能性があり、ドライバ/ファームウェアを正常に戻すためにカードをリセットおよび/または削除する必要がある可能性があります。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン	Le
Transparent Exchange	FFh	C2h	00h	01h	Cmd データの長さ	Cmd TLV	00h

応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	6X XXh	失敗

Cmd TLV

Cmd	意味
Transceive Flag: 90 02 [Flag] 00h	下記の Transceive TLV の Flag を設置する Flag[7:5]: RFU; 0 に設置する Flag[4]: ISO14443-4 を無効に設置する Flag[3]: パリティ照合処理の受信を禁止に設置する Flag[2]: パリティ照合処理の転送を禁止に設置する Flag[1]:CRC 処理の受信を禁止に設置する

Cmd	意味
	Flag[0]: CRC 処理の転送を禁止に設置する この TLV が見つからない場合は、前のコマンドで設定した Flag 値を使用します。 Flag 値を設定したことがない場合は、現在のプロトコル値を使用します。
Transmit Bit Frame: 91 01h [NumBit]	下記の Transceive TLV の Bit Fram を設置する。この TLV が見つからない場合、デフォルト値は 0 です。 NumBit[7:3]: RFU; 0 に設置する NumBit[2:0]:最後のバイトの有効ビットの数（0 はすべてのビットが有効であることを意味します）
Timer: 5F 46 04h [TIME]	下記の Transceive TLV のタイムアウト時間を設置する。 ［TIME］：4 バイトの値（MSB の前）、範囲は 1 us～1000000 us。実際のタイムアウト時間は、最も近い 302.07 x 20~15 us に四捨五入されます。 この TLV が見つからない場合は、以前設定した FWTI 値をタイムアウト時間として使用します。
Set FWTI: FF 6E 03 03 01h [FWTI]	Transceive の FWT/タイムアウトを設置する。以前に「FF C 2 h...」コマンドで FWTI を設定していない場合、デフォルト値は 0 になります。 FWTI: 0 ~ 15, FWT/タイムアウト = 302.07 x 2FWTI us
Transceive: 95h [Size] [Data]	SizeBER-TLV 長さフィールド中の符号化データのサイズ Data 転送待ちのデータ

Rsp TLV

Rsp	意味
Receive Bit framing: 92 01h [NumBit]	NumBit[7:3]: RFU; 0 に設置する。 NumBit[2:0]:最後のバイトの有効ビットの数（0 はすべてのビットが有効であることを意味します）
Response: 95h [Size] [Data]	Size: BER-TLV 長さフィールド中の符号化データのサイズ Data 受信データ。
Response Status: 96 02h [Status] 00h	Status [7:4]: RFU. Status[3]: フレーミングエラー Status[2]:パリティエラー。 Status [1]: RFU. Status [0]: CRC エラー

5.3.5.4.1. 送受信のフラグデータオブジェクト（Transmission and Reception Flag Data Object）

このコマンドは、次の送信のためのフレーミングおよび RF パラメータを定義するために使用されます。

送受信のフラグデータオブジェクト

タグ	長さ（1 バイト）	数値		
		バイト 0		バイト 1
		ビット	説明	
90h	02h	0	0 – 送信したデータに CRC を追加します 1 – 送信したデータに CRC を追加しません	00h
		1	0 – 受信したデータに対して CRC 検査を実施する 1 – 受信したデータに対して CRC 検査を実施しない	
		2	0 – 送信したデータにパリティを挿入します 1 – 送信したデータにパリティを挿入しません	
		3	0 – 受信したデータのパリティを期待します 1 – パリティを期待していません（パリティチェックなし）	
		4	0 – 送信したデータのプロトコルプロローグを追加したり、応答から捨てます 1 - 追加またはプロトコルのプロローグを破棄することはありません（ある場合）（例えば、PCB、CID、NAD）	
		5-7	RFU	

5.3.5.4.2. ビットフレーミングデータオブジェクトを送信する（Transmission Bit Framing Data Object）

このコマンドは、送受信されていないデータの最後のバイトの有効ビット数を定義するために使用されます。

ビットフレーミングデータオブジェクトを送信する

タグ	長さ（1 バイト）	数値	
		ビット	説明
91h	01h	0-2	最後のバイトの有効ビット数（0 はすべてのビットが有効であることを意味します）
		3-7	RFU

伝送ビットフレーミングデータオブジェクトは、「送信」または「送受信」のみのデータオブジェクトと一緒でなければなりません。このデータオブジェクトが存在しない場合、それはすべてのビットが有効であることを意味します。

5.3.5.4.3. データオブジェクトを送受信する（Transceive Data Object）

このコマンドは、ICC からのデータを送受信するために使用されます。送信が完了すると、リーダーは、タイマーデータオブジェクトに指定された時間まで待機します。

何のタイマデータオブジェクトは、データフィールドで定義されていない場合、リーダーは Set Parameter FWTI データオブジェクトに指定された期間を待っています。FWTI が設定されていない場合、リーダーは、約 302 µs を待ちます。

データオブジェクトを送受信する

タグ	長さ		数値
95h	01-7Fh		データ（1~127 バイト）
95h	81h	80h もしくはもっと	データ（128~N バイト）

5.3.5.4.4. タイマーデータオブジェクト（Timer Data Object）

このコマンドは、1 µs の単位で 32 ビットのタイマーデータオブジェクトを作成するために使用されます。

例えば、5000 µs のタイマーデータオブジェクトがある場合、リーダー／ライターは次の Transceive TLV を待ち、5000µs 経過後にタイムアウトする。

タイマーデータオブジェクト

タグ	長さ（1 バイト）	数値
5F 46h	04h	タイマー（4 バイト）

5.3.5.4.5. ビットフレーミングデータオブジェクトを応答する（Response Bit Framing Data Object）

このコマンドは、応答中に受信した送信ビットフレームデータオブジェクトを提示するために使用します。

タグ	長さ（1 バイト）	数値	
		ビット	説明
92h	01h	0-2	最後のバイトの有効ビット数（0 はすべてのビットが有効であることを意味します）
		3-7	RFU

伝送ビットフレーミングデータオブジェクトは、「送信」または「送受信」のみのデータオブジェクトと一緒でなければなりません。このデータオブジェクトが存在しない場合、それはすべてのビットが有効であることを意味します。

5.3.5.4.6. ステータスデータオブジェクトを応答する（Response Status Data Object）

このコマンドは、応答中に受信したデータの状態を提示するために使用します。

ステータスデータオブジェクトを応答する

タグ	長さ（1 バイト）	数値		
		バイト 0		バイト 1
		ビット	説明	
96h	02h	0	0 - CRC が OK、若しくはチェックしていません 1 - CRC チェックが失敗しました	RFU
		1	0 – 衝突なし 1 – 衝突が検出されました	
		2	0 – パリティエラーなし 1 – パリティエラーが検出されました	
		3	0 – フレームエラーなし 1 – フレームエラーが検出されました	
		4 - 7	RFU	

5.3.5.4.7. データオブジェクトを応答する（Response Data Object）

このコマンドは、応答中に受信したデータの状態を提示するために使用します。

データフォーマットを応答する

タグ	長さ		数値
97h	01-7Fh		応答データ（1-127 バイト)
	81h	80 - FFh	応答データ（128-255 バイト)
	82h	- -	応答データ（256-N バイト)

5.3.5.5. プロトコル切り替え（Switch Protocol）[FF C2 00 02 ...]

このコマンドを使用すると、ユーザーはプロトコルを切り替えて指定し、プロトコル・レベルとパラメータを選択できます。

注意する必要があるのは、このコマンドがカードのサポートの内部処理プロセスに干渉する可能性があり、ドライバ/ファームウェアに通知せずにカードの状態を変更する可能性があり、ドライバ/ファームウェアを正常に戻すためにカードをリセットおよび/または削除する必要がある可能性があります。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン	Le
Switch Protocol	FFh	C2h	00h	02h	Cmd データの長さ	Cmd TLV	00h

応答コード

応答データ	SW1 SW2	意味
Rsp TLV	90 00h	データ成功。
--	90 00h	成功
--	6X XXh	失敗

Cmd TLV

Cmd	意味
Set Baud: FF 6E 03 05 01h [Baud]	プロトコル切り替え時に使用するパート 4/レベルのボーレートを設定します。“FF C2h ...”コマンドで[Baud]を設定していない場合、デフォルト値は 98h (106 kbps)になります。 ISO14443：98h (106 kbps)、99h (212 kbps)、9A (424 kbps)、9B (848 kbps)。 ISO15693：80h (26 kbps)、08h (53 kbps)
Switch Protocol: 8F 02h [RF] [Layer]	プロトコルを指定の RF と/もしくはレベルに切り替える [RF]: 00hISO14443A, 01h: ISO14443B 02h: ISO15693, 03h: FeliCa, FFh: 現在の[RF]: 他：RFU [Layer]: 02h: レイヤ 2/セクション 03h: レイヤ 3/セクション 04h: レイヤ 4/セクション (A/B のみ適用) その他 ： RFU

Cmd	意味
	注意：レイヤ 2/セクションに切り替えた場合は、トランスペアレントセッション（ポーリング無効）状態である必要があります。

Rsp TLV

Rsp	意味
Response: 8Fh [Size] [Data]	Size: BER-TLV 長さフィールド中の符号化データのサイズ DataATR（パート 4 の場合）、最終 SAK（クラス A のパート 3 の場合）、または ATQB の PI（クラス B のパート 3 の場合）。

5.3.5.5.1. プロトコルデータオブジェクトを切り替える（Switch Protocol Data Object）

このコマンドは、プロトコルおよび規格の異なる層を指定するために使用されます。

プロトコルデータオブジェクトを切り替える

タグ	長さ（1 バイト）	数値	
		バイト 0	バイト 1
8Fh	02h	00h – ISO/IEC14443 A タイプ 01h – ISO/IEC14443 B タイプ 02h – ISO15693 03h – FeliCa 他 – RFU	02h – 2 層に切り替える 03h – 3 層に切り替えるまたは活性化する 04h – 4 層に活性化する 他 - RFU

5.3.5.5.2. データオブジェクトを応答する（Response Data Object）

このコマンドは、応答中に受信したデータの状態を提示するために使用します。

データフォーマットを応答する

タグ	長さ（1 バイト）	数値
5F 51h	データ長さ	ATR
8Fh	データ長さ	最終 SAK（クラス A のパート 3 の場合）、または ATQB の PI（クラス B のパート 3 の場合）。

5.3.5.6. PCSC 2.0 パート 3 の例

8. 透明セッション開始する

   コマンド：**FF C2 00 00 02 81 00**

   応答：**C0 03 00 90 00 90 00**

9. アンテナフィールドをオフにする

   コマンド：**FF C2 00 00 02 83 00**
   応答：**C0 03 00 90 00 90 00**

10. アンテナフィールドをオンにする

    コマンド：**FF C2 00 00 02 84 00**

    応答：**C0 03 00 90 00 90 00**

11. アクティベーISO 14443-4A

    コマンド： **FF C2 00 02 04 8F 02 00 04**

    応答： **C0 03 01 64 01 90 00** （カードがない場合）

    C0 03 00 90 00 5F 51 [Len] [ATR] 90 00

12. 0AH に PCB を設定し、送信データで CRC、パリティ、プロトコルプロローグを有効にします。

    コマンド：**FF C2 00 01 0A 90 02 00 00 FF 6E 03 07 01 0A**

    応答：**C0 03 00 90 00 90 00**

13. カードに APDU「80B2000008」を送信し、応答を取得します。

    コマンド：**FF C2 00 01 0E 5F 46 04 40 42 0F 00 95 05 80 B2 00 00 08**

    応答：**C0 03 00 90 00 92 01 00 96 02 00 00 97 0C [カードの応答] 90 00**

14. 透明セッションを終了する。

    コマンド：**FF C2 00 00 02 82 00**

    応答：**C0 03 00 90 00 90 00**

5.3.6. PICC の専属の私有（Pseudo）APDU

以下私有 APDU は非接触カードへの間接アクセスに使用され、PCSC Pseudo APDU への追加です。これらの APDU の内部処理プロセスは PCSC Pseudo APDU と同様です。

5.3.6.1. 値ブロック書き込み（Write Value Block）[FF D7 …]

このコマンドは、MIFARE 規格に準拠したカードのブロックに 4 バイトの値を書き込むために使用します。このコマンドを呼び出す前に、ユーザーはまず認証を行い、このブロックにへのアクセス権を取得する必要があります。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Write Value Block	FFh	D7h	00h	ブロック番号	05h	下記の表を確認ください

コマンドデータ

バイト 0	バイト 1	バイト 2	バイト 3	バイト 4
00h	4 バイト値（MSB の前）			

例 1： Decimal –4 = {FFh, FFh, FFh, FCh}

VB_Value			
MSB			LSB
FFh	FFh	FFh	FCh

例 2： Decimal 1 = {00h, 00h, 00h, 01h}

VB_Value			
MSB			LSB
00h	00h	00h	01h

応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	6X XXh	失敗

5.3.6.2. 値ブロック読み取り（Read Value Block） [FF B1 …]

このコマンドは、MIFARE 規格に準拠したカードの有効ブロックから 4 バイトの値を読み取るために使用します。このコマンドを呼び出す前に、ユーザーはまず認証を行い、このブロックにへのアクセス権を取得する必要があります。

コマンド

コマンド	CLA	INS	P1	P2	Le
Read Value Block	FFh	B1h	00h	ブロック番号	04h

例 1： Decimal –4 = {FFh, FFh, FFh, FCh}

数値			
MSB			LSB
FFh	FFh	FFh	FCh

例 2： Decimal 1 = {00h, 00h, 00h, 01h}

数値			
MSB			LSB
00h	00h	00h	01h

応答

応答データ	SW1 SW2	意味
4 バイト値（MSB の前）	90 00h	データ成功。
--	6X XXh	失敗

5.3.6.3. 値の減少/増加（Decrement/Increment Value）[FF D7 …]

このコマンドは、ソースブロックから 4 バイトの値を減少/増加させ、結果をターゲットブロックに格納するために使用します（カードは MIFARE 規格に準拠している必要があります）。 結果を同じソースブロックに格納する場合は、ターゲットブロックの番号を 0 またはソースブロック番号に設定できます。このコマンドを呼び出す前に、ユーザーはまず認証を行い、ソースブロックとターゲットブロックにへのアクセス権を取得する必要があります。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Decrement/Increment Value	FFh	D7h	ターゲットブロック#	ソースブロック#	05h	下記の表を確認ください

コマンドデータ

バイト 0	バイト 1	バイト 2	バイト 3	バイト 4
01h	4 バイト追加値（MSB の前）			
02h	4 バイト減少値（MSB の前）			

応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	6X XXh	失敗

5.3.6.4. 値ブロックコピー（Copy Value Block）[FF D7 …]

このコマンドは、ソースブロックから値をターゲットブロックにコピーために使用します（カードは MIFARE 規格に準拠している必要があります）。このコマンドを呼び出す前に、ユーザーはまず認証を行い、ソースブロックとターゲットブロックにへのアクセス権を取得する必要があります。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Copy Value Block	FFh	D7h	00	ソースブロック#	02h	下記の表を確認ください

コマンドデータ

バイト 0	バイト 1
03h	ターゲットブロック#

応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	6X XXh	失敗

5.3.7. PCSC 準拠のタグをアクセスする（ISO 14443-4）

すべての ISO14443-4 に準拠したカード（PICC カード）は、ISO7816-4 の APDU を理解できます。ACR1581U リーダライタが ISO 14443-4 規格に準拠したカードと通信する場合は、ISO 7816-4 で規定された APDU と応答をインタラクティブにするだけです。ACR1581U が内部で ISO14443 の 1 – 4 パートのプロトコルを処理する。

MIFARE Classic （1K/4K）、MIFARE Mini および MIFARE Ultralight タグは T=CL エミュレーションを介してサポートされます。MIFARE タグを標準な ISO 14443-4 タグとして取り扱えばいいです。詳しい情報が **PICC の PCSC 私有（Pseudo）APDU（独自の拡張機能付き）** を参照してください。

ISO 7816-4 APDU フォーマット

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン	Le
ISO 7816 4 パートのコマンド					データの長さ		応答データの予想の長さ

ISO 7816-4 仕様の応答データフォーマット（データ+2 バイト）

応答	データ出力		
結果	応答データ	SW1	SW2

一般的な的な ISO 7816-4 コマンドの応答コード

結果	SW1 SW2	意味
成功	90 00h	操作が成功に完了しました。
エラー	63 00h	操作が失敗しました。

典型的なシーケンスは：

1. タグを提出して、PICC インターファースと接続します。
2. タグ中の情報を読み取り/更新する。

これを実行します：

1. タグと接続する。

   タグの ATR は 3B 88 80 01 00 00 00 00 33 81 81 00 3Ah です。

   その中、

   ATQB アプリケーションのデータ= 00 00 00 00、ATQB プロトコル 情報= 33 81 81。これは ISO 14443-4 Type B タグです。

2. APDU を送信して、乱数を入手する。

   00 84 00 00 08

   >> 1A F7 F3 1B CD 2B A9 58h [90 00h]

注：*ISO 14443-4 Type A* のタグに対して、*APDU"FF CA 01 00 00h"*によって *ATS* を入手できます。

例：

// ISO 14443-4 Type B PICC（ST19XR08E）から８バイトを読み取ります。

APDU = {80 B2 80 00 08h}

CLA = 80h
INS = B2h
P1 = 80h
P2 = 00h
Lc = なし

データ＝なし

Le = 08h

応答：00 01 02 03 04 05 06 07h [$9000h]

5.3.8. サポート PICC ATR

デフォルトでは、次の PICC タイプ/技術がサポートされています。リーダライタにカードをタッチすると、PCSC API 中の SCardStatus()コマンドは次の ATR を CCID ホストにに返す。

カードタイプ/技術	ATR
MIFARE Std 1k[2]	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A
MIFARE Std 4k[2]	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 02 00 00 00 00 69
MIFARE UltraLight[2]	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 03 00 00 00 00 68
MIFARE Ultralight EV1[2]	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 3D 00 00 00 00 56
MIFARE Plus SL1 2k[2]	デフォルト：MIFARE Std 1k と同じ 予備：3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 36 00 00 00 00 5D
MIFARE Plus SL1 4k[2]	デフォルト：MIFARE Std 4k と同じ 予備：3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 37 00 00 00 00 5C
MIFARE Plus SL2 2k	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 38 00 00 00 00 53
MIFARE Plus SL2 4k	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 39 00 00 00 00 52
MIFARE Ultralight C[2]	デフォルト：3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 3A 00 00 00 00 51 予備：MIFARE Ultralight と同じ
SmartMX、模擬 MIFARE Std 1k[2]	デフォルト：MIFARE Std 1k と同じ 予備：ISO14443 -4、Type A 同じ
SmartMX、模擬 MIFARE Std 4k[2]	デフォルト：MIFARE Std 4k と同じ 予備：ISO14443 -4、Type A 同じ
ISO14443 -4、Type A	3B 8n 80 01 T1 ..Tn Tck n = ATS 内履歴バイト数 T1 ..Tn = ATS 内履歴バイト数 Tck = xor 8n 80 01 T1 ..Tn
ISO14443 -4、Type B	3B 88 80 01 T1 ..T8 Tck T1 ..T4 = ATQB 中のアプリケーションデータ T5 ..T7 =ATQB 中のプロトコル情報 T8 = ATA 中の MBLI Tck = xor 88 80 01 T1 ..T8
FeliCa	3B 8F 80 01 80 4F 0C A0 00 00 03 06 11 00 3B 00 00 00 00 42

[2] **ポーリング/ATR オプションの設定（Set Polling/ATR Option）[E0 00 00 23 01 …]** 予備の ATR 定義の構成とキャンセルについては、ダイレクトコマンドの Bit 3 和 Bit 7 を参照してください。

カードタイプ/技術	ATR
ISO15693-3 Generic	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 00 00 00 00 00 63
Infineon My-D Vicinity (SRF55Vxxx)	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 0E 00 00 00 00 6D
ST LRI	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 13 00 00 00 00 70
NXP I-Code SLI	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 14 00 00 00 00 77
NXP I-Code SLIX/SLIX2	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 35 00 00 00 00 56
PicoPass 2K	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 17 00 00 00 00 79
PicoPass 2KS	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 18 00 00 00 00 76
PicoPass 16K	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 19 00 00 00 00 77
PicoPass 16KS	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1A 00 00 00 00 74
PicoPass 16K (8 x 2)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1B 00 00 00 00 75
PicoPass 16KS (8 x 2)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1C 00 00 00 00 72
PicoPass 32KS (16 + 16)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1D 00 00 00 00 73
PicoPass 32KS (16 + 8x2)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1E 00 00 00 00 70
PicoPass 32KS (8x2 + 16)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1F 00 00 00 00 71
PicoPass 32KS (8x2 + 8x2)	ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 20 00 00 00 00 4E

一般的なアプリケーションの応答時間を短縮するために、デフォルトでは次の PICC タイプ/テクノロジーのサポートが無効になっています。ユーザーは、直接コマンド“Set operation Mode”で、さまざまなタイプ/テクノロジーのサポートを有効にすることができます。対応するタイプ/テクノロジーが有効になり、リーダライタにカードをタッチ、PC _ to _ RDR _ IccPowerOn コマンドは次の ATR を CCID ホストにに返します。

カードタイプ/技術	ATR
SRI (SRIX4K/SRT512)	3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 07 00 00 00 00 69
Topaz	3B 8F 80 01 80 4F 0C A0 00 00 03 06 02 00 30 00 00 00 00 5A
PicoPass 2K	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 17 00 00 00 00 75
PicoPass 2KS	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 18 00 00 00 00 7A
PicoPass 16K	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 19 00 00 00 00 7B
PicoPass 16KS	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1A 00 00 00 00 78
PicoPass 16K (8 x 2)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1B 00 00 00 00 79
PicoPass 16KS (8 x 2)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1C 00 00 00 00 7E
PicoPass 32KS (16 + 16)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1D 00 00 00 00 7F
PicoPass 32KS (16 + 8x2)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1E 00 00 00 00 7C
PicoPass 32KS (8x2 + 16)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1F 00 00 00 00 7D
PicoPass 32KS (8x2 + 8x2)	ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 20 00 00 00 00 42
Innovatron	3B 88 80 01 80 4F 05 F0 49 4E 4E 4F 35
CTS	3B 87 80 01 80 4F 04 F0 43 54 53 79

6.0. Escape コマンド

以下のコマンドは、PCD/NFC の構成、およびリーダ/ライタへのアクセスのための特別な機能に使用されます。CCID ホストが PCSC API の SCARD__CTL__CODE（3500）の SCardControl（）でこれらのコマンドをカードリーダーに送信することができる。Escape コマンドを受信すると、リーダライタはコマンドを解読して実行し、レスポンスを生成して CCID ホストに返信します。

注釈：
これらのコマンドは正しいインタフェースを介して送信する必要があります。例えば、*E0 00 00 25 01 00* （*6.1.1*節）は*PICC*インタフェースを介して送信されるべきです（*6.0*節）。

6.1. PICCEscape コマンド

6.1.1. RF 制御（RF Control）[E0 00 00 25 01 ...]

このコマンドは RF 制御を設定するために使われます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
RF Control	E0h	00h	00h	25h	01h	RF 状態

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	RF 状態

RF 状態：1 バイト

RF 状態	説明
00h	RF オフ
01h	RF オン ポーリング
02h	RF オン ポーリングしない

デフォルト設定– *01h (RF オン ポーリング)*

6.1.2. PCD/PICC 状態取得（Get PCD/PICC Status）[E0 00 00 25 00]

このコマンドは PCD/PICC の状態を取る際に使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get PCD/PICC Status	E0h	00h	00h	25h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	Get PCD/PICC Status

PCD/PICC 状態：1 バイト

RF 状態	説明
00h	RF オフ
01h	PICC ない
02h	PICC 準備完了
03h	PICC 選択済み/アクティブ
FFh	エラー

6.1.3. ポーリング/ATR オプションの取得（Get Polling/ATR Option）[E0 00 00 23 00]

このコマンドはポーリングオプションを設定/取得するために使用され、他のコマンドを使用することなく設定を保存できます。このコマンドは、最初のリーダ/ライタ構成にのみ使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get Polling/ATR Option	E0h	00h	00h	23h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	PICC のポーリング/ATR オプション

6.1.4. ポーリング/ATR オプションの設定（Set Polling/ATR Option）[E0 00 00 23 01 …]

このコマンドはポーリングオプションを設置する時に使われます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Set Polling/ATR Option	E0h	00h	00h	23h	01h	PICC のポーリング/ATR オプション

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	PICC のポーリング/ATR オプション

PICC ポーリング/ATR オプション　_1 バイト

操作パラメーター	パラメーター	説明	オプション
Bit 0	ポーリングを有効する	PICCが検出待ちのタブタイプをポーリングします。	1 = 検出 0 = スキップ
Bit 1	RF オフ間隔を有効にする		
Bit 2	RFU		

操作パラメーター	パラメーター	説明	オプション
Bit 3	追加の MIFARE タイプに対する第 3 部分カード ATR の識別を有効にする	PICC が検出待ちのタブタイプをポーリングします。	1 = 検出 0 = スキップ
Bit 4 ~ 5	RF オフ		下記の表を確認ください
Bit 6	RFU		
Bit 7	SmartMX/JCOS カードシミュレーション MIFARE 用のパート 4 ATR を有効化	PICC が検出待ちのタブタイプをポーリングします。	1 = 検出 0 = スキップ

RF オフ間隔–2 Bit　　**ケース 1：**RF オフ無効 （Bit 1 = 0）

操作パラメーター		USB 実行(D0)
Bit 5	Bit 4	
0	0	RF オフない
0	1	
1	0	
1	1	

ケース 2：RF オフ有効 (Bit 1 = 1)

操作パラメーター		USB 実行(D0)
Bit 5	Bit 4	
0	0	250 ms
0	1	500 ms
1	0	1000 ms
1	1	2500 ms

デフォルト設定–*8 Bh*（ポーリングを有効にし、*RF* オフ間隔をを有効にし、第 *3* 部分カードが *ATR* における追加の *MIFARE* タイプの識別を有効にし、*RF* オフ間隔［*00*］、*SmartMX/JCOS* カードシミュレーション *MIFARE* に適用する第 *4* 部分 *ATR* を有効にする）

6.1.5. PICC ポーリングタイプ取得（Get PICCPolling Type）[E0 00 01 20 00]

このコマンドは許可されるテクノロジー/ポーリングタイプを取得するために使用され、他のコマンドを使用することなく設定を保存できます。最初のリーダ/ライタ構成にのみ使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get PICC Polling Type	E0h	00h	01h	20h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン	
						バイト 1	バイト 0
結果	E1h	00h	00h	00h	02h	PICC ポーリングタイプ	

6.1.6. PICC ポーリングタイプ設定（Set PICC Polling Type）[E0 00 01 20 02 …]

このコマンドは PICC ポーリングタイプを設置する時に使われます。

コマンド	CLA	INS	P1	P2	Lc	データ出力	
						バイト 1	バイト 0
Set PICC Polling Type	E0h	00h	01h	20h	02h	PICC ポーリングタイプ	

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン	
						バイト 1	バイト 0
結果	E1h	00h	00h	00h	02h	PICC ポーリングタイプ	

PICC ポーリングタイプ-2 バイト、ビットマスクは次の通りです

バイト	操作パラメーター	パラメーター	説明	オプション
バイト 1	Bit 0	ISO 14443A	PICCが検出待ちのタブタイプをポーリングします。RFUビットは0に設定してください。	1 = 検出 0 = スキップ
	Bit 1	ISO 14443B		
	Bit 2	FeliCa		
	Bit 3	RFU		
	Bit 4	Topaz		
	Bit 5	Innovatron		
	Bit 6	SRI/SRIX		
	Bit 7	RFU		
バイト 0	Bit 0	Picopass (ISO14443B)		
	Bit 1	Picopass (ISO15693)		
	Bit 2	ISO15693		
	Bit 3	CTS		
	Bit 4-7	RFU		

デフォルト設定 – バイト *1*：*07h (ISO14443A、ISO14443B、FeliCa)*

バイト *0*：*05h (Picopass (ISO14443B)、ISO15693)*

例：

コマンド：E0 00 01 20 02 07 05

応答：E1 00 00 00 02 07 05

ポーリングタイプ:　1 バイト = 07h = 0000 0111b = ISO14443A、ISO14443B、FeliCa

バイト 0 = 05h = 0000 0101b = Picopass (ISO14443B)、ISO15693

6.1.7. 自動 PPS 取得（Get Auto PPS ）[E0 00 00 24 00]

PICC が認識されるたびに、リーダーは最大接続速度によっては定義変更れた PCD および PICC との間の通信速度を変更しようとします。カードが提案された接続速度をサポートしていない場合、リーダーはより遅い速度でとカードと接続しようとします。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get Auto PPS	E0h	00h	00h	24h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン	
結果	E1h	00h	00h	00h	02h	最高速度	現在速度

6.1.8. 自動 PPS 設定（Set Auto PPS ）[E0 00 00 24 01…]

このコマンドは自動的な PPS のを設置するために使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Set Auto PPS	E0h	00h	00h	24h	01h	最高速度

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン	
結果	E1h	00h	00h	00h	02h	最高速度	現在速度

PPSの速率

速率	説明
00h	106 kbps;自動PPSが設定されていないと同じです。
01h	212 kbps

速率	説明
02h	424 kbps
03h	848 kbps

デフォルト設定– *02h(424 kbps)*

注釈：

1、通常、アプリケーションが使用中の*PICC*の最大接続速度を知っている必要があります。環境にも達成可能な最大速度に影響します。リーダーは提案されている通信速度をによるして、*PICC*と話をします。*PICC*や環境が提案されている通信速度の要件を満たしていない場合、*PICC*はアクセスできなくなります。

2、高いレート設定がリーダライタの動作に影響を与える場合は、低いレート設定に切り替えます。

6.1.9. PICC タイプ取得（Read PICC Type）[E0 00 00 35 00]

PICC タイプを読み取る時にこのコマンドを使用します。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get PICC Type	E0h	00h	00h	35h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン	
結果	E1h	00h	00h	00h	02h	タイプ	状態

タイプ：1 バイト

タイプ	説明
CCh	PICC ない
04h	Topaz
10h	MIFARE
11h	FeliCa
20h	Type A, Part 4
23h	Type B, Part 4
25h	Innovatron
28h	SRIX
30h	PicoPass
FFh	その他

状態：1 バイト

状態	説明
00h	RF オフ
01h	PICC ない
02h	PICC 準備完了
03h	PICC 選択済み/アクティブ
FFh	エラー

6.1.10. RF パワー設定取得（Get RF Power Setting）[E0 00 00 50 00]

このコマンドは RF のパワー設定を読み取る際に使用される。ファームウェア要件：

- ACR1581U-C FW 1.09.00 及びその以降

コマンド

コマンド	CLA	INS	P1	P2	Le
Get RF Power Setting	E0h	00h	00h	50h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	RF パワー

6.1.11. RF パワー設定（Set RF Power Setting）[E0 00 00 50 01 …]

このコマンドは RF パワーを設置する時に使われる。ファームウェア要件は Get RF Power Setting と同じです。

注釈：Set RF Power Setting を実行すると、リーダライタが一度リセットされる。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Set RF Power Setting	E0h	00h	00h	50h	01h	RF パワー

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	RF パワー

パーセンテージモード

RF パワー -1 バイト

パラメーター	説明
00h	手動RFパワー設定を無効にする
01h	20%
02h	40%
03h	60%
04h	80%
05h	100%

デフォルトに– *00h*

*ハードウェアの制約により、パーセンテージモードの RF 電力値は有効にならない可能性があります。

6.1.12. 選択的保留設定取得（Get Selective Suspend Setting）[E0 00 00 E5 00]

このコマンドは選択保留設定を読み込むために使用される。ファームウェア要件：

- ACR1581U-C　FW 1.09.00 及びその以降

コマンド

コマンド	CLA	INS	P1	P2	Lc
Get Selective Suspend Setting	E0h	00h	00h	E5h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	選択的保留

6.1.13. 選択的保留設定設置（Set Selective Suspend Setting）[E0 00 00 E5 01…]

このコマンドは選択的保留設定を設置するために使用され、ファームウェアに対する要求は「選択的な一時停止設定を取得」コマンドと同じです。

注釈：リーダーがキーボードシミュレーションモードである場合、選択的保留設定機能を有効にすることはできません。その逆も同様です。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Set Selective Suspend Setting	E0h	00h	00h	E5h	01h	選択的な一時停止

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	選択的な一時停止

選択的な一時停止 – 1バイト

パラメーター	説明
00h	無効
01h	有効

デフォルトに– *00h*

6.1.14. PICC– HID キーボードの Escape コマンド

6.1.14.1. 出力フォーマット取得（Get Output Format）[E0 00 00 90 00]

このコマンドは RTC の出力フォーマットを取得する時に使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get Output Format	E0h	00h	00h	90h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン	
結果	E1h	00h	00h	00h	02h	出力フォーマット	出力順番

6.1.14.2. 出力フォーマット設定（Set Output Format）[E0 00 00 90 02…]

このコマンドは出力フォーマットを設定する時に使用されます。

次のファームウェアバージョンでは、02h と 04h の出力順序をサポートできる：

- ACR1581U-C　FW 1.09.00 及びその以降

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力	
Set Output Format	E0h	00h	00h	90h	02h	出力フォーマット	出力順番

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン	
結果	E1h	00h	00h	00h	02h	出力フォーマット	出力順番

出力フォーマット　　1 バイト。

操作パラメーター	パラメーター	説明	オプション
Bit 7 ~ 4	文字の大文字と小文字	PICCが検出待ちのタブタイプをポーリングします。	1 = 検出 0 = スキップ
Bit 3 ~ 0	表示モード		

出力順番　　1 バイト。

状態	説明
00h	デフォルト設定(UIDバイト0、UIDバイト1 … UIDバイトN) 例：aa cc bb dd (原始/実際のUID順番)
01h	カードタイプ順番反転(UIDバイトN、UIDバイトN-1 … UIDバイト0) 例：dd bb cc aa (原始/実際UID順番反転)
02h	ISO 14443とFelicaのみ順番反転 例：dd bb cc aa （選択したカードタイプのオリジナル／実際のUID順序が反転）
04h	ISO 15693のみ順番反転 例：dd bb cc aa （選択したカードタイプのオリジナル／実際のUID順序が反転）

大文字と小文字：高 4 位(Bit 7 から Bit 4 まで)

状態(bit 7 から bit 4 まで)	説明(x bit を注目する必要がない)
1XXX	保留
00x0	小文字
00x1	大文字
000x	4バイトのUIDのみサポート
001x	4、7、8、10バイトのUIDサポート

表示モード：低 4 位(Bit 3 から Bit 0)

状態(bit 7 から bit 4 まで)	説明(x bit を注目する必要がない)
0h	Hex
1h	Dec (バイトごと)
2h	Dec
3h	6H-6H
4h	8H-8H
5h	10H-10H
6h	14H-14H
7h	20H-20H
8h	6H-8D
9h	6H-10D
Ah	8H-10D
Bh	10H-14D
Ch	2H4H-8D
Dh	14H-17D

デフォルト設定– 出力フォーマット：*30h*

出力順番：*00h*

6.1.14.3. UID 開始、中間、終了ビット文字の取得（Get Character at Start, Between, at End UID）[E0 00 00 91 00]

このコマンドは UID 開始、中間、終了のビットを取得する際に使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get Character of UID	E0h	00h	00h	91h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン		
結果	E1h	00h	00h	00h	03h	中間	終わり	スタート

6.1.14.4. UID 開始、中間、終了ビット文字の設定（Set Character at Start, Between, at End UID）[E0 00 00 91 03…]

このコマンドは UID 開始、中間、終了のビットを設定する際に使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力		
Set Character of UID	E0h	00h	00h	91h	03h	中間	終わり	スタート

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン		
結果	E1h	00h	00h	00h	03h	中間	終わり	スタート

中間：1 バイト（各 UID 間のキャラクター）

状態	説明
FFh	中間にはキャラクターがない
その他	汎用シリアルバス（USB）HID使用テーブルを参照

終わり 1 バイト（末尾の文字を出力）

状態	説明
FFh	中間にはキャラクターがない
その他	汎用シリアルバス（USB）HID使用テーブルを参照

終始 1 バイト（開始文字を出力）

状態	説明
FFh	中間にはキャラクターがない
その他	汎用シリアルバス（USB）HID使用テーブルを参照

注釈：

1. *AZERTY*キーボードレイアウトは、零*(0)*とバックスペースキー“ではなく;” “,” “,” “,” “-“ を中間の文字としてサポートされます。

6.1.14.5. キーボードレイアウト言語の取得（Get Keyboard Layout Language）[E0 00 00 92 00]

このコマンドはキーボードレイアウト言語を取得する時に使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get Keyboard Layout Language	E0h	00h	00h	92h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	キーボードレイアウト言語

6.1.14.6. キーボードレイアウト言語の設定（Set Keyboard Layout Language）[E0 00 00 92 01 …]

このコマンドはキーボードレイアウト言語を設定する時に使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Set Keyboard Layout Language	E0h	00h	00h	92h	01h	キーボードレイアウト言語

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	キーボードレイアウト言語

キーボードレイアウト言語：1 バイト

状態	説明
00h	英語
01h	フランス語
02h	保留
03h	リトアニア語

デフォルト設定– *00h(英語)*

6.1.14.7. ホストインターフェース取得 （Get Host Interface） [E0 00 00 93 00]

このコマンドはホストインターフェースを取得する時に使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get Host Interface	E0h	00h	00h	93h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	ホストインターフェース

6.1.14.8. ホストインターフェース設定 （Set Host Interface） [E0 00 00 93 01…]

このコマンドはホストインターフェースを設定する時に使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Set Host Interface	E0h	00h	00h	93h	01h	ホストインターフェース

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	ホストインターフェース

ホストインターフェース:バイト

状態	説明
00h	HIDキーボードのみ適応
01h	CCIDカードリーダーのみ適用
02h	HIDキーボード+CCIDカードリーダー

デフォルト設定 – *01h*（*CCID* リーダーライターのみ）

6.1.14.9. スクリーンキーボード設定取得（Get On-Screen Keyboard Setting）[E0 00 00 94 00]

このコマンドは、スクリーンキーボードの有効状態を取得するために使用される。ファームウェア要件：

- ACR1581U-C　FW 1.09.00 及びその以降

コマンド

コマンド	CLA	INS	P1	P2	Lc
Get On-Screen Keyboard Setting	E0h	00h	00h	94h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	スクリーンキーボード設定

6.1.14.10. スクリーンキーボード設定設置（Set On-Screen Keyboard Setting）[E0 00 00 94 01 …]

このコマンドはスクリーンキーボード設定を設置する時に使われる。ファームウェア要件は Get On-Screen Keyboard Setting とコマンド同じです。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Set On-Screen Keyboard Setting	E0h	00h	00h	94h	01h	スクリーンキーボード設定

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	スクリーンキーボード設定

スクリーンキーボード設定：1 バイト

状態	説明
00h	キーボードのイジェクトを無効にする
01h	キーボードのイジェクトを有効にする

デフォルト設定– *00h*(キーボードのイジェクトを無効する)

6.1.15. PICC-カードシミュレーションの Escape コマンド

6.1.15.1. カードエミュレーションモードに入る（Enter Card Emulation Mode）[E0 00 00 40 03 …]

このコマンドは、NFC フォーラムタイプ 2 タグまたは Felica カードをシミュレートするために、リーダーをカードシミュレーションモードに設定するために使用される。

注釈：*NFC フォーラムタイプ 2 タグをシミュレートする際、Lock バイトはサポートされていません。UID は、ユーザが設定可能です。*

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力		
Enter Card Emulation Mode	E0h	00h	00h	40h	03h	NFC モード	00h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	03h	NFC モード

NFC 設備モード：3 バイト

状態	説明
02h	NFCフォーラムタイプ2ラベルモード
03h	FeliCa
その他	読み取りモード

注意：異なるカードシミュレーションモードに切り替える前に、まずカードの読み取り/書き込みモードに入ってください。カードシミュレーションモードの初期完了後にレスポンスが表示されます。

バイトナンバー	0	1	2	3	USB でバイトアドレスへのアクセス
シリアルナンバー	SN0	SN1	SN2	BCC0	Nil
保留	保留	保留	保留	保留	Nil
内部/ロック	保留	内部	Lock0	Lock1	Nil
データ読み取り/書き込み	Data0	Data1	Data2	Data3	0-3
データ読み取り/書き込み	Data4	Data5	Data6	Data7	4-7
データ読み取り/書き込み	Data8	Data9	Data10	Data11	8-11
データ読み取り/書き込み	Data12	Data13	Data14	Data15	12-15
データ読み取り/書き込み	Data16	Data17	Data18	Data19	16-19
データ読み取り/書き込み	Data20	Data21	Data22	Data23	20-23
データ読み取り/書き込み	Data24	Data25	Data26	Data27	24-27

アクセス可能領域（1988 バイト）

バイトナンバー	0	1	2	3	USB でバイトアドレスへのアクセス
データ読み取り/書き込み	Data28	Data29	Data30	Data31	28-31
データ読み取り/書き込み	Data32	Data33	Data34	Data35	32-35
データ読み取り/書き込み	Data36	Data37	Data38	Data39	36-39
データ読み取り/書き込み	Data40	Data41	Data42	Data43	40-43
データ読み取り/書き込み	Data44	Data45	Data46	Data47	44-47
データ読み取り/書き込み	Data48	Data49	Data50	Data51	48-51
データ読み取り/書き込み	Data52	Data53	Data54	Data55	52-55
データ読み取り/書き込み			…		…
データ読み取り/書き込み	Data1984	Data1985	Data1986	Data1987	1984~1987

表 7：NFC フォーラムタイプ 2 ラベルのメモリマップ（2000 バイト）

メモリ	1 データブロック（16 バイト）	USB でバイトアドレスへのアクセス
データリーダー/ライター	Block 0	0-15
データ読み取り/書き込み	Block 1	16-31
データ読み取り/書き込み	Block 2	32-47
データ読み取り/書き込み	Block 3	48-63
データ読み取り/書き込み	Block 4	64-79
データ読み取り/書き込み	Block 5	80-95
データ読み取り/書き込み	Block 6	96-111
データ読み取り/書き込み	Block 7	112-127
データ読み取り/書き込み	Block 8	128-143
データ読み取り/書き込み	Block 9	144-159

表 8：FeliCa カードのメモリマップ（160 バイト）

その中：

デフォルト： ブロック 0 データ:{10h, 01h, 01h, 00h, 09h, 00h, 00h, 00h, 00h, 00h, 01h, 00h, 00h, 00h, 00h, 1Ch}

デフォルトブロック 0 のデータ NFC タイプ 3 タグの属性情報ブロック

注釈：

1. *FeliCa* カードエミュレーションのサポートは暗号化せずに読み取り/書き込み。
2. *FeliCa* カード識別番号（*IDm*）はユーザに定義可能で、メーカーコードが（*0388*）に固定されています。

6.1.15.2. カードエミュレーションのデータを読み取る（Read Card Emulation Data）（NFC フォーラムタイプ 2 ラベル）[E0 00 00 60 04 …]

このコマンドはカードエミュレーションカードのデータを読み取るために使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン			
Read Card Emulation Data	E0h	00h	00h	60h	04h	00h	NFC モード	終始オフセット	長さ

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	長さ	データ

終始オフセット： 1 バイト – **表 7** 中 Data0 からのアドレス

長さ： 1 バイト –バイト数量

6.1.15.3. カードエミュレーションのデータ書き込む（Write Card Emulation Data）（NFC フォーラムタイプ 2 ラベル）[E0 00 00 60 …]

このコマンドは、エミュレートされたカードへの書き込みに使用されています。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン				
Write Card Emulation Data	E0h	00h	00h	60h	長さ＋04h	01h	NFC モード	終始オフセット	長さ	データ

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン		
結果	E1h	00h	00h	00h	03h	長さ	90h	00h

NFC 設備モード：1 バイト

状態	説明
02h	NFCフォーラムタイプ2ラベルモード
03h	FeliCa
その他	読み取りモード

終始オフセット： 1 バイト – **表 7** 中 Data0 からのアドレス

長さ： 1 バイト –バイト数量

6.1.15.4. カードエミュレーションのデータを読み取る（Read Card Emulation Data）（NFC フォーラムタイプ 2 ラベル）（拡張）

このコマンドはカードエミュレーションカードのデータを読み取るために使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc		コマンドデータイン			
Read Card Emulation Data	E0h	00h	01h	60h	05h	00h	NFC モード	終始オフセット Bit [15:8]	終始オフセット Bit [7:0]	長さ

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	長さ	データ

終始オフセット： 2 バイト – **表 7** 中 SN 0 からアドレスを読み取る

長さ： 1 バイトー読み取り待ちのバイト

6.1.15.5. カードエミュレーションのデータ書き込む（Write Card Emulation Data）（NFC フォーラムタイプ 2 ラベル）（拡張）

このコマンドは、エミュレートされたカードへの書き込みに使用されています。

コマンド

コマンド	CLA	INS	P1	P2	Lc		コマンドデータイン				
Write Card Emulation Data	E0h	00h	01h	60h	長さ＋ 05h	01h	NFC モード	終始オフセット Bit [15:8]	終始オフセット Bit [7:0]	長さ	データ

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン		
結果	E1h	00h	00h	00h	03h	長さ	90h	00h

NFC 設備モード：1 バイト

状態	説明
02h	NFCフォーラムタイプ2ラベルモード
その他	読み取りモード

終始オフセット： 2 バイト – **表 7** 中 SN 0 からアドレスを書き込む

長さ： 1 バイトー書き込み待ちのバイト

6.1.15.6. NFC フォーラムタイプ 2 ラベル模擬 ID を設定（Set Card Emulation of NFC Forum Type 2 Tag ID）[E0 00 00 61 03 …]

このコマンドは、シミュレートされる NFC フォーラムタイプ 2 タグの UID を設定するために使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Set Card Emulation Lock Data	E0h	00h	00h	61h	03h	3 バイトの UID

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン	
結果	E1h	00h	00h	00h	02h	90h	00h

6.1.15.7. NFC カードエミュレーションロックデータロックを設定する（Set Card Emulation Lock Data in NFC）[E0 00 00 65 01…]

このコマンドは NFC 通信中、カードエミュレーションのデータをロックするために使用されます。データがロックされると、NFC で書き換えることができません。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Set Card Emulation Lock Data	E0h	00h	00h	65h	01h	ロック

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	ロック

ロック：1 バイトーNFC 通信で書き換えされないように、データを保護します。

操作パラメーター	パラメーター	説明	オプション
Bit 7 ~ 2	保留	保留	
Bit 1	Felica ロックを有効にする	データは NFC 通信で書き換えられません。USB 直接コマンドでデータを変更できます。	0：ロックを無効にする 1：ロックを有効にする
Bit 0	NFC フォーラムタイプ 2 タブを有効する		

6.1.15.8. カードエミュレーションの FeliCa の IDM を設定する（Set Card Emulation FeliCa IDm）[E0 00 00 64 06 …]

このコマンドはカードエミュレーションの FeliCa カード上で、6 バイトの FeliCa カードフラグを設定するために使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Set Card Emulation FeliCa IDm	E0h	00h	00h	64h	06h	IDm

応答コード

応答	CLA	INS	P1	P2	Le	データ出力
結果	E1h	00h	00h	00h	06h	IDm

その中：

IDm 6 バイト

6.1.15.9. カードエミュレーション状態取得（Get Card Emulation Status）[E0 00 00 69 00]

このコマンドは NFC 通信中、カードエミュレーションのデータを取得するために使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc
Get Card Emulation Status	E0h	00h	00h	69h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	状態

状態：1 バイト

操作パラメーター	モード	説明
Bit 7 ~ 6	保留	保留
Bit 5	模擬カードがアクティブ化された	1 = アクティブ化される
Bit 4	模擬カードが外れた	1 = カードが外れた
Bit 3	模擬カードが全部読み取りました	1 = すべてのデータが読み取りました
Bit 2	模擬カードが読み取りました	1 = データが読み取りました
Bit 1	模擬カード書き込まられた	1 = データ書き込まられた
Bit 0	模擬カードが検出された	1 = カード検出

6.1.15.10. シミュレーション NFC フォーラムタイプ 2 ラベルモードのコマンドサンプルセット

このコマンドセットは、ACR1581U の NFC フォーラムタイプ 2 ラベルモードをシミュレートすることで、ACS サイトをトリガする https://www.acs.com.hk。ステップ：

1. 次のコマンドを使用してカードシミュレーションモードに入ります。
   - カードエミュレーションモードに入る（Enter Card Emulation Mode）
     E0 00 00 40 03 02 00 00
2. 下記のコマンドで NDEF データを含めています。データ書きます：
   - カードシミュレーションデータを書き込むカードエミュレーションのデータ書き込む（Write Card Emulation Data）（NFC フォーラムタイプ 2 ラベル）
     E0 00 00 60 1A 01 02 00 18 E1 10 F4 00 03 0F D1 01 0B 55 02 61 63 73 2E 63 6F 6D 2E 68 6B FE

このコマンドセットはサンプルの長い URL サイトを起動します

https://www.example.com/this/is/a/very/long/url/that/keeps/going/on/and/on/with/even/more/segments/added/to/make/sure/it/exceeds/the/typical/length/limit/of/260/bytes/which/is/surprisingly/easy/to/do/if/you/keep/adding/more/and/more/segments/like/this/one/and/even/more

ACR1581U で NFC フォーラムタイプ 2 ラベルモードをシミュレーションする。ステップ：

1. 次のコマンドを使用してカードシミュレーションモードに入ります。
    - カードエミュレーションモードに入る（Enter Card Emulation Mode）コマンドを送信する
      E0 00 00 40 03 02 00 00

2. 下記のコマンドで NDEF データを含めています。データ書きます：

   - カードエミュレーションのデータ書き込む（Write Card Emulation Data）（NFC フォーラムタイプ 2 ラベル）コマンドを送信する NDEF メッセージは 256 バイトを超えるため、2 つの部分に分けて NFC フォーラムタイプ 2 タグに送信する必要があります。

     E0 00 00 60 AC 01 02 00 A8 E1 10 F4 00 03 FF 01 09 C1 01 00 00 01 02 55 02 65 78 61 6D 70 6C 65 2E 63 6F 6D 2F 74 68 69 73 2F 69 73 2F 61 2F 76 65 72 79 2F 6C 6F 6E 67 2F 75 72 6C 2F 74 68 61 74 2F 6B 65 65 70 73 2F 67 6F 69 6E 67 2F 6F 6E 2F 61 6E 64 2F 6F 6E 2F 77 69 74 68 2F 65 76 65 6E 2F 6D 6F 72 65 2F 73 65 67 6D 65 6E 74 73 2F 61 64 64 65 64 2F 74 6F 2F 6D 61 6B 65 2F 73 75 72 65 2F 69 74 2F 65 78 63 65 65 64 73 2F 74 68 65 2F 74 79 70 69 63 61 6C 2F 6C 65 6E 67 74 68 2F 6C 69 6D 69 74 2F 6F 66 2F 32 36 30 2F 62 79 74

     E0 00 00 60 6E 01 02 A8 6A 65 73 2F 77 68 69 63 68 2F 69 73 2F 73 75 72 70 72 69 73 69 6E 67 6C 79 2F 65 61 73 79 2F 74 6F 2F 64 6F 2F 69 66 2F 79 6F 75 2F 6B 65 65 70 2F 61 64 64 69 6E 67 2F 6D 6F 72 65 2F 61 6E 64 2F 6D 6F 72 65 2F 73 65 67 6D 65 6E 74 73 2F 6C 69 6B 65 2F 74 68 69 73 2F 6F 6E 65 2F 61 6E 64 2F 65 76 65 6E 2F 6D 6F 72 65 FE

注釈：

NDEF（*NFC* データ相互フォーマット）に関する詳細な情報と規定を了解する必要がある場合は、*NDEF* 仕様を参照することをお勧めします。この仕様では、*NDEF* レコードの構造と使用について包全面的なガイドラインと詳細情報を提供し、これらの *NDEF* レコードは *NFC* データのインタラクションによく使用されています。*NDEF* 仕様は、*ACR1581U* デバイス環境における *NDEF* コマンドとデータの解読と利用方法を深く理解するのに役立ちます。

6.1.16. PICC-検出モードの Escape コマンド

6.1.16.1. 検出モードに入る（Enter Discovery Mode）[E0 00 00 6A 01 …]

このコマンドは検出モードに入る時に使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Enter Discovery Mode	E0h	00h	00h	6Ah	01h	検出モード

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	検出モード

検出モード：1 バイト

状態	説明
00h	カードリーダーモード

状態	説明
02h	NFCフォーラムタイプ2ラベルモード
03h	FeliCa

6.2. ICCEscape コマンド

6.2.1. 専用モード取得（Get Exclusive Mode ）[E0 00 00 2B 00]

このコマンドはリーダーの専用モードの設定を取得する時に使われる。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get Exclusive Mode	E0h	00h	00h	2Bh	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	専用モード

6.2.2. 専用モード設定（Set Exclusive Mode ）[E0 00 00 2B01 …]

このコマンドはリーダーの専用モードを有効/無効するのを設定する時に使われる。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Set Exclusive Mode	E0h	00h	00h	2Bh	01h	専用モード

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	専用モード

専用モード(1 バイト)

専用モード	説明
00h	共有モード：ICC と PICC インタフェースは同時に動作することができる。
01h	専用モード：ICC カード挿入時オートポーリング機能とアンテナが自動にするオされ、PICC が無効になる（デフォルト）
その他	RFU

デフォルト設定—*01h*（専用モード）

6.2.3. カード電源設置取得（Get Card Power Config）[E0 00 00 0B 00]

このコマンドは、ICC カードの電源構成を取得するために使用され、最初のリーダ/ライタ構成のみに使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get Card Power Config	E0h	00h	00h	0Bh	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	カード電源配置

6.2.4. カード電源配置設定（Set Card Power Config）[E0 00 00 0B 01…]

このコマンドはと ICC カードの電源配置を設定する時に使われます。最初のリーダ/ライタ構成にのみ使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Set Card Power Config	E0h	00h	00h	0Bh	01h	配置

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	カード電源配置

カードの電源配置（1 バイト）

カード電源配置	説明
00h	自動検出、1.8V -> 3V -> 5V
01h	5 V のみ
02h	3 V のみ
03h	1.8 V のみ
04h	自動検出、5V -> 3V -> 1.8V
その他	RFU

デフォルト設定 – *04h (自動検出、5V -> 3V -> 1.8V)*

6.3. 周辺設備制御と他の Escape コマンド

6.3.1. ファームウェアバージョン取得（Get Firmware Version）[E0 00 00 18 ...]

このコマンドはファームウェアのバージョンを入手する時に使われます。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get Firmware Version	E0h	00h	00h	18h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	ファームウェアのバージョンの長さ	ファームウェアのバージョン

例：
コマンド：　　E0 00 00 18 00
応答コード：　E1 00 00 00 12 41 43 52 31 35 38 31 20 46 57 20 31 2E 30 30
十六進ファームウェアのバージョン：　41 43 52 31 35 38 31 20 46 57 20 31 2E 30 30
ASCII ファームウェアバージョン：ACR1581 FW 1.00

6.3.2. シリアルナンバー取得（Get Serial Number ）[E0 00 00 33 00]

シリアルナンバーを取得する時にこのコマンドを使用します。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get Serial Number	E0h	00h	00h	33h	00h

応答コード

応答	CLA	INS	P1	P2	Le	データ出力
結果	E1h	00h	00h	00h	シリアルナンバーの長さ	シリアルナンバー

6.3.3. USB 記述子内の S/N を設定する（Set S/N in USB Descriptor）[E0 00 00 F0]

このコマンドは USB 記述子内の S/N を設定するために使われます。

コマンド

コマンド	CLA	INS	P1	P2	Le	コマンドデータイン	
Set S/N in USB Descriptor	E0h	00h	00h	F0h	02h	00h	USB 記述子内の SN を有効する

応答コード

応答	CLA	INS	P1	P2	Le	データ出力		
結果	E1h	00h	00h	00h	03h	USB 記述子内の SN を有効する	90h	00h

USB 記述子内の SN を有効する（1 バイト）

USB 記述子内の SN を有効する	説明
00h	USB 記述子内の SN を無効する
01h	USB 記述子内の SN を有効する

FW1.09.00 以降のバージョンのデフォルト設定–*00h*

FW1.09.00 以降のバージョンのデフォルト設定–*01h*

6.3.4. ブザー制御の設定-単発（Set Buzzer Control - Single Time）[E0 00 00 28 01 …]

このコマンドは単発のブザーを設定するために使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Buzzer Control	E0h	00h	00h	28h	01h	ブザー状態

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	ブザー状態

ブザーの状態(1 バイト)

ブザー状態	説明
00h	OFF

ブザー状態	説明
01 ~ FFh	オン、持続時間は 10 ms を単位にする

6.3.5. ブザー制御の設定-重複（Set Buzzer Control - Repeatable）[E0 00 00 28 03 …]

このコマンドはブザーの周期を設定するために使用されます周期

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Buzzer Control	E0h	00h	00h	28h	03h	ブザー状態

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	03h	ブザー状態

ブザーの状態(3 バイト)

操作パラメーター	ブザー状態	説明
パラメーター1 – バイト 0	オン時間帯	01 ~ FF: オンにする持続時間は 10 ms を単位にする
パラメーター2 – バイト 1	OFF 時間帯	01 ~ FF: OFF にする持続時間は 10 ms を単位にする
パラメーター3 – バイト 2	重複時間	01 ~ FF: 繰り返し回数

6.3.6. LED 状態取得（Get LED Status）[E0 00 00 29 00]

このコマンドは LED の状態を取得するために使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get LED Status	E0h	00h	00h	29h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	LED 状態

6.3.7. LED 制御設定（Set LED Control）[E0 00 00 29 01 …]

このコマンドは LED 制御を設定するために使われます

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Set LED Control	E0h	00h	00h	29h	01h	LED 状態

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	LED 状態

LED 状態（1 バイト）

LED 状態	説明
Bit 0 = 青色 LED	1 = ON ; 0 = OFF
Bit 1 : 緑の LED	1 = ON ; 0 = OFF
Bit 2-7 ： RFU	その他

6.3.8. UI 操作取得（Get UI Behaviour）[E0 00 00 21 00]

このコマンド PCD UI の操作を取得するために使用され、他のコマンドを使用することなく設定を保存できます。最初のリーダ/ライタ構成にのみ使用されます。

コマンド

コマンド	CLA	INS	P1	P2	Le
Get PICC UI Behaviour	E0h	00h	00h	21h	00h

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	PICC UI 操作

6.3.9. UI 操作設定（Set UI Behaviour）[E0 00 00 21 01 …]

このコマンドは PICC　UI 操作のパラメ――を設定するために使われます。

UI 操作 bit 1、bit 2、bit 5 は、次のファームウェアバージョンで使用できる：

- ACR1581U-C　FW 1.09.00 及びその以降

コマンド

コマンド	CLA	INS	P1	P2	Lc	データ出力
Set PICC UI Behaviour	E0h	00h	00h	21h	01h	PICC UI 操作

応答コード

応答	CLA	INS	P1	P2	Le	コマンドデータイン
結果	E1h	00h	00h	00h	01h	PICC UI 操作

UI 操作-1 バイト、ビットマスクは次の通りです

操作パラメーター	パラメーター	説明	オプション
Bit 0	動作中（LED が速く点滅する）	リーダーのUI操作	1 = 有効にする 0 = 無効にする
Bit 1	PICCのポーリングステータスLED		
Bit 2	PICC 活性化状態のLED		
Bit 3	カードがアンテナ領域に入るイベント（ブザーが一時的に鳴る）		
Bit 4	カードがアンテナ領域から外すイベント（ブザーが一時的に鳴る）		
Bit 5	電源投入イベント（ブザーが一時的に鳴る）		

PICC デフォルト設定– *2Fh*

注釈：

1. *UI*操作コマンドの取得／設定は*SAM*インタフェースはに適用しない。

附录 A. NDEF メッセージ

このセクションでは、NDEF メッセージを使用して Ntag に URL をエンコードする方法について説明します。

このコマンドのデータフォーマットを了解したい場合、“NFC Forum NFC Data Exchange Format (NDEF) Specifications 1.0”を参照してください。

例：

NDEF メッセージ = { D1 01 0B 55 02 61 63 73 2E 63 6F 6D 2E 68 6Bh}

オフセット	コンテンツ	長さ	説明
0	D1	1	NDEF データヘッダ TNF = 01h、SR=1、MB=1、ME=1
1	01	1	レコード名の長さ （1 バイト）
2	0B	1	URI ペイロードの長さ（11 バイト）
3	55 （“U”）	1	レコードタイプ“U”
4	02	1	省略：“https://www.”
5	61 63 73 2E 63 6F 6D 2E 68 6B	10	URL 自体。“acs.com.hk”

Ntag にエンコード = {03 0F D1 01 0B 55 01 61 63 73 2E 63 6F 6D 2E 68 6B FEh}

オフセット	コンテンツ	長さ	説明
0	03	1	TLV データヘッダ 03h = NDEF メッセージ
1	0F	1	NDEF メッセージの長さ（15 バイト）
2	D1 01 0B 55 01 61 63 73 2E 63 6F 6D 2E 68 6B	15	NDEF メッセージ
17	FE	1	TLV データヘッダ FEh = 終了記録

附录 B. ACR1281U との互換性

この節では、ACR1281Uとに対応する各種コマンドについて説明する。ファームウェアには 1.08.01 以上のバージョンが必要です。

附录 B.1. 認証キーのダンロード（Load Authentication Keys）

このコマンドはリーダーにキーをロードする時に使われる。このキーは MIFARE 1K/4K メモリカードの特定なセクターを認証するために使用されます。リーダーは二種の認証キーのアドレスが提供られている：失いやすいキーのアドレスと失いやすくないキーのアドレス。

コマンド

コマンド	CLA	INS	P1	P2	Le	コマンドデータイン
Load Authentication Keys	FFh	82h	キー構造	キーナンバー	06h	キー

その中：

キー構造 (1 バイト)

00h = キーが失いやすいキーのメモリにロードされる。

20h = キーが失いにくいキーのメモリにロードされる。

その他＝予約済み

キーナンバー (1 バイト)

00h – 1Fh = キーを保存するための失いやすくないキーのメモリ。キーは永遠にリーダーに保存されて、リーダが PC から切断されても消えません。その失いにくいキーのメモリには最大３２個キーをストアーできる。

20h（セッション鍵） = 臨時のキーを格納するための失いやすいキーのメモリ。**リーダーが PC から切断されると、キーはメモリから消去される。**失いやすいキーのメモリは一つしかありません。これらは異なるセッションにセッション鍵として使用できる。デフォルト値 = FF FF FF FF FF FFh

キー (6 バイト)

リーダーにロードされるきーの値。

例：{FF FF FF FF FF FFh}

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了）

= 63 00h （操作が失敗しました）。

附录 B.2. MIFARE Classic ((1K/4K)) カードに対しての認証（Authentication for MIFARE Classic ((1K/4K)）

このコマンドはリーダーにストアーられているキーで MIFARE 1K/4K カード（PICC）を認証する時に使われます。二種の認証キーを使われる：TYPE_A と TYPE_B。

コマンド

コマンド	CLA	INS	P1	P2	P3	コマンドデータイン
Authentication 6 バイト (廃棄)	FFh	88h	00h	ブロック番号	キーのタイプ	キーナンバー

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Authentication 10 バイト	FFh	86h	00h	00h	05h	データバイト認証

認証データバイト（（5 バイト））

バイト 1	バイト 2	バイト 3	バイト 4	バイト 5
バージョン 01h	00h	ブロック番号	キーのタイプ	キーナンバー

その中：

ブロック番号 （1 バイト）

認証されていないメモリブロック。

注：一枚の *MIFARE 1K* カードが *16* 個セクターに分けて、各セクターには *4* 個の連続的なブロックが含めている。例：セクター*00h* が含めているブロック*{00h*、*01h*、*02h* および *03h}*；セクター*01h* が含めているブロック*{04h*、*05h*、*06h* および *07h}*；ラストセクター*0Fh* が含めているブロック*{3Ch*、*3Dh*、*3Eh* および *3Fh}*。

当ブロックが成功に認証されると、同じセクターの残りのブロックが認証される必要がありません。詳しい情報は *MIFARE 1K/4K* 基準を参照してください。

キータイプ （1 バイト）

60h = Key A キーとして、認証用に使われる。

61h = Key B キーとして、認証用に使われる。

キーナンバー (1 バイト)

00h – 1Fh = キーを保存するための失いやすくないキーのメモリ。キーは永遠にリーダーに保存されて、リーダが PC から切断されても消えません。その失いにくいキーのメモリには最大３２個キーをストアーできる。

20h（セッション鍵）= 臨時のキーを格納するための失いやすいキーのメモリ。**リーダーが PC から切断されると、キーはメモリから消去されます。**失いやすいキーのメモリは一つしかありません。これらは異なるセッションにセッション鍵として使用できる。デフォルト値 = FF FF FF FF FF FFh

応答

応答	データ出力	
結果	SW1	SW2

その中：

SW1 SW2 = 90 00h （操作が成功に完了）

= 63 00h （操作が失敗しました）。

附录 B.3. 手動 PICC ポーリング（Manual PICC Polling）

このコマンドは、リーダ/ライタの誘導範囲内に PICC が存在するかどうかを検出するために使用される。自動 PICC ポーリング機能が無効にされる時、このコマンドを使用します。

Manual PICC Polling コマンドフォーマット（6 バイト）

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Manual PICC Polling	E0h	00h	00h	22h	01h	0Ah

Manual PICC Polling 応答フォーマット（6 バイト）

応答	CLA	INS	P1	P2	Le	データ出力
結果	E1h	00h	00h	00h	01h	状態

その中：

状態 （1 バイト）

00h = PICC 検出された

FFh = PICC 検出されていない

附录 B.4. PICC 操作パラメーターを読取る（Read PICC Operating Parameter）

このコマンドは PICC 操作のパラメーターを検査するために使用されます。

Read PICC Operating Parameter コマンドフォーマット(5 バイト)

コマンド	CLA	INS	P1	P2	Lc
Read PICC Operating Parameter	E0h	00h	00h	20h	00h

Read PICC Operating Parameter 応答フォーマット（6 バイト）

応答	CLA	INS	P1	P2	Le	データ出力
結果	E1h	00h	00h	00h	01h	操作パラメーター

附录 B.5. PICC 操作パラメーター設定（Set PICC Operating Parameter）

このコマンドは PICC 操作のパラメーターを設定するために使われます。

Set PICC Operating Parameter コマンドフォーマット（6 バイト）

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン
Set PICC Operating Parameter	E0h	00h	00h	20h	01h	操作パラメーター

操作パラメーター　（1 バイト）

操作パラメーター	パラメーター	説明	オプション
Bit 0	ISO 14443 A タイプ	PICC ポーリング中に検出する必要なタグのタイプ	1 = 検出 0 = スキップ
Bit 1	ISO 14443 B タイプ		1 = 検出 0 = スキップ
Bit 2 – 7	RFU	RFU	RFU

注意：操作のデフォルト値 = *03Fh*。

Set PICC Operating Parameter 応答フォーマット（6 バイト）

応答	CLA	INS	P1	P2	Le	データ出力
結果	E1h	00h	00h	00h	01h	操作パラメ―タ―

附录 B.6. カード挿入カウンタ初期化（Initialize Cards Insertion Counter）

このコマンドは、カードの挿入/検出カウンタを初期化するために使用されます。

Initialize Card Insertion Counter コマンドフォーマット（9 バイト）

コマンド	CLA	INS	P1	P2	Lc	コマンドデータイン			
Initialize Card Insertion Counter	E0h	00h	00h	09h	04h	ICC Cnt (LSB)	ICC Cnt (MSB)	PICC Cnt (LSB)	PICC Cnt (MSB)

その中：

ICC Cnt (LSB) (1 バイト)

ICC 挿入カウンタ（LSB)

ICC Cnt (MSB) (1 バイト)

ICC 挿入カウンタ(MSB)

PICC Cnt (LSB) (1 バイト)

PICC 挿入カウンタ(LSB)

PICC Cnt (MSB) (1 バイト)

PICC 挿入カウンタ(MSB)

Initialize Card Insertion Counter コマンドの応答フォーマット（5 バイト）

応答	CLA	INS	P1	P2	Le
結果	E1h	00h	00h	00h	00h

附录 B.7. カード挿入カウンタを読み取る（Read Cards Insertion Counter）

このコマンドは、カードの挿入/検出カウンタを読み取るために使用される。

Read Card Insertion Counter コマンドフォーマット（5 バイト）

コマンド	CLA	INS	P1	P2	Lc
Read Card Insertion Counter	E0h	00h	00h	09h	00h

Read Card Insertion Counter 応答フォーマット（9 バイト）

応答	CLA	INS	P1	P2	Le	データ出力			
結果	E1h	00h	00h	00h	04h	ICC Cnt (LSB)	ICC Cnt (MSB)	PICC Cnt (LSB)	PICC Cnt (MSB)

その中：

ICC Cnt (LSB) (1 バイト)

ICC 挿入カウンタ（LSB)

ICC Cnt (MSB) (1 バイト)

ICC 挿入カウンタ(MSB)

PICC Cnt (LSB) (1 バイト)

PICC 挿入カウンタ(LSB)

PICC Cnt (MSB) (1 バイト)

PICC 挿入カウンタ(MSB)

附录 B.8. カード挿入カウンタ更新（Update CardInsertion Counter）

このコマンドは、カードの挿入/検出カウンタを更新するために使用される。

Update Card Insertion Counter コマンドフォーマット（5 バイト）

コマンド	CLA	INS	P1	P2	Lc
Update Card Insertion Counter	E0h	00h	00h	0Ah	00h

Update Card Insertion Counter コマンドの応答フォーマット（9 バイト）

応答	CLA	INS	P1	P2	Le	データ出力			
結果	E1h	00h	00h	00h	04h	ICC Cnt (LSB)	ICC Cnt (MSB)	PICC Cnt (LSB)	PICC Cnt (MSB)

その中：

ICC Cnt (LSB) (1 バイト)

ICC 挿入カウンタ（LSB)

ICC Cnt (MSB) (1 バイト)

ICC 挿入カウンタ(MSB)

PICC Cnt (LSB) (1 バイト)

PICC 挿入カウンタ(LSB)

PICC Cnt (MSB) (1 バイト)

PICC 挿入カウンタ(MSB)

附录 B.9. 互換性と移行説明

製品の最適化アップグレードの過程で、ACR1281 がサポートするいくつかのコマンドは、ACR1581 で更新または置換された。

次の表は、2 つのモデル間のスムーズな移行を支援するための明確な対応関係を示している。

LED 制御、PICC ポーリングなどの最も一般的な機能は依然として正常に使用されていますが、いくつかのコマンドは新しい名前や構成構造が使用されている。

移行中や統合中に何か問題が発生した場合は、info@acs.com.hk までにメールを送信してください。弊社の技術チームは、喜んでお客様にサポートを提供致します。

注釈：

ACR1281U が備えるカード挿入カウンタ機能は、ACR1581U ではサポートされていない。

この機能に依存するアプリケーションは、適切な変更が必要です。

ACR1281U API	ACR1281U 指令	ACR1581U サポート可否	ACR1581U 対応指令	説明
LED 制御	E0 00 00 29 01	YES	**LED 制御設定（Set LED Control）[E0 00 00 29 01 ...]**	意味は同じですが、LED の色は異なる（赤/緑ではなく、青/緑です）
LED 状態	E0 00 00 29 00	YES	**LED 状態取得（Get LED Status）[E0 00 00 29 00]**	上と同じ

ACR1281U API	ACR1281U 指令	ACR1581U サポート可否	ACR1581U 対応指令	説明
デフォルト LED とブザー操作を設定	E0 00 00 21 01	YES	**UI 操作設定（Set UI Behaviour）[E0 00 00 21 01 …]**	ビット定義が変更され、排他モードブザーが削除され、電源投入ブザー制御が追加された
デフォルト LED とブザー操作の読み込み	E0 00 00 21 00	YES	**UI 操作取得（Get UI Behaviour）[E0 00 00 21 00]**	上と同じ
自動 PICC ポーリング設定	E0 00 00 23 01	YES	**ポーリング/ATR オプションの設定（Set Polling/ATR Option）[E0 00 00 23 01** …]	ビット定義が変更され、PICC が動作していないときにアンテナを閉じる設定が削除され、ATR で第 3 部分のカードに追加の MIFARE タイプ識別が追加されました
自動 PICC ポーリング読取る	E0 00 00 23 00	YES	**ポーリング/ATR オプションの取得（Get Polling/ATR Option）[E0 00 00 23 00]**	上と同じ