Advanced Card Systems Ltd.
Card & Reader Technologies

ACR3901U-S1
ACS セキュア Bluetooth®
インテリジェント接触リーダー

リファレンスマニュアル　V1.10

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

バージョン履歴

发布日期	修订说明	版本号
2015-07-10	• 初回発布	1.00
2015-09-17	• 商品マーケティング名更新 • 書式設定更新 • セクション 5.4.2 更新：LED ステータス • セクション 6.2 更新：プロフィール選択 • セクション 6.5.5.5 更新：Sleepモード選択 • セクション 6.5.6 更新：顧客マスターキーリセット	1.01
2015-11-05	• セクション 5.4.2 更新：LED ステータス • セクション 6.2 更新：プロフィール選択 • セクション 6.3 更新：認証 • セクション 6.5.6 更新：顧客マスターキーリセット • セクション 6.5.5.4 更新：マスターキーコマンドの書き換え • セクション 6.6 更新：相互認証表	1.02
2016-09-16	• 更新の製品写真 • 商品マーケティング名更新 • 誤ったチェックサムコマンドの例を更新	1.03
2017-01-11	• 電池寿命更新 • セクション 6.5.5.7 追加：Tx パワーの設定 • セクション 6.5.5.8 追加：Tx パワーの読み取り • セクション 6.5.7 追加：Bluetooth mode でカードの設定パラメータ	1.04
2017-10-12	• セクション 6.5.5.1 からセクション 6.5.6（ダイレクトコマンド）までをセクション 8.1 に変更する • セクション 6.3 更新：認証 • セクション 6.6 更新：相互認証 • セクション 6.6.1 更新：SPH_to_RDR_ReqAuth • セクション 6.6.3 更新：SPH_to_RDR_AuthRsp • セクション 6.6.4 更新：RDR_to_SPH_AuthRsp2 • セクション 6.6.5 更新：SPH_to_RDR_DataReq • セクション 8.1.9 更新：顧客マスターキーリセットリクエスト	1.05

发布日期	修订说明	版本号
2017-12-28	• アップデート 6.3：認証 • アップデート 6.4.2：相互認証後の Bluetooth フレームフォーマット • アップデート 6.6.1：SPH_to_RDR_ReqAuth • アップデート 6.6.3：SPH_to_RDR_AuthRsp • セクション 6.6.5 を更新する：SPH_to_RDR_DataReq • アップデート 8.1.4：マスターキーコマンドのリセット • 更新セクション 8.1.9：顧客マスタキーリセット要求	1.06
2018-07-27	• アップデート 6.3：認証 • アップデート 6.5：Bluetooth 通信プロトコル • 第 6.5.5 項を追加：APDU2 コマンド（FW v1.20 以降） • セクション 7.1.4：PC_to_RDR_XfrBlock の更新 • アップデート 7.2.1：RDR_to_PC_DataBlock	1.07
2018-12-17	• アップデート 6.0：ハードウェアのデザイン • アップデート 7.0： ホストプログラミング API • 除去セクション 10.0：PC_to_RDR_XfrBlock を介して、他のコマンドを実行する • 削除された付録 A：サポートしているカードのタイプ	1.08
2018-12-27	• アップデート 6.1.6.2：RDR_to_SPH_AuthRsp1 • アップデート 6.1.6.4：RDR_to_SPH_AuthRsp2 • アップデート 6.1.6.5：SPH_to_RDR_DataReq • アップデート 6.1.6.6：RDR_to_SPH_DataRsp • アップデート 6.2.1.1：PC_to_RDR_IccPowerOn • セクション 6.2.1.8 追加：PC_to_RDR_Escape • セクション 6.2.2.4 追加：RDR_to_PC_Escape	1.09
2019-06-13	• 商品マーケティング名更新	1.10

目次

図示一覧表

テーブル一覧表

1.0. 紹介

ACR3901U-S1 ACS セキュア Bluetooth®スマートカードリーダーは、PC/モバイルデバイスとスマートカードリーダー間の通信インターフェースです。異なるタイプのスマートカードは異なるコマンドと通信プロトコルを採用しているため、ほとんどの場合、スマートカードと PC/モバイルデバイスの間に直接通信できません。ACR3901U-S1 ACS セキュア Bluetooth®カードリーダーが様々のカードに PC/モバイルデバイスから統一されたインタフェースを確立します。これはカードの様々の特性と互換性がありますので、ソフトウェアプログラマがスマートカード操作の技術的な詳細を心配する必要はありません。多くの場合、これらの詳細はスマートカードシステムの実装に関連していません。

1.1. 参照ファイル

下記のファイルは www.usb.org でダウンロードできます。

- 《ユニバーサル·シリアル·バス仕様 2.0》(即ち USB 仕様)、2000 年 4 月 27 日
- 《ユニバーサル·シリアル·バス共通クラス仕様 1.0》、1997 年 12 月 16 日
- 《ユニバーサル·シリアル·バス·デバイス·クラス:集積回路(S)カード·インターフェース·デバイス用のスマートカード CCID 仕様 1.1》、2005 年 4 月 22 日

下記のファイルは www.ansi.org でオーダーできます。

- 《ISO/IEC 7816-1:識別カード ー 接点付きの集積回路(S)カード ー パート 1:物理特性》
- 《ISO/IEC 7816-2:識別カード ー 接点付きの集積回路(S)カード ー パート 2:接点のサイズと場所》
- 《ISO/IEC 7816-3:識別カード ー 接点付きの集積回路(S)カード ー パート 3:電子信号及び伝送プロトコル》

1.2. シンボルと略語

略語	説明
ATR	リセット応答(Answer-To-Reset)
CCID	チップ/スマートカードインターフェースデバイス(Chip/Smart Card Interface Device)
ICC	集積回路カード(Integrated Circuit Cards)
IFSC	プロトコル T=1 の集積回路カードの情報フィールドのサイズ(Information Field Sized for ICC for protocol T=1)
IFSD	プロトコル T=1 のチップ/スマートカードインターフェースデバイスの情報フィールドのサイズ(Information Field Sized for ICC for protocol T=1)
NAD	ノードの場所(Node Address)
PPS	プロトコル及びパラメ—タ—セレクション(Protocol and Parameters Selection)
RFU	保留して将来使います。1
TPDU	転送プロトコルデータユニット Application Protocol Data Unit
USB	ユニバーサル·シリアル·バス(Universal Serial Bus)

表1 : シンボルと略語

1 特別の説明がなければ、0に設置しなければなりません。

2.0. 特性

- USB フルスピード·インターフェース
- ブルートゥース®インターフェース
- プラグアンドプレイ - CCID サポートにより、最大限の互換性と機動性を実現
- スマートカードリーダー:
  - 接触式インターフェース
    - ISO 7816 クラス A 、B、C の(5V、3V 及び 1.8V)カードをサポート
    - 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
    - 様々のメモリカードサポート
    - PPS サポート(プロトコルとパラメータの選択)
    - 短絡保護保有
    - AES128 暗号化アルゴリズムをサポートしている
- 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
  - (PC / SC の上のラッパー経由で)、CT- API をサポート
- 内蔵の部品:
  - LED
- USB ファームウェアアップグレード可能 [2]
- Android™ 4.3と以降のバージョンサポートしている [3]
- iOS 5.0と以降のバージョンサポートしている [4]
- 以下の基準に一致している
  - EN60950/IEC 60950
  - ISO 7816
  - Bluetooth®
  - EMV™ Level 1 (接触)
  - PC/SC
  - CCID
  - CE
  - FCC
  - RoHS 2
  - REACH
  - VCCI (日本)
  - MIC (日本)
  - Microsoft® WHQL

[2] *PC リンクモードに適用する。*
[3] *ACS で定義された Android ライブラリを使用しています*
[4] *ACS で定義された iOS ライブラリを使用しています*

3.0. サポートしているスマートカード

3.1. MCU カード

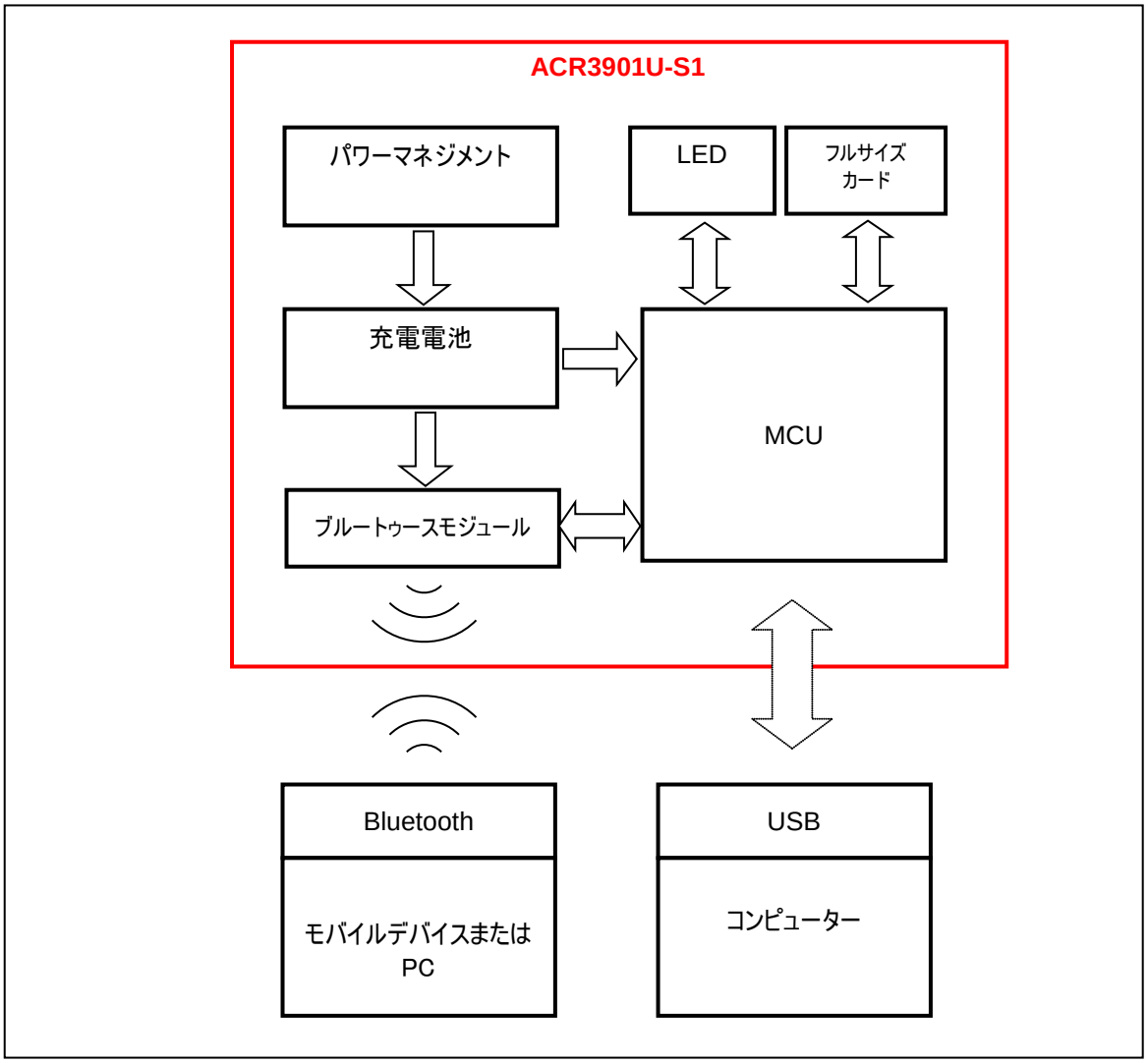
ACR3901U-S1 は PC/SC 仕様に準拠しているスマートカードリーダーです。ISO 7816 クラス A 、B、C の(5V、3V 及び 1.8V)カードをサポートしている。全ての T = 0 または T = 1 プロトコルに準拠している MCU カードをサポートしている。
カードの ATR が専用の操作モードを指定すれば(TA2が存在している;TA2 のビット 5 は 0 でなければなりません)、しかし ACR3901U-S1 がこのモードをサポートできない場合、ACR3901U-S1 はカードをリセットして、交渉モードに設置します。交渉モードを設置できないと、ACR3901U-S1 がこのカードを拒否します。
カードの ATR が交渉のモード(TA2 が存在指定ない)及び通信パラメータ(デフォルトパラメータじゃなくて)を指定すれば、ACR3901U-S1 がその通信パラメータを使用してみて、PPS を実行する。ACR32 が PPS を拒否しったら、デフォルトパラメ—タ—を使用する(F=372, D=1)。
上記のパラメ—タ—の意味について、ISO 7816-3 仕様を参照してください。

3.2. メモリカード

ACR3901U-S1 がサポートしているメモリカード、例:

- I2C バスプロトコルに準拠し、一回で 128 バイト/ページを書くことができるメモリカード(フリーメモリカード)、以下を含めて:
  - Atmel®:AT24C01/02/04/08/16/32/64/128/256/512/1024
  - SGS-Thomson:ST14C02C、ST14C04C
  - Gemplus:GFM1K、GFM2K、GFM4K、GFM8K
- パスワードと認証によるセキュアなメモリ IC カード、以下を含めて:
  - Atmel®:AT88SC153 和 AT88SC1608
- 書き込み保護機能付インテリジェント 1 KB の EEPROM カード、以下を含めて:
  - Infineon®:SLE4418、SLE4428、SLE5518 和 SLE5528
- インテリジェント 256 バイトの EEPROM、書き込みのカードプロテクト機能付カード、以下を含めて:
  - Infineon®:SLE4432、SLE4442、SLE5532 和 SLE5542
- '104'タイプ EEPROM(読み取りオンリー型トークンカウンタカード、以下を含めて:
  - Infineon®:SLE4406、SLE4436、SLE5536 和 SLE6636
- インテリジェント 416 バイトの EEPROM、書き込みのカードプロテクト機能付カード、以下を含めて:
  - Infineon®:SLE4404
- アプリケーションゾーンでのセキュリティ·ロジックを使用したカード、以下を含めて:
  - Atmel®:AT88SC101、AT88SC102 和 AT88SC1003

4.0. システムのブロック·デザイン



図示1. ::ACR3901U-S1 アーキテクチャ

5.0. ハードウェアのデザイン

5.1. バッテリー

ACR3901U-S1 は 320 ミリアンペアの充電式リチウムイオン電池を使用しています。

5.1.1. バッテリーの充電

ACR3901U-S1 のバッテリーが切れると、次のいずれかのモードで充電することができる：OFF モード、USB モード、Bluetooth モード;前提条件は電源コンセントに接続されています。

5.1.2. バッテリーの寿命

電池寿命は装置の使用に依存しています。様々な作業条件に応じて、バッテリ寿命の推定値は以下のように:

モード	推定の電池寿命
動作モード	24 日*(1)
スタンバイモード	28 日(2)
オフモード	2 年

表2：推定のバッテリの寿命

***注:**異なるスマートカードで結果が違います。

(1) Bluetooth モードでは、1 分間の操作で 1 日に 10 回の操作を実行します。

(2) Bluetooth モードでは、スリープ時間を 60 秒に設定し、1 日に 1 回ウェイクアップします。

5.2. ブルートゥースインターフェース

ACR3901U-S1 は低消費のブルートゥース 4.0 インターフェースで、デバイスを PC/モバイルデバイスとペアリングします。

5.3. USB インターフェース

Micro USB が充電バッテリーのポートとして、コンピュータに ACR3901U-S1 を接続するために使用されています。このポートは、PC-リンクモードで ACR3901U-S1 を動作させるために使用されています。

5.3.1. 通信パラメーター

ACR3901U-S1 は USB 2.0 仕様の USB インターフェース介してコンピュータに接続されます。フルスピードモードをサポートできて、12 mbps で働いています。

ピン	信号	機能
1	V_{BUS}	カードに+5 V の電源を提供します。
2	D-	ACR3901U-S1 と PC は差動信号でデータを転送します。
3	D+	ACR3901U-S1 と PC は差動信号でデータを転送します。
4	GND	参照用の電圧レベル

表3： USB インターフェース配線

5.3.2. エンドポイント

ACR3901U-S1 が下記のエンドポイントを介して、ホストの PC と通信します:

エンドポイント制御（Control Endpoint）	設置と制御の用
バルクアウト（Bulk OUT）	ホストから ACR3901U-S1 に送信するコマンドに対して（ペイロードの大きさは 64 バイトです）
バルクイン（Bulk IN）	ACR3901U-S1 からホストに返す応答に対して（ペイロードの大きさは 64 バイトです）
割り込み入力（Interrupt IN）	ACR3901U-S1 からホストに送信する状態メッセージに対して（ペイロードの大きさは 8 バイトです）

5.4. ユーザーインターフェース

5.4.1. モード選択スイッチ

ACR3901U-S1 は三つのモードを提供しています:USB、OFF とブルートゥース。ユーザーはデータ伝送インターフェースとして、一つのモードを選択することができます。

シンボル	スイッチ	アクティブ
	USB	PC リンク
	OFF	パワーOFF
	Bluetooth	Bluetooth

表4:モード選択スイッチ

5.4.2. LED ステータスインジケータ

異なる操作状態を表示するために、ACR3901U-S1 は三つの LED を提供しています。

- 赤の LED - 電池状態
- 青の LED - Bluetooth mode でカードとリーダーの状態
- 緑の LED -USB モードでカードとリーダーの状態

色	LED ステータス	状態
赤	点灯	バッテリーは充電中です（バッテリーが完全に充電された後はオフになります）
	ゆっくり点滅 （1 秒/点滅）	充電必要
青	速い-ゆっくり点滅 （高速：250ms /フラッシュ、遅い：500ms /フラッシュ）	Bluetooth デバイスとペアリング準備完了
	ゆっくり点滅 （2 秒/点滅）	ブルートゥース設備と接続済、カード操作なし
	速く点滅する	リーダーとモバイルデバイスがデータを通信している
	点灯	カードが接続して、電源いれました
緑	ゆっくり点滅 （2 秒/点滅）	カード操作なしで、ACR3901T-W1 は PC 命令を待っています
	速く点滅する	リーダーと PC がデータを通信している
	点灯	カードが接続して、電源いれました

表5 : LED ステータスインジケータ

注:リーダーがブルートゥースモジュールからのエラーメッセージを受信すると、青と緑の *LED が1秒で点灯して、消します。*

5.5. スマートカードインターフェース

ACR3901U-S1 と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、ACR3901U-S1 の実用的な機能性を高めるために一定の制限や機能拡張をする。

5.5.1. 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 50mA よりも高くてはならない。

5.5.2. プログラミング電圧 VPP（C6）

ISO7816-3 仕様によると、スマートカードコンタクト C6（VPP）がスマートカードにプログラミング電圧を供給する。市場内のすべてのスマートカードが EEPROM ベースであり、外部プログラミング電圧の供給の必要がないです。
ACR3901U-S1 のコンタクト C6（VPP）が通常の制御信号として実装されました。このコンタクトの電気仕様は信号 RST（コンタクト C2）の電気仕様と同じです。

5.5.3. 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、ACR3901U-S1 に適切なコマンドを送信してカードタイプを選択する必要があります。メモリカートと MCU ベースカードも含めています。
MCU ベースのカードに対して、T=0 と T=1 を同時にサポートしている場合、リーダーがプロトコルとパラメ―タ―オプション（**PPS**）を選ぶことによって、優先プロトコルとして、T=0 または T=1 の一つを選びます。MCU カードは 1 つだけのプロトコル（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選ぶことと関係なくて、リーダーは自動的にこのプロトコルタイプを選択します。

5.5.4. 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カードは C1（VCC）、C2（RST)、C3（CLK）、C5（GND）及び C7（I/O）これらのコンタクトだけ使用します。4.8 MHz の周波数が CLK 信号（C3）に適用します。

5.5.5. カード引き裂き保護

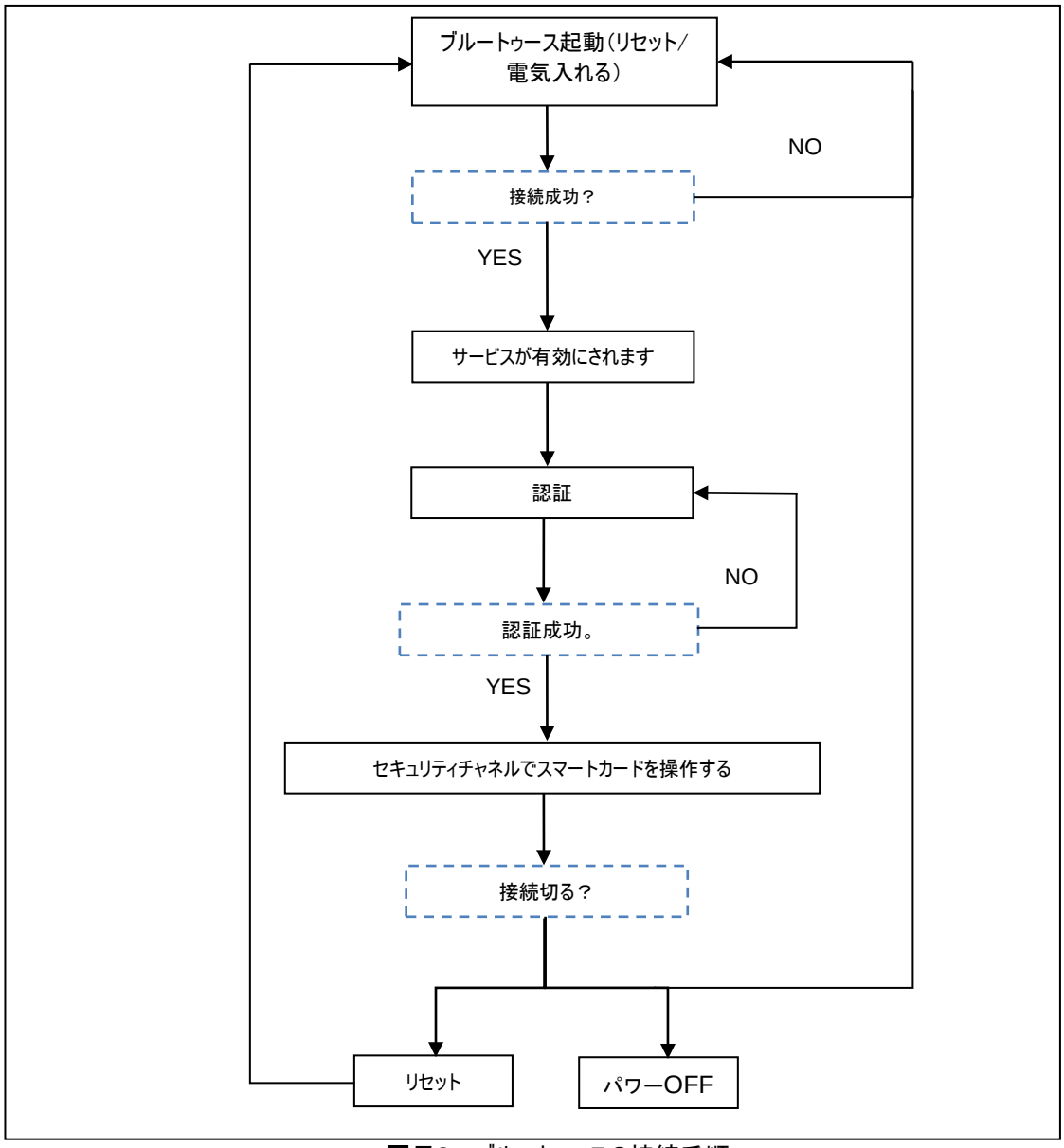
電気入れる状態で、急に引き出されたカードを保護するために、ACR3901U-S1 がメカニズムを提供している。カードが取り外されている時、ACR3901U-S1 とカード間の信号線への電力供給がすぐにに非アクティブ化されます。原則として、電気的な損傷を回避するために、パワーダウンしてから、カードをリーダーから除去されるべきです。
注:*ACR3901U-S1 は、決して挿入されたカードへの電源供給に切り替わりません。ホストから適切なコマンドが読者に送られ、明示的にこの操作を行う必要があります。*

6.0. ハードウェアのデザイン

6.1. ブルートゥース通信プロトコル

6.1.1. ブルートゥース接続手順

下記は Bluetooth の接続フローです:



図示2. :ブルートゥースの接続手順

6.1.2. コンポーネントファイルを選ぶ

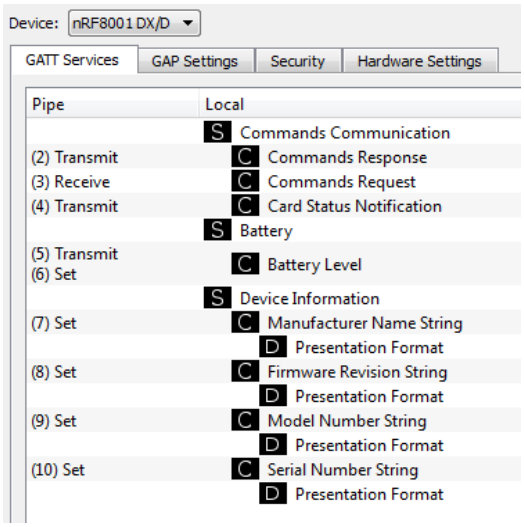
ACR3901U-S1 はデータを送信するためのインタフェースとしての Bluetooth 技術を使用するように設計されたスマートカードリーダーです。3 つのパイプでコマンド通信できるカスタマイズされたサービスが使用されている：一つのパイプがコマンド要求のために使用され、二つのパイプは、コマンド応答に使用され、もう一つのパイプでカードのモードと状態をペアリングのデバイスに知らせます。
また、リーダーは Bluetooth モードで動作している時、現在の電力消費量が重要なので、従って、カスタマイズのバッテリサービスが現在のバッテリーステータスをペアリング装置に通知するために使用されます。電池状態の変化があった場合、リーダーは、特定のパイプを介してペアリング装置に通知します。簡単に言えば、電池の容量は三つのレベルに分けています。下記のテーブルは各レベルと対応の返す数値を示します：

状態	電圧	返ってきた数値
十分	≥ 3.3 V	FEh
電源不足	<3.3 V、それに ≥ 2.9 V	FFh/FEh/00h 以外の数値
電池なし	<2.9 V	00h
USB モード		FFh

カードの状態には変更があるまたはリーダーがSleepモードに入る場合、カード状態通知サービスはペアリングサービスに知らせます。下記はステータステーブルと対応の返す数値：

状態	返ってきた数値
カード挿入されていない	50 02h
カードが挿入されました	50 03h
リーダーは Sleep モードに入りました	50 04h

ユーザーに多くの情報を提供するように、カスタマイズされたデバイス情報サービスを追加しました。デバイス情報（メーカネーム、ファームウェアバージョン、型番及びシリアルナンバー）をユーザー自分で読み取れます。またアプリケーションからリクエストがあって、デバイス情報を読み取れます。



図示3. : nRFgo Studio GATT 配置ページ

nRFgo-Studio 配置には一つのサービスが追加されて、総計 10 項のサービスがあります：

```
#define PIPE_GAP_DEVICE_NAME_SET 1
#define PIPE_COMMANDS_COMMUNICATION_COMMANDS_RESPONSE_TX 2
#define PIPE_COMMANDS_COMMUNICATION_COMMANDS_REQUEST_RX 3
#define PIPE_COMMANDS_COMMUNICATION_CARD_STATUS_NOTIFICATION_TX 4
#define PIPE_BATTERY_BATTERY_LEVEL_TX 5
#define PIPE_BATTERY_BATTERY_LEVEL_SET 6
#define PIPE_DEVICE_INFORMATION_MANUFACTURER_NAME_STRING_SET 7
#define PIPE_DEVICE_INFORMATION_FIRMWARE_REVISION_STRING_SET 8
#define PIPE_DEVICE_INFORMATION_MODEL_NUMBER_STRING_SET 9
#define PIPE_DEVICE_INFORMATION_SERIAL_NUMBER_STRING_SET 10
```

#define NUMBER_OF_PIPES 10

*#define PIPE_GAP_DEVICE_NAME_SET が実行している間にもアプリケーションコントロールにデバイスネームを変更されます。*Bluetooth mode では、公告ネームのフォーマットは“ACR3901U-S1XXXXXXX”です。その中“XXXXXXX”はリーダーのシリアルナンバーの最後の 7 バイトです。
公告ネームを“ACR3901U-S1XXXXXXX”にするために、Bluetooth mode’を有効にします。
Bluetooth mode を有効にする：

1. 設定（06h）配置を Bluetooth mode にアップロードします。
2. チャンネル 1 でデバイスネームのフォーマットを“ACR3901U-S1XXXXXXX”に設定します。(PIPE_GAP_DEVICE_NAME_SET)
3. 接続（0Fh）。
4. 公告。

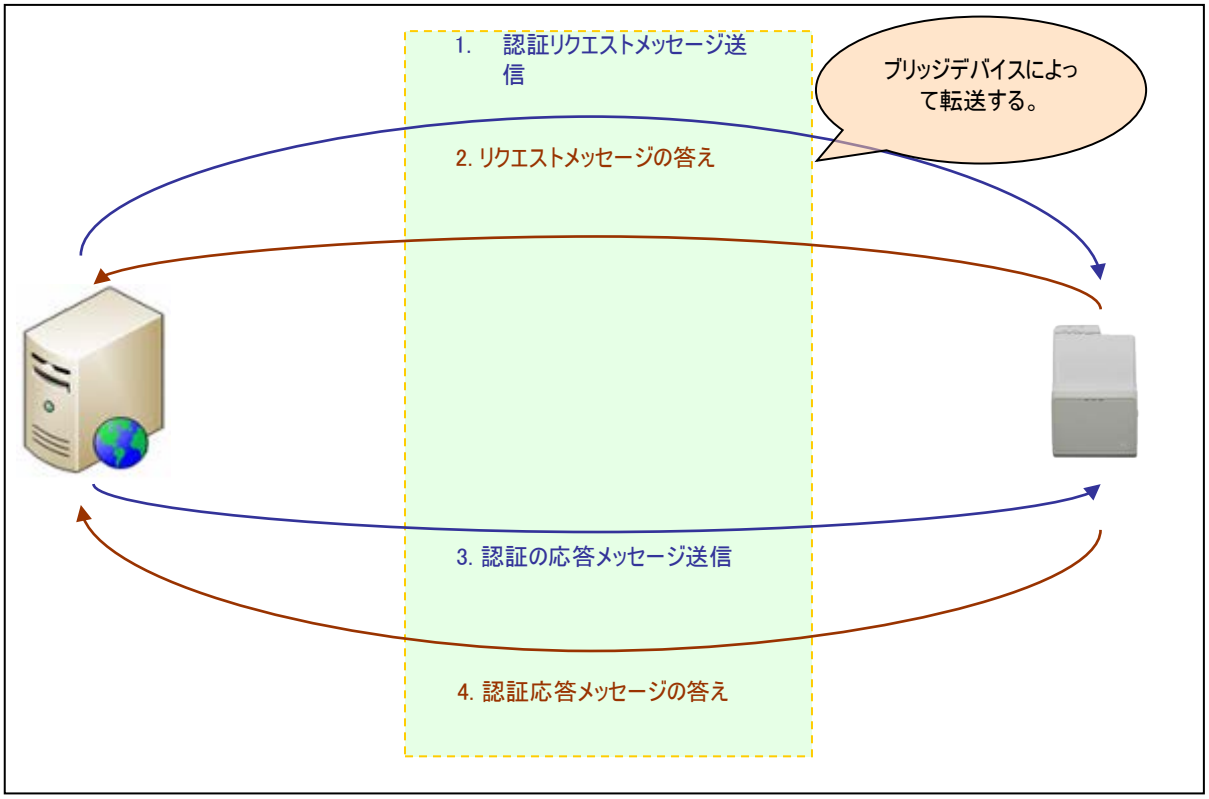
属性名称	UUID	ハンドル
DeviceName	2A00	03h
Send (Reader → Paired device)	8002	0Bh
受信 (Paired device →Reader)	8003	0Eh
CardStatus	8004	10h
BatteryLevel	2A19	14h
Manufacturer	2A29	18h
FW_Version	2A26	1Bh
ModelNumber	2A24	1Eh
SerialNumber	2A25	21h

表6：ACR3901U-S1 のサービスハンドルと UUID メッセージリスト

6.1.3. 認証

機密データが ACR3901U-S1 にロードする前に、データ処理サーバは、リーダ内部で保護されたデータを変更する権限のため ACR3901U-S1 によって認証されなければなりません。ACR3901U-S1 は相互認証を採用します。

良い説明のために、下の図示を参照してください（シンプルさとより良い説明のために、下の図示がブリッジデバイスを省略している）



図示4. ：認証手順

認証に成功すると、ACR3901U-S1 とデータサーバにそれぞれ 16 バイトのプロセスキーが生成されます。
デフォルトの顧客マスタキー: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

注意：認証キーは 6 回まで間違っていますが、それを超えると、リーダーはロックされて使用できなくなります。詳しくは ACS の販売員までお問い合わせてください。

6.1.4. フレームフォーム

6.1.4.1. ブルートゥースフレームフォーマット

HID フレーム	長さ(バイト)	説明
識別子	1	コマンド
長さ	2	長さ {ペイロード+チェックサム}
ペイロード	0-N	データ
チェックサム	1	XOR {識別子、長さ、ペイロード}

表7 : ブルートゥースフレームフォーマット

フレームフォームは:

識別子 + LEN1 + LEN2 + N-バイトペイロード + チェックサム

コマンドの長さ(識別子、長さ及びペイロードを含めて)は20バイトより長い場合、リーダーまたはペアリングデバイスは自動にいくつのフレームに分けます。

データチェックサムは、無線にデータ送信時に導入された可能性のあるエラーを検出するのに使用されます。データチェックサムを計算する必要な場合:XOR {識別子、長さ、ペイロード}。

例: 62010063 => チェックサム = 63h

6.1.4.2. 相互認証したブルートゥースフレームフォーマット

誰かブルートゥース通信チャネルを介しての攻撃を防止するために、相互認証を投入します。相互認証が成功すると、**表 7** 中のブルートゥースフレームフォーマットが暗号化され、ヘッダ1バイト、Len2 バイト、チェック 1 バイトにカプセル化されます。相互認証したブルートゥースフレームフォーマットの構造は:

ヘッダ + Len + (識別子 + 長さ + ペイロード + チェックサム)* + チェックバイト

***注**:各 16 バイトのデータは、AES-128 CBC 暗号化モードを使用してプロセスキーによって暗号化されます。AES-128 CBC 暗号化モードでは、初期ベクトルは 16 バイト(00h)である。*

HID フレーム	長さ(バイト)	説明	
ヘッダバイト	1	値:72h/22h	
Len	2	長さ {識別子 + 長さ + ペイロード + チェックサム + チェック+終止バイト}	
識別子	1	コマンド	ブルートゥースフレームフォーマットの暗号化データ: この部分の最終のデータの長さは 16*N バイト (N>0)
長さ	2	長さ {ペイロード+チェックサム}	
ペイロード	0-N	データ	
チェックサム	1	XOR {識別子、長さ、ペイロード}	
チェックバイト	1	XOR {ヘッダ、Len、暗号化された(識別子、長さ、ペイロード、チェックサム)}	

表8 : 相互認証した暗号化されたブルートゥースフレームフォーマット

6.1.5. ブルートゥース通信プロトコル

ACR3901U-S1 は、予め定義されたプロトコルを採用して、ブルートゥース・インターフェースでペアリングされたデバイスと通信します。プロトコルは、CCID コマンドパイプと応答パイプの形式に似ています。

コマンド	サポートモード	送信側	説明
62h	認証済	ペアリングデバイス	ICC 電気入れる
63h	認証済	ペアリングデバイス	ICC 電源がオフになる（ICC Power Off）
65h	認証済	ペアリングデバイス	カードあるかどうかチェック（Get Card Presence）
6Fh	認証済	ペアリングデバイス	APDU 交換（Exchange APDU）
67h	認証済	ペアリングデバイス	交換 APDU2
61h	認証済	ペアリングデバイス	パラメ―タ―を設定する
6Bh	認証済	ペアリングデバイス	外部コマンド
70h	接続済/認証済	ペアリングデバイス	SPH_to_RDR_ReqAuth*
71h	接続済/認証済	ペアリングデバイス	SPH_to_RDR_AuthRsp*

表9 :コマンドコードの概要

コマンド	サポートモード	送信側	説明
12h	認証済	リーダー	ICC Power On コマンドの応答
13h	認証済	リーダー	ICC Power Off コマンドの応答
14h	認証済	リーダー	Get Card Presence コマンドの応答
11h	認証済	リーダー	Exchange APDU コマンドの応答
17h	認証済	リーダー	Exchange APDU2 コマンドの応答
16h	認証済	リーダー	設定のパラメーターに対して応答する
15h	認証済	リーダー	外部コマンドの応答
20h	接続済/認証済	リーダー	RDR_to_SPH_AuthRsp1*
21h	接続済/認証済	リーダー	RDR_to_SPH_AuthRsp2*

表10 :コマンドコードの概要

***注:**これらのコマンド/応答は相互認証に使用される通信コードです。*

6.1.5.1. カードに電源を入れる（Card Power On）

このコマンドはリーダにパワーアップのリクエストを送信するために使用されます。

コマンドのフォーマット

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**62h**	
1	*LEN1 LEN2 (wLength)*	2	0100h	このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	*CSUM (wChecksum)*	1	63h	CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	*1*	*12h*	
1	*LEN1 LEN2 (wLength)*	*2*		このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	*N バイトの ATR*	*N*		カードリセット応答
3+N	*CSUM (wChecksum)*	*1*		CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット（エラー）

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	*1*	*92h*	-
1	*LEN1 LEN2 (wLength)*	*2*	*0200h*	このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	*Error Code (bErrorCode)*	*1*	-	*エラーコード、付録 A 参照*
4	*CSUM (wChecksum)*	*1*	-	*CSUM は、コマンド内のすべてのバイトの XOR 値を表します。*

例:
リクエスト = 62 01 00 63
応答 = 12 14 00 3B BE 11 00 00 41 01 38 00 00 00 00 12 34 56 78 01
90 00 73
ATR = 3B BE 11 00 00 41 01 38 00 00 00 00 12 34 56 78 01

6.1.5.2. カードに電源をオフにする（Card Power Off）

このコマンドはリーダにパワーオフのリクエストを送信するために使用されます。

コマンドのフォーマット

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**63h**	
1	*LEN1 LEN2 (wLength)*	2	0100h	このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	*CSUM (wChecksum)*	1	62h	CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**13h**	
1	*LEN1 LEN2 (wLength)*	2	0100h	このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	*CSUM (wChecksum)*	1	12h	CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット（エラー）

オフセット	フィールド	大きさ	数値	説明
0	bMessageType	1	93h	-
1	LEN1 LEN2 (wLength)	2	0200h	このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	Error Code (bErrorCode)	1	-	*エラーコード、付録 A 参照*

例:
リクエスト = 62 01 00 62
応答 = 13 01 00 12

6.1.5.3. カードプレゼンス取得（Get Card Presence）

このコマンドはカードが挿入されているかどうかをチェックするために使用されます。

コマンドのフォーマット

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**65h**	
1	*LEN1 LEN2 (wLength)*	2	0100h	このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	*CSUM (wChecksum)*	1	64h	CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	14h	
1	*LEN1 LEN2 (wLength)*	2	0200h	このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	*STA*	1		カードステータス: 00 = 未知状態 01 = カードなし 02 = カードある、アクティブされていない 03 = カードある、アクティブされた
4	*CSUM (wChecksum)*	1		CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット（エラー）

オフセット	フィールド	大きさ	数値	説明
0	bMessageType	1	94h	-
1	LEN1 LEN2 (wLength)	2	0200h	このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	Error Code (bErrorCode)	1	-	*エラーコード、付録 A 参照*

例:
リクエスト = 65 01 00 64
応答 = 14 02 00 03 15

6.1.5.4. APDU コマンド（APDU Command）

このコマンドはリーダに APDU コマンドを送信するために使用されます。

コマンドのフォーマット

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**6Fh**	
1	*LEN1 LEN2 (wLength)*	2		このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	*APDU CMD*	N		APDU コマンド
3+N	*CSUM (wChecksum)*	1		CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	11h	
1	*LEN1 LEN2 (wLength)*	2		このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	*APDU 応答*	N		APDU フォーマットデータ
3+N	*CSUM (wChecksum)*	1		CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット（エラー）

オフセット	フィールド	大きさ	数値	説明
0	bMessageType	1	91h	-
1	LEN1 LEN2 (wLength)	2	0200h	このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。
3	Error Code (bErrorCode)	1	-	*エラーコード、付録 A 参照*

例:
リクエスト = 6F 06 00 80 84 00 00 08 65
応答 = 11 0B 00 C1 7A 3B AA D6 5A FA CE 90 00 18

6.1.5.5. APDU2 コマンド（FW v1.20 以降）

このコマンドは、拡張 APDU をサポートする APDU コマンドをリーダーに送信します。
コマンドのフォーマット

オフセット	データフィールド		大きさ	数値	説明
0	*bMessageType*		1	**67h**	-
1	*LEN1 LEN2 (wLength)*		2	-	このメッセージ中で、次のフィールドからのバイト数を表して、2 バイトの長さを示します。 LEN1 は LSB、LEN2 は MSB です。 （最大長さは 263）
3	*データ*	*Param*	1	-	パラメーター: 短 APDU レベル 00h – ディフォルト 拡張 APDU レベル 00h - コマンド APDU がこのコマンドで開始および終了します。 01h - このコマンドでコマンド APDU が開始され、次の APDU コマンドで続行されます。 02h - データフィールドは引き続きコマンド APDU を渡し、APDU コマンドを終了します。 03h - データフィールドはコマンド APDU の後に続けて別のデータブロックが続きます。 10h - 空のデータフィールド、次の応答は応答 APDU を渡し続けます
4		*APDU CMD*	N	-	APDU コマンド （最大長さは 261）
4+N	*CSUM (wChecksum)*		1	-	CSUM はデータ中でのすべての XOR 値を示す。

応答データフォーマット

オフセット	データフィールド	大きさ	数値	説明
0	*bMessageType*	1	**17h**	-
1	*LEN1 LEN2 (wLength)*	2	-	このメッセージ中で、次のフィールドからのバイト数を表して、2 バイトの長さを示します。 LEN1 は LSB、LEN2 は MSB です。

オフセット	データフィールド		大きさ	数値	説明
3	*データ*	*Param*	1	-	パラメ―タ―: 短 APDU レベル 00h – ディフォルト 拡張 APDU レベル 00h - 応答 APDU がこのコマンドで開始および終了します。 01h - 応答 APDU がこのコマンドで開始および続ける。 02h - データフィールドは引き続き応答 APDU を渡し、APDU 応答を終了します。 03h - データフィールドは応答 APDU の後に続けて別のデータブロックが続きます。 10h - 空のデータフィールド、次のコマンドはコマンド APDU を渡し続けます
4		*APDU RSP*	N	-	APDU 応答
4+N	*CSUM (wChecksum)*		1	-	CSUM はデータ中でのすべての XOR 値を示す。

応答データフォーマット(WTX)

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**18h**	-
1	*LEN1 LEN2 (wLength)*	2	0300h	このメッセージ中で、次のフィルドからのバイト数を表して、2 バイトの長さを示します。LEN1 は LSB、LEN2 は MSB です。
3	*STA*	1	-	カードステータス: 00 = 未知状態 01 = カードなし 02 = カードが存在し、アクティブ化されていない 03 = カードが存在し、アクティブ化されてる
4	*WTXM*	1	-	待ち時間遅延乗数
5	*CSUM (wChecksum)*	1	-	CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット（エラー）

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**97h**	-
1	*LEN1 LEN2 (wLength)*	2	0200h	このメッセージ中で、次のフィルドからのバイト数を表して、2 バイトの長さを示します。LEN1 は LSB、LEN2 は MSB です。
3	*Error Code (bErrorCode)*	1	-	エラーコード、*付録 A 参照*

オフセット	フィールド	大きさ	数値	説明
4	*CSUM (wChecksum)*	1	-	CSUM はコマンド中でのすべての XOR 値を示す。

例:
カードに 600 バイトのデータを送信する

1. コマンド= 67 07 01 01（データの 261 バイト）チェックサム
   レスポンス= 17 02 00 10 チェックサム
2. コマンド= 67 07 01 03（データの 261 バイト）チェックサム
   レスポンス= 17 02 00 10 チェックサム
3. コマンド= 67 50 00 02（データの 78 バイト）チェックサム
   レスポンス= 17 04 00 00 90 00 チェックサム

カードから 600 バイトのデータを受信する

1. コマンド= 67 09 00 00 00 B0 87 00 00 02 58 チェックサム
   コマンド= 17 02 01 01（データの 256 バイト）チェックサム
2. コマンド= 67 02 00 10 チェックサム
   コマンド= 17 02 01 03（データの 256 バイト）チェックサム
3. コマンド= 67 02 00 10 チェックサム
   応答=17 5C 00 02（データの 88 バイト）90 00 チェックサム

6.1.5.6. ダイレクトコマンド（Escape Commands）

このコマンドでリーダーの拡張機能をアクセスできます。

コマンドのフォーマット

オフセット	フィールド		大きさ	数値	説明
0	*bMessageType*		1	**6Bh**	ダイレクトコマンドのヘッダ
1	*LEN1 LEN2 (wLength)*		2		フィールドは 2 バイト長で、フィールドの次のフィールドから始まるこのメッセージの他のバイト数を表し、LEN1 は LSB であり、LEN2 は MSB です。
3	*abData 1*	*CommandCode*	1		コマンドヘッダ
4		*Len (CommandLength)*	1		このメッセージ中で、次のフィールドからのバイト数を表して、1 バイトの長さを示します。
5		*Data*	N		0 =< N <= 255
5+N	*CSUM (wChecksum)*		1		CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット

オフセット	フィールド		大きさ	数値	説明
0	*bMessageType*		1	15h	ダイレクトコマンド応答のヘッダ
1	*LEN1 LEN2 (wLength)*		2		フィールドは 2 バイト長で、フィールドの次のフィールドから始まるこのメッセージの他のバイト数を表し、LEN1 は LSB であり、LEN2 は MSB です。
3	abData2	*ResponseCode*	1		応答のヘッダ
4		*Len (CommandLength)*	1		このメッセージ中で、次のフィールドからのバイト数を表して、1 バイトの長さを示します。
5		*Data*	N		0 =< N <= 255
5+N	*CSUM (wChecksum)*		1		CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット（エラー）

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**95h**	-
1	*LEN1 LEN2 (wLength)*	2	0200h	このメッセージ中で、次のフィールドからのバイト数を表して、2 バイトの長さを示します。LEN1 は LSB、LEN2 は MSB です。
3	*Error Code (bErrorCode)*	1	-	エラーコード、付録 A 参照
4	*CSUM (wChecksum)*	1	-	CSUM はコマンド中でのすべての XOR 値を示す。

6.1.5.7. カードセットパラメータ（Card Set Parameters）

このコマンドはパワーアップされているカードのパラメーターを変更する時に使われます。

コマンドのフォーマット

オフセット	データフィールド		大きさ	数値	説明
0	*bMessageType*		1	**61h**	-
1	*LEN1 LEN2 (wLength)*		2	-	このメッセージ中で、次のフィルドからのバイト数を表して、2 バイトの長さを示します。LEN1 は LSB、LEN2 は MSB です。
3	*abData 1*	*ProtocolNum*	1	-	プロトコルのデータ構造： 00h = T=0 プロトコルの構造 01h = T=1 プロトコルの構造
4		*ProtocolDataStructure*	N	-	プロトコルのデータ構造
4+N	*CSUM (wChecksum)*		1	-	CSUM はコマンド中でのすべての XOR 値を示す。

応答データフォーマット

オフセット	データフィールド		大きさ	数値	説明
0	*bMessageType*		1	**16h**	ダイレクトコマンド応答のヘッダ
1	*LEN1 LEN2 (wLength)*		2	-	このメッセージ中で、次のフィルドからのバイト数を表して、2 バイトの長さを示します。LEN1 は LSB、LEN2 は MSB です。
3	*abData 2*	*ProtocolNum*	1	-	プロトコルのデータ構造： 00h = T=0 プロトコルの構造 01h = T=1 プロトコルの構造
4		*ProtocolDataStructure*	N	-	プロトコルのデータ構造
4+N	*CSUM (wChecksum)*		1	-	CSUM はコマンド中でのすべての XOR 値を示す。

T=0 プロトコルのデータ構造（ProtocolNum = 0、wLength = 0700h）

オフセット	データフィールド	大きさ	数値	説明
4	*bmFindexDindex*	1	-	B7-4 – FI – ISO/IEC 7816-3:1997 中のテーブル 7 をインデックスして、クロックレートの変換係数を選択します。B3-0 – DI - ISO/IEC 7816-3:1997 中のテーブル 8 をインデックスして、ボーレートの変換係数を選択します
5	*bmTCCKST0*	1	-	B0 – 0b, B7-2 – 000000b B1 – 使用している約束（b1=0：直接；b1=1：逆） *注：CCID がこのビットを無視します。*
6	*bGuardTimeT0*	1	-	2 文字間の余分な GuardTime。 正常な GuardTime（12 ETU）に 0ー254 ETU を追加します。 FFh と 00h が同じです。
7	*bWaitingIntegerT0*	1	-	T=0 の場合 WI が WWT を定義する時に使われます
8	*bClockStop*	1	-	ICC クロック停止サポート 00h = クロックを停止することは許可されていません 01h = クロック信号が低い時に停止されます 02h = クロック信号が高い時に停止されます 03h = クロック信号が低い時または高い時に停止されます

T=1 プロトコルのデータ構造（ProtocolNum = 1、wLength = 0900h）

オフセット	データフィールド	大きさ	数値	説明
4	*bmFindexDindex*	1	-	B7-4 – FI – ISO/IEC 7816-3:1997 中のテーブル 7 をインデックスして、クロックレートの変換係数を選択します。 B3-0 – DI - ISO/IEC 7816-3:1997 中のテーブル 8 をインデックスして、ボーレートの変換係数を選択します
5	*BmTCCKST1*	1	-	B7-2 – 000100b B0 – チェックサムタイプ（b0=0：LRC； b0=1：CRC） B1 – 使用している約束（b1=0：直接；b1=1：逆） 注：CCID がこのビットを無視します。
6	*BGuardTimeT1*	1	-	余計な GuardTime（2 文字間の余分な GuardTime は 0ー254etu）は FFh である場合、GuardTime を1etu 減らします。
7	*BwaitingIntegerT1*	1	-	B7-4 = BWI 値 0-9 有効 B3-0 = CWI 値 0-Fh 有効

オフセット	データフィールド	大きさ	数値	説明
8	*bClockStop*	1	-	ICC クロック停止サポート 00h = クロックを停止することは許可されていません 01h = クロック信号が低い時に停止されます 02h = クロック信号が高い時に停止されます 03h = クロック信号が低い時または高い時に停止されます
9	*bIFSC*	1	-	交渉された IFSC の大きさ
10	*bNadValue*	1	00h	NAD = 00h だけサポートできます

例 : (T0 プロトコル)
リクエスト = 61 07 00 00 11 00 00 0A 00 7D
応答 = 16 07 00 00 11 00 00 0A 00 0A

例 : (T1 プロトコル)
リクエスト = 61 09 00 01 96 10 00 45 00 FE 00 54
応答= 16 09 00 01 96 10 00 45 00 FE 00 23

6.1.6. 相互認証と暗号化プロトコル

Bluetooth mode で相互認証が成功してから、セッション中の通信プロトコルを暗号化し、転送します。

コマンド	サポートモード	送信側	説明
70h	接続済	ペアリングデバイス	SPH_to_RDR_ReqAuth
71h	接続済	ペアリングデバイス	SPH_to_RDR_AuthRsp
72h	認証済	ペアリングデバイス	SPH_to_RDR_DataReq
20h	接続済	リーダー	RDR_to_SPH_AuthRsp1
21h	接続済	リーダー	RDR_to_SPH_AuthRsp2
22h	認証済	リーダー	RDR_to_SPH_DataRsp

表11 :相互認証コマンドの概要

6.1.6.1. SPH_to_RDR_ReqAuth

このコマンドは、鍵生成装置に認証を行うために ACR3901U-S1 を要求します。

認証プロセスの詳細については、**認証**セッションを参照してください。

オフセット	フィールド	大きさ	数値	説明	暗号化
0	*bMessageType*	1	**70h**		NO
1	*LEN1 LEN2 (wLength)*	2	0100h	このフィルドの長さは2バイトです。このメッセージの中で、このフィルドの次フィルドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。	
3	*wChecksum*	1	71h	CSUM はコマンド中でのすべての XOR 値を示す。	

受信したコマンドメッセージはエラーがない場合は、RDR_to_SPH_AuthRsp1 が受信するはずです。そうじゃないと、エラーメッセージが含まれた RDR_to_SPH_ACK の応答を受信するはずです。

6.1.6.2. RDR_to_SPH_AuthRsp1

このコマンドはペアリングでバイズから送信された SPH_to_RDR_ReqAuth の応答です。
詳しい情報が**認証**を参照してください。

オフセット	フィールド	大きさ	数値	説明	暗号化
0	*bMessageType*	1	**20h**		NO
1	*LEN1 LEN2 (wLength)*	2	1100h	このフィルドの長さは2バイトです。このメッセージの中で、このフィルドの次フィルドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。	NO
3	*abRndNum*	16		abRndNum[0:15] – 16 バイトの乱数 すべての 16 バイトの乱数は現在に ACR3901U-S1 中でストレージされている顧客マスターキーで暗号化しなければなりません。	YES
19	*wChecksum*	1		CSUM はコマンド中でのすべての XOR 値を示す。	NO

6.1.6.3. SPH_to_RDR_AuthRsp

このコマンドは認証プロセスの第二段階です。デバイスは SPH_to_RDR_ReqAuth コマンドを ACR3901U-S1 に送信して、エラーがない場合はリーダーは RDR_to_SPH_AuthRsp1 メッセージを戻します。。RDR_to_SPH_AuthRsp1 には顧客マスターキーで暗号化された 16 バイトの乱数が含まれています。ペアリングされたキーの生成デバイスは正しい顧客マスターキーで復号化する必要があります。また、16バイトの乱数のあとに追加します。それに顧客マスターキーで32バイトの乱数全体を復号化して、結果をこのコマンドで ACR3901U-S1 に返すことによって、認証を完了します。

認証プロセスの詳細については、**認証**セッションを参照してください。

オフセット	フィールド	大きさ	数値	説明	暗号化
0	*bMessageType*	1	**71h**		NO
1	*LEN1 LEN2 (wLength)*	2	2100h	このメッセージ中で、次のフィルドからのバイト数を表して、2 バイトの長さを示します。LEN1 は LSB、LEN2 は MSB です。	NO
3	*abAuthData*	32		abAuthData[0:15] – データ処理サーバに生成された 16 バイトの乱数 abAuthData [16:31] - ACR3901U-S1 から受信した復号化された 16 バイトの乱数。 すべての 32 バイトのデータは、AES128 CBC 暗号化モードのクライアントマスターキーによって復号化されます。	YES
35	*wChecksum*	1		CSUM はコマンド中でのすべての XOR 値を示す。	NO

受信されたコマンドメッセージはエラーがないで、ACR3901U-S1 に返された乱数も正しい場合、応答は RDR_to_SPH_AuthRsp2 です。

6.1.6.4. RDR_to_SPH_AuthRsp2

このコマンドはペアリングでバイズから送信された *SPH_to_RDR_AuthRsp* の応答です。

詳しい情報が**認証**を参照してください。

オフセット	フィールド	大きさ	数値	説明	暗号化
0	*bMessageType*	1	**21h**		NO
1	*LEN1 LEN2 (wLength)*	2	1100h	このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。	NO
3	*abRndNum*	16		abRndNum[0:15] – データ処理サーバに受信された 16 バイトの乱数　。 すべての 16 バイトの乱数は現在に ACR3901U-S1 中でストレージされている顧客マスターキーで暗号化しなければなりません。	YES
19	*wCheckSum*	1		CSUM はコマンド中でのすべての XOR 値を示す。	NO

6.1.6.5. SPH_to_RDR_DataReq

このコマンドは相互認証プロセスが終ると、ペアリングデバイスから ACR3901U-S1 に送信されます。
Bluetooth mode で相互認証が成功してから、**6.1.5.1** から **6.1.5.7** 節までの通信プロトコルを暗号化し、転送します。

オフセット	フィールド	大きさ	数値	説明	暗号化
0	*bMessageType*	1	**72h**		NO
1	*LEN1 LEN2 (wLength)*	2		このフィルドの長さは2バイトです。このメッセージの中で、このフィルドの次フィルドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。	NO
3	*abEncryptedData*	N*16		16 バイトのデータはすべて AES128 CBC 暗号化モードのプロセスキーで暗号化されます。	YES
3 + N*16	*wChecksum*	1		CSUM はコマンド中でのすべての XOR 値を示す。	NO

abEncryptedData は、暗号化された（識別子+長さ+パケット+チェックサム）データ、長さ N * 16 バイトです。データ内の各バイトが AES128 CBC 暗号化モードで、相互認証によって生成されたプロセスキーを使用して、暗号化処理を実行します。

AES-128 CBC 暗号化モードでは、初期ベクトルは 16 バイトです。
元のデータの長さは N*16 より短い場合、暗号化する前に最後に FFh を追加して、長さを 16*N バイトにします。

HID フレーム	長さ（バイト）	説明	
識別子	1	コマンド	本当のデータは abEncryptedData で復号して、ダミーデータを削除します。
長さ	2	長さ {ペイロード+チェックサム}	
ペイロード	0-N	データ	
チェックサム	1	XOR {識別子、長さ、ペイロード}	

例:
相互認証が成功してから、ペアリングデバイスが電気入れるコマンドをリーダーに送信します。コマンドは:

72 11 00 XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX

その中:
コマンドヘッダ:72
電気入れるコマンドの暗号化データ（16 バイト）:XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX

受信したコマンドメッセージはエラーがない場合は、RDR_to_SPH_DataRsp が受信するはずです。

abEncryptedData は、通信プロトコルの暗号化されたデータです。16 バイトのデータはすべて AES-128 CBC 暗号化モードのプロセスキーで暗号化されます。

6.1.6.6. RDR_to_SPH_DataRsp

相互認証が成功してから、リーダはこのコマンドをペアリングデバイスに送信します。
Bluetooth mode で相互認証が成功してから、**6.1.5.1** から **6.1.5.7** 節までの通信プロトコルを暗号化し、転送します。

オフセット	フィールド	大きさ	数値	説明	暗号化
0	*bMessageType*	1	**22h**		NO
1	*LEN1 LEN2 (wLength)*	2		このフィールドの長さは2バイトです。このメッセージの中で、このフィールドの次フィールドからの他のバイト数を表します。LEN1 は LSB で、LEN2 は MSB です。	NO
3	*abEncryptedData*	N*16		すべての 16 バイトの乱数を AES128 CBC 暗号化モードで顧客マスターキーを使用して、復号化します。	YES
3 + N*16	*wChecksum*	1		CSUM はコマンド中でのすべての XOR 値を示す。	NO

6.2. USB 通信プロトコル

ACR3901U-S1 は USB を介して、ホストとのインターフェースを確立します。業界の規範 - CCID 標準は、USB チップ - スマートカードインタフェース装置に関わっているプロトコルを定義します。CCID 仕様はスマートカードを動作させるために必要な全てのプロトコルをカバーしています。

ACR3901U-S1 の USB エンドポイントの装置と使用は CCID 標準の Rev 1.0 のパート3に準拠するはずです。概要を以下に要約されている:

1. ***制御コマンド*** 制御パイプ(デフォルトのパイル)で送信されます。クラス固有の要求や USB 規格の要求が含めています。デフォルトのパイプで送信されたコマンドはデフォルトのパイプでホストにレポート情報を返します。
2. ***CCID イベント***割り込みパイプで送信されます。
3. ***CCID コマンド***BULK-OUT エンドポイントで送信されます。ACR3901U-S1 に送信された全てのコマンドは自分の関係エンディング応答を持っています。いくつかのコマンドは中間応答も持っています。
4. ***CCID 応答 BULK-IN エンドポイントで送信されます。***ACR3901U-S1 に送信された全てのコマンドは必ず同期に送信されます。(例:ACR3901U-S1 にとって、bMaxCCIDBusySlots は 01hに相当です)。

ACR3901U-S1 がサポートしている CCID 特性は下記のクラス記述子を参照します:

オフセット	フィールド	大きさ	数値	説明
0	*bLength*	1		この記述子のバイトサイズ
1	*bDescriptorType*	1		CCID 機能記述子のタイプ。
2	*bcdCCID*	2		2 進化した 10 進数での CCID 仕様のリリース番号。
4	*bMaxSlotIndex*	1		ACR3901U-S1 は一つのスロットを持っています。
5	*bVoltageSupport*	1		ACR3901U-S1 は 1.8V、3.0V と 5V のスロットサポート。
6	*dwProtocols*	4		ACR3901U-S1 は T=0 及び T=1 プロトコルをサポートできる。
10	*dwDefaultClock*	4		デフォルトの ICC クロック周波数は 4.8MHz です。
14	*dwMaximumClock*	4		最大の ICC クロック周波数は 4.8MHz です。
18	*bNumClockSupported*	1		クロック周波数の手動設定をサポートしていません。
19	*dwDataRate*	4		デフォルトの ICC I/O ボーレートは 12903 bps です。
23	*dwMaxDataRate*	4		ICC I/O サポートできる最大のボーレートは 600 kbps です。
27	*bNumDataRatesSupported*	1		ボーレートの手動設定をサポートできません。
28	*dwMaxIFSD*	4		ACR3901U-S1 は T=1 プロトコルでサポートできる最大の IFSD は 254 です。
32	*dwSynchProtocols*	4		ACR3901U-S1 は同期カードをサポートできません。
36	*dwMechanical*	4		ACR3901U-S1 は特別な機械的特性をサポートできません。
40	*dwFeatures*	4		ACR3901U-S1 は下記の特性を持っています: • パラメータに応じて、自動的に ICC のクロック周波数を変更します • 周波数と FI、DI パラメータに基づいて、自動的ボーレートを変更します • ACR3901U-S1 との TPDU レベル変更
44	*dwMaxCCIDMessageLength*	4		ACR3901U-S1 が受け入れられる最大なメッセージの長さは 271 バイトです。

オフセット	フィールド	大きさ	数値	説明
48	*bClassGetResponse*	1		TPDU レベル交換に影響しません。
49	*bClassEnvelope*	1		TPDU レベル交換に影響しません。
50	*wLCDLayout*	2		LCD なし。
52	*bPINSupport*	1		PIN 認証サポート。
53	*bMaxCCIDBusySlots*	1		唯一の 1 スロットを同時に忙しくなることができます。

6.2.1. CCID Bulk-OUT メッセージ

6.2.1.1. PC_to_RDR_IccPowerOn

このコマンドはスロットを活性化して、カードから ATR を返すために使われます。

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**62h**	
1	*dwLength*	4	00000000h	メッセージの余分なバイトのサイズ。
5	*bSlot*	1		このコマンドのスロット番号を識別します。
6	*bSeq*	1		コマンドのシーケンス番号。
7	*bPowerSelect*	1		ICC に印加される電圧: 00h = 自動電圧選択 01h = 5 V 02h = 3 V
8	*abRFU*	2		保留して将来使います。

このコマンドメッセージの応答は RDR_to_PC_DataBlock 応答メッセージです。返したデータはリセット応答（ATR）です。

6.2.1.2. PC_to_RDR_IccPowerOff

スロットの活性化をキャンセルする時、このコマンドを使います。

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**63h**	
1	*dwLength*	4	00000000h	メッセージの余分なバイトのサイズ。
5	*bSlot*	1		このコマンドのスロット番号を識別します。
6	*bSeq*	1		コマンドのシーケンス番号。
7	*abRFU*	3		保留して将来使います。

このメッセージの応答は *RDR_to_PC_Parameters* メッセージです。

6.2.1.3. PC_to_RDR_GetSlotStatus

現在のスロットの状態情報を取得する時に、このコマンドを使います。

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**65h**	
1	*dwLength*	4	00000000h	メッセージの余分なバイトのサイズ。
5	*bSlot*	1		このコマンドのスロット番号を識別します。
6	*bSeq*	1		コマンドのシーケンス番号。
7	*abRFU*	3		保留して将来使います。

このメッセージの応答 *RDR_to_PC_SlotStatus* 応答メッセージです。

6.2.1.4. PC_to_RDR_XfrBlock

ICC にデータブロックを転送する時にこのコマンドを使用します。

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**6Fh**	
1	*dwLength*	4		このメッセージの abData データフィールドのサイズ
5	*bSlot*	1		このコマンドのスロット番号を識別します。
6	*bSeq*	1		コマンドのシーケンス番号。

オフセット	フィールド	大きさ	数値	説明
7	*bBWI*	1		CCID ブロックを転送するタイムアウト時間を拡張する時に使用します。「この数にブロック待機時間を掛け」の期限が切れた後、CCID はブロックをタイムアウトします。
8	*wLevelParameter*	2		短い APDU レベル、RFU = 0000h 拡張 APDU レベル: 0000h - コマンド APDU がこのコマンドで開始および終了します。 0001h - このコマンドでコマンド APDU が開始され、次の PC_to_RDR_XfrBlock で続行されます。 0002h-abData フィールドは引き続きコマンド APDU を渡し、APDU コマンドを終了します。 0003h-abData フィールドは、コマンド APDU に続いて別のデータブロックを引き続き渡します。 0010h - 空の abData フィールド、次の RDR_to_PC_DataBlock は応答 APDU を引き続き渡します
10	*abData*	バイト配列		CCID に送信されたデータブロック。情報は ICC に「そのまま」送信されます（TPDU 交換レベル）。

このメッセージの応答は *RDR_to_PC_Parameters* メッセージです。

6.2.1.5. PC_to_RDR_GetParameters

スロットのパラメーターを取得する時にこのコマンドを使用します。

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**6Ch**	
1	*DwLength*	4	00000000h	メッセージの余分なバイトのサイズ。
5	*BSlot*	1		このコマンドのスロット番号を識別します。
6	*BSeq*	1		コマンドのシーケンス番号。
7	*AbRFU*	3		保留して将来使います。

このメッセージの応答は *RDR_to_PC_Parameters* メッセージです。

6.2.1.6. PC_to_RDR_ResetParameters

スロットのパラメーターをデフォルト値に戻す時ニコのコマンドを使用します。

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**6Dh**	
1	*DwLength*	4	00000000h	メッセージの余分なバイトのサイズ。
5	*BSlot*	1		このコマンドのスロット番号を識別します。
6	*BSeq*	1		コマンドのシーケンス番号。
7	*AbRFU*	3		保留して将来使います。

このメッセージの応答は *RDR_to_PC_Parameters* メッセージです。

6.2.1.7. PC_to_RDR_SetParameters

スロットのパラメーターを設置する時にこのコマンドを使用します。

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**61h**	
1	*dwLength*	4		メッセージの余分なバイトのサイズ。
5	*bSlot*	1		このコマンドのスロット番号を識別します。
6	*bSeq*	1		コマンドのシーケンス番号。
7	*bProtocolNum*	1		下記は指定されたプロトコルのデータ構造です。 00h = T=0 プロトコルの構造 01h = T=1 プロトコルの構造 下記の値を保留して将来使います: 80h = 2 線プロトコルの構造 81h = 3 線プロトコルの構造 82h = I2C プロトコル構造
8	*abRFU*	2		保留して将来使います。
10	*abProtocolDataStructure*	バイト配列		プロトコルのデータ構造

T=0 プロトコルのデータ構造（*dwLength*=00000005h）

オフセット	フィールド	大きさ	数値	説明
10	*bmFindexDindex*	1		B7-4 – FI – ISO/IEC 7816-3:1997 中のテーブル 7 をインデックスして、クロックレートの変換係数を選択します。 B3-0 – DI - ISO/IEC 7816-3:1997 中のテーブル 8 をインデックスして、ボーレートの変換係数を選択します
11	*bmTCCKST0*	1		B0 – 0b, B7-2 – 000000b B1 – 使用している約束（b1=0：直接；b1=1：逆） ***注:CCID がこのビットを無視します。***
12	*bGuardTimeT0*	1		2 文字間の余分な GuardTime。正常な GuardTime（12 ETU）に 0ー254 ETU を追加します。FFh と 00h が同じです。
13	*bWaitingIntegerT0*	1		T=0 の場合 WI が WWT を定義する時に使われます
14	*bClockStop*	1		ICC クロック停止サポート 00h = クロックを停止することは許可されていません 01h = クロック信号が低い時に停止されます 02h = クロック信号が高い時に停止されます 03h = クロック信号が低い時または高い時に停止されます

T=1 プロトコルのデータ構造（dwLength=00000007h）

オフセット	フィールド	大きさ	数値	説明
10	*bmFindexDindex*	1		B7-4 – FI – ISO/IEC 7816-3:1997 中のテーブル 7 をインデックスして、クロックレートの変換係数を選択します。 B3-0 – DI - ISO/IEC 7816-3:1997 中のテーブル 8 をインデックスして、ボーレートの変換係数を選択します
11	*BmTCCKST1*	1		B7-2 – 000100b B0 – チェックサムタイプ（b0=0：LRC； b0=1：CRC） B1 – 交換用（b1=0：直接；b1=1：逆）***注**：CCID がこのビットを無視します。*
12	*BGuardTimeT1*	1		余計な GuardTime（2 文字間の余分な GuardTime は 0－254etu）は FFh である場合、GuardTime を1etu 減らします。
13	*BwaitingIntegerT1*	1		B7-4 = BWI 値 0-9 有効 B3-0 = CWI 値 0-Fh 有効
14	*bClockStop*	1		ICC クロック停止サポート 00h = クロックを停止することは許可されていません 01h = クロック信号が低い時に停止されます 02h = クロック信号が高い時に停止されます 03h = クロック信号が低い時または高い時に停止されます
15	*bIFSC*	1		交渉された IFSC の大きさ
16	*bNadValue*	1	00h	NAD = 00h だけサポートできます

このメッセージの応答は *RDR_to_PC_Parameters* メッセージです。

6.2.1.8. PC_to_RDR_Escape

このコマンドは拡張機能にアクセスするために使用されます。

オフセット	データフィールド	大きさ	数値	説明
0	*bMessageType*	1	**6Bh**	-
1	*dwLength*	4	-	このメッセージの abData データフィールドのサイズ
5	*bSlot*	1	-	このコマンドのスロット番号を識別します。
6	*bSeq*	1	-	コマンドのシーケンス番号。
7	*abRFU*	3	-	保留して将来使います。
10	*abData*	バイト配列	-	CCID に送信されるデータブロック。

このメッセージの応答は *RDR_to_PC_Parameters* メッセージです。

6.2.2. CCID Bulk-IN メッセージ

6.2.2.1. RDR_to_PC_DataBlock

このコマンドは ACR3901U-S1 によって送信されて、*PC_to_RDR_IccPowerOn* と *PC_to_RDR_XfrBlock* メッセージに対しての応答です。

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**80h**	CCID によってデータブロックを送信しています。
1	*dwLength*	4		メッセージの余分なバイトのサイズ。
5	*bSlot*	1		Bulk-OUT メッセージ中の値と同じです。
6	*bSeq*	1		Bulk-OUT メッセージ中の値と同じです。
7	*bStatus*	1		CCID 規格（Rev 1.1）6.2.1 節で定義されたスロットステータスレジスタ
8	*bError*	1		**付録 A** CCID 規格（Rev 1.1）6.2.6 節で定義されたスロットステータスレジスタ
9	*bChainParameter*	1		拡張 APDU レベル： 0000h - コマンド APDU がこのコマンドで開始および終了します。 0001h - このコマンドでコマンド APDU が開始され、次の PC_to_RDR_XfrBlock で続行されます。 0002h-abData フィールドは引き続きコマンド APDU を渡し、APDU コマンドを終了します。 0003h-abData フィールドは、コマンド APDU に続いて別のデータブロックを引き続き渡します。 0010h - 空の abData フィールド、次の PC_to_RDR_XfrBlock は応答 APDU を引き続き渡します
10	*abData*	バイト配列		このデータフィールドは CCID から返したデータを含めています。

6.2.2.2. RDR_to_PC_SlotStatus

このコマンドは ACR3901U-S1 によって送信されて、*PC_to_RDR_IccPowerOff* と *PC_to_RDR_GetSlotStatus* メッセージに対しての応答です。

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**81h**	
1	*dwLength*	4	00000000h	メッセージの余分なバイトのサイズ。
5	*bSlot*	1		Bulk-OUT メッセージ中の値と同じです。
6	*bSeq*	1		Bulk-OUT メッセージ中の値と同じです。

オフセット	フィールド	大きさ	数値	説明
7	*bStatus*	1		CCID 規格（Rev 1.0）4.2.1 節で定義されたスロットステータスレジスタ
8	*bError*	1		**付録 A** CCID 規格（Rev 1.0）4.2.1 節で定義されたスロットステータスレジスタ
9	*bClockStatus*	1		数値： 00h = クロック動作中 01h = L 状態に止まっている 02h = H 状態に止まっている 03h = 不明な状態に止まっている 残された値を保留して将来使います。

6.2.2.3. RDR_to_PC_Parameters

このコマンドは ACR3901U-S1 によって送信されて、*PC_to_RDR_GetParameters*、*PC_to_RDR_ResetParameters* 及び *PC_to_RDR_SetParameters* メッセージに対しての応答です。

オフセット	フィールド	大きさ	数値	説明
0	*bMessageType*	1	**82h**	
1	*dwLength*	4		メッセージの余分なバイトのサイズ。
5	*bSlot*	1		Bulk-OUT メッセージ中の値と同じです。
6	*bSeq*	1		Bulk-OUT メッセージ中の値と同じです。
7	*bStatus*	1		CCID 規格（Rev 1.0）4.2.1 節で定義されたスロットステータスレジスタ
8	*bError*	1		**付録 A** CCID 規格（Rev 1.0）4.2.1 節で定義されたスロットステータスレジスタ
9	*bProtocolNum*	1		下記は指定されたプロトコルのデータ構造です。 00h = T=0 プロトコルの構造 01h = T=1 プロトコルの構造 下記の値を保留して将来使います： 80h = 2 線プロトコルの構造 81h = 3 線プロトコルの構造 82h = I2C プロトコル構造
10	*abProtocolDataStructure*	バイト配列		プロトコルのデータ構造は CCID 仕様（1.0 バージョン）5.2.3 節の説明を参照してください。

6.2.2.4. RDR_to_PC_Escape

このメッセージは ACR3901U-S1 によって送信されて、PC_to_RDR_Escape メッセージに対しての応答です。

オフセット	データフィールド	大きさ	数値	説明
0	*bMessageType*	1	**83h**	-
1	*dwLength*	4	-	メッセージの余分なバイトのサイズ。
5	*bSlot*	1	-	Bulk-OUT メッセージ中の値と同じです。
6	*bSeq*	1	-	Bulk-OUT メッセージ中の値と同じです。
7	*bStatus*	1	-	CCID 規格(Rev 1.0)4.2.1 節で定義されたスロットステータスレジスタ
8	*bError*	1	-	**付録 A** CCID 規格(Rev 1.0)4.2.1 節で定義されたスロットステータスレジスタ
9	*bRFU*	1	-	将来の使用のために予約 Reserved for Future Use
10	*abData*	バイト配列	-	CCID から送信されたでータ

7.0. ホストプログラミング API

7.1. 周辺デバイス制御

リーダーの周辺機器制御コマンドは Bluetooth mode でダイレクトコマンド（0x6B）を介して、もしくは USB モードで PC_to_RDR_Escape コマンドを介して実現されます。

7.1.1. シリアルナンバーを取得する（Get Serial Number）

このコマンドはリーダーのシリアル番号を取得する時に使われます。

コマンドフォーマット

オフセット	フィールド		大きさ	数値	説明
0	abData1	CommandCode	1	**02h**	Write Serial Number コマンドコード。
1		Len (CommandLength)	1	00h	データ中での余分のバイトの数
2		データ	0	-	-

応答データフォーマット

オフセット	フィールド		大きさ	数値	説明
0	abData2	ResponseCode	1	**82h**	Write Serial Number コマンド応答コード
1		Len (CommandLength)	1	-	データ中での余分のバイトの数
2		データ	10	-	10 バイトのシリアルナンバー

例:

リクエスト = 02 00

応答 = 82 0A FF FF FF FF FF FF FF FF FF FF

シリアルナンバー: FF FF FF FF FF FF FF FF FF FF

7.1.2. 乱数を取得する（Get Random Number Command）

このコマンドでリーダーの乱数（AES 暗号化アルゴリズムと認証マスタキーで暗号化された）を取得します。

注: *ブルートゥースモードのみ*

コマンドフォーマット

オフセット	フィールド		大きさ	数値	説明
0	*abData 1*	*CommandCode*	1	**03h**	Get Random Number コマンドコード。
1		*Len (CommandLength)*	1	00h	データ中での余分のバイトの数
2		*データ*	0	-	-

応答データフォーマット

オフセット	フィールド		大きさ	数値	説明
0	*abData2*	*ResponseCode*	1	**83h**	Get Random Number コマンド応答コード
1		*Len (CommandLength)*	1	10h	データ中での余分のバイトの数
2		*データ*	16	-	10 バイトのシリアルナンバー

例:

リクエスト = 03 00

応答 = 83 10 F2 8F B7 EF BA 43 C4 6B 85 D8 51 7B 84 08 C3 25

7.1.3. ファームウェアのバージョンを取得する（Get Firmware Version Command）

このコマンドはリーダーのファームウェアのバージョンを取得する時に使われます。

コマンドフォーマット

オフセット	フィールド		大きさ	数値	説明
0	abData1	CommandCode	1	**04h**	Get Firmware Version コマンドコード。
1		Len (CommandLength)	1	00h	データ中での余分のバイトの数
2		データ	0	-	-

応答データフォーマット

オフセット	フィールド		大きさ	数値	説明
0	abData2	ResponseCode	1	**84h**	Get Firmware Version コマンド応答コード
1		Len (CommandLength)	1	05h	データ中での余分のバイトの数
2		データ	5	-	“Vx.xx”フォーマットの 5 バイトの乱数

例:
リクエスト = 04 00
応答 = 84 05 56 31 2E 31 34
ファームバージョン (HEX) = 56 31 2E 31 34
ファームバージョン (ASCII) = “V1.14”

7.1.4. マスタキーをリセットする（Rewrite Master Key Command）

このコマンドはマスタキーをリセットする時に使われます。

コマンドフォーマット

オフセット	フィールド		大きさ	数値	説明
0	abData1	*CommandCode*	1	**07h**	Rewrite Master Key コマンドコード。
1		*Len (CommandLength)*	1	20h	データ中での余分のバイトの数
2		*データ*	32	-	元の顧客マスタ鍵で暗号化された乱数（KeyRstRnd [0:15]）+元の顧客マスタ鍵で暗号化された 16 バイトの新しい顧客マスタ鍵

応答データフォーマット

オフセット	フィールド		大きさ	数値	説明
0	abData2	*ResponseCode*	1	**87h**	Rewrite Master Key コマンド応答コード
1		*Len (CommandLength)*	1	01h	データ中での余分のバイトの数
2		*データ*	1	-	00h = 成功 01h = 失敗

例:

詳細は**顧客マスターキーをリセットするリクエスト(Customer Master Key Reset Request)**セクションを参照して下さい。

7.1.5. スリープモードオプション（Sleep Mode Option）

このコマンドはデバイスがスリープモードに入る前にの時間間隔を設定する時に使用されます。60 秒後に何も操作しない場合、デフォルトでは、リーダーがスリープモードに入ます。

コマンドフォーマット

オフセット	フィールド		大きさ	数値	説明
0	*abData1*	*CommandCode*	1	**0Dh**	Sleep Mode Option コマンドコード。
1		*Len (CommandLength)*	1	01h	データ中での余分のバイトの数
2		*データ*	1	-	00h = 60 秒（デフォルト） 01h = 90 秒 02h = 120 秒 03h = 180 秒 04h = 無効にする

応答データフォーマット

オフセット	フィールド		大きさ	数値	説明
0	*abData2*	*ResponseCode*	1	**8Dh**	Sleep Mode Option コマンド応答コード
1		*Len (CommandLength)*	1	01h	データ中での余分のバイトの数
2		*データ*	1	-	00h = 成功 01h = 失敗

例:
リクエストを 90s に設定する= 0D 01 01
応答= 8D 01 00

7.1.6. デバイスアドレスを取得する（Get Device Address）

このコマンドはデバイスのアドレスを取得するために使用されます。USB モードのみに使われます。

コマンドフォーマット

オフセット	フィールド		大きさ	数値	説明
0	*abData1*	*CommandCode*	1	**0Eh**	Get Device Address コマンドコード。
1		*Len (CommandLength)*	1	00h	データ中での余分のバイトの数
2		*データ*	0	-	

応答データフォーマット

オフセット	フィールド		大きさ	数値	説明
0	abData2	*ResponseCode*	1	**8Eh**	Get Device Address コマンド応答コード
1		*Len (CommandLength)*	1	06h	データ中での余分のバイトの数
2		*データ*	6	-	6 バイトのブルートゥースアドレス

例:

リクエスト = 0E 00

応答 = 8E 06 AA BB CC DD EE FF

デバイスアドレス: AA BB CC DD EE FF

7.1.7. Tx パワーを設定する（Set Tx Power）

ブルートゥースの通信パワーを設定するために使われます。

コマンドフォーマット

オフセット	フィールド		大きさ	数値	説明
0	*abData1*	*CommandCode*	1	**08h**	Set Tx Power コマンドコード。
1		*Len (CommandLength)*	1	01h	データ中での余分のバイトの数
2		*データ*	1	-	00h = -18 dBm (デフォルト) 距離：~4 米 01h = -12 dBm 距離：~7 米 02h = -6 dBm 距離：~16 米 03h = 0 dBm 距離：~25 米

応答データフォーマット

オフセット	フィールド		大きさ	数値	説明
0	*abData2*	*ResponseCode*	1	**88h**	Set Tx Power コマンド応答コード
1		*Len (CommandLength)*	1	01h	データ中での余分のバイトの数
2		*データ*	1	-	00h = 成功 01h = 失敗

例:
リクエスト = 08 01 00
応答 = 88 01 00

7.1.8. Tx パワーを読み取る（Read Tx Power Value）

ブルートゥースの通信パワーを読み取るために使われます。

コマンドフォーマット

オフセット	フィールド		大きさ	数値	説明
0	abData1	*CommandCode*	1	**09h**	Read Tx Power Value コマンドコード。
1		*Len (CommandLength)*	1	00h	データ中での余分のバイトの数
2		*データ*	0	-	-

応答データフォーマット

オフセット	フィールド		大きさ	数値	説明
0	abData2	*ResponseCode*	1	**89h**	Read Tx Power Value コマンドコード。
1		*Len (CommandLength)*	1	01h	データ中での余分のバイトの数
2		*データ*	1	-	00h = -18 dBm (デフォルト) 距離：~4 米 01h = -12 dBm 距離：~7 米 02h = -6 dBm 距離：~16 米 03h = 0 dBm 距離：~25 米

例：
リクエスト = 09 00
応答 = 89 01 00

7.1.9. 顧客マスターキーをリセットするリクエスト(Customer Master Key Reset Request)

このコマンドはリーダーが顧客マスターキーを生成する時に認証に必要な乱数をリクエストします。

コマンドフォーマット

オフセット	フィールド		大きさ	数値	説明
0	abData1	*CommandCode*	1	**0Fh**	Customer Master Key Reset Request コマンドコード。
1		*Len (CommandLength)*	1	00h	データ中での余分のバイトの数
2		*データ*	0	-	-

応答データフォーマット

オフセット	フィールド		大きさ	数値	説明
0	abData2	*ResponseCode*	1	**8Fh**	Rewrite Master Key 応答コード
1		*Len (CommandLength)*	1	10h	データ中での余分のバイトの数
2		*データ*	16	-	リーダーが生成した 16 バイトの乱数 (KeyRSTRnd[0:15])

例:

1. 乱数を生成する。
   入力:0F 00
   顧客マスターキーリセットコマンドの応答 = 8F 10 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11

2. AES128 CBC モードでは、オリジナルの顧客マスターキーを使用して乱数と新しい顧客マスターキーを暗号化します。これはアプリケーションの暗号化エンジンによって行われ、暗号化された結果は後で使用するために保存されます。
   乱数:11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11
   暗号化された乱数:F1 9F D2 D2 BA 1C 22 E1 6D C1 FE 1B 4B 43 D5 30
   新しい顧客マスターキー:11 22 33 44 55 66 77 88 11 22 33 44 55 66 77 88
   暗号化した新しい顧客マスターキー:27 E7 DA BE A6 1E 4B CD 29 F6 9B 36 25 05 8E 41

3. マスタキーをリセットする(**マスタキーをリセットする(Rewrite Master Key Command)**を参照して下さい)。
   マスタキーをリセットするコマンドリクエスト:07 20 F1 9F D2 D2 BA 1C 22 E1 6D C1 FE 1B 4B 43 D5 30 27 E7 DA BE A6 1E 4B CD 29 F6 9B 36 25 05 8E 41
   マスタキーを書き換えるコマンド:87 01 00

7.2. メモリカードのコマンドセット

7.2.1. メモリカード – 1、2、4、8 及び 16 kilobit I2C カード

7.2.1.1. SELECT_CARD_TYPE

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

注釈:SCardConnect( ) API *によって確立されたロジックなスマートカードリーダー通信後に使用しかできません。SCardConnect( ) API についての詳しい説明は PC/SC 基準を参照してください。*

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**カードタイプ**
FFh	A4h	00h	00h	01h	01h

応答データフォーマット(RDR_to_PC_DataBlock 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)

7.2.1.2. SELECT_PAGE_SIZE

このコマンドはスマートカードを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。

カードが削除されているか、または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。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU				
CLA	**INS**	**Zone Address**	**Byte Address**	**MEM_L**
FFh	01h	00h	00h	01h

その中:

Page size = 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
= 07h:128 バイトの書き込みページ

応答データフォーマット(RDR_to_PC_DataBlock 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)

7.2.1.3. READ_MEMORY_CARD

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU				
CLA	INS	Byte Address		MEM_L
		MSB	LSB	
FFh	B0h			

その中：

Byte Address メモリカードのメモリアドレス位置
MEM_L メモリカード中の読み取られていないデータの長さ

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

BYTE 1	…	…	BYTE N	SW1	SW2

その中：

BYTE x メモリカードから読み出されたデータ
SW1 SW2 = 90 00h（エラーなしの場合）

7.2.1.4. WRITE_MEMORY_CARD

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	INS	Byte Address		MEM_L	バイト 1	....	....	Byte n
		MSB	LSB					
FFh	D0h							

その中：

Byte Address メモリカードのメモリアドレス位置
MEM_L メモリに書き入れていないデータの長さ
Byte x メモリカードに書き入れていないデータ

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.2. メモリカード – 32、64、128、256、512 及び 1024 kilobit I2C カード

7.2.2.1. SELECT_CARD_TYPE

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

***注釈**:SCardConnect( ) API によって確立されたロジックなスマートカードリーダー通信後に使用しかできません。*

SCardConnect( ) API についての詳しい説明は PC/SC 基準を参照してください。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**カードタイプ**
FFh	A4h	00h	00h	01h	02h

応答データフォーマット(RDR_to_PC_DataBlock 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)

7.2.2.2. SELECT_PAGE_SIZE

このコマンドはスマートカードを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。

カードが削除されているか、または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。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**Page size**
FFh	01h	00h	00h	01h	

その中:

Data カードに送信されていない TPDU
Page size = 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
= 07h:128 バイトの書き込みページ

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)

7.2.2.3. READ_MEMORY_CARD

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU				
CLA	INS	Byte Address		MEM_L
		MSB	LSB	
FFh				

その中：

INS = B0h：32 kilobit、64 kilobit、128 kilobit、256 kilobit と 512 kilobit の IIC カード
= 1011 000*b：1024 kilobit IIC カード,
その中 * はアドレッシング 17 ビットの MSB を示している。

Byte Address メモリカードのメモリアドレス位置

MEM_L メモリカード中の読み取られていないデータの長さ

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

BYTE 1	…	…	BYTE N	SW1	SW2

その中：

BYTE x メモリカードから読み出されたデータ

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.2.4. WRITE_MEMORY_CARD

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	INS	Byte Address		MEM_L	バイト 1	....	....	Byte n
		MSB	LSB					
FFh								

その中：

INS = D0h 32 kilobit、64 kilobit、128 kilobit、256 kilobit と 512 kilobit の IIC カード
= 1101 000*b：1024 kilobit IIC カード,
その中 * はアドレッシング 17 ビットの MSB を示している。

Byte Address メモリカードのメモリアドレス位置

MEM_L メモリに書き入れていないデータの長さ

Byte x メモリカードに書き入れていないデータ

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.3. メモリカード – ATMEL AT88SC153

7.2.3.1. SELECT_CARD_TYPE

このコマンドはカードリーダーに挿入されて、選択したカードにパワーダウン/アップを実行します。同時にリセットを実行する時に使われます。それはまた、8 バイトのページの書き込みページサイズを選択する。

***注釈**:SCardConnect( ) API によって確立されたロジックなスマートカードリーダー通信後に使用しかできません。SCardConnect( ) API についての詳しい説明は PC/SC 基準を参照してください。*

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**卡片类型**
FFh	A4h	00h	00h	01h	03h

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.3.2. READ_MEMORY_CARD

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU				
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**
FFh		00h		

その中：

INS = B0h：00b を読み取る
= B1h：01b ゾーンを読み取る
= B2h：10b ゾーンを読み取る
= B3h：11b ゾーンを読み取る
= B4h： ヒューズを読み取る

Byte Address メモリカードのメモリアドレス位置

MEM_L メモリカード中の読み取られていないデータの長さ

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

BYTE 1	…	…	BYTE N	SW1	SW2

その中：

BYTE x メモリカードから読み出されたデータ

SW1、SW2 = 90 00h（エラーなしの場合）

7.2.3.3. WRITE_MEMORY_CARD

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	INS	P1	Byte Address	MEM_L	バイト 1	…	....	Byte n
FFh		00h						

その中：

INS = D0h：00b ゾーンを書く
= D1h：01b ゾーンを書く
= D2h：10b ゾーンを書く
= D3h：11b ゾーンを書く
= D4h：ヒューズを書き入れる

Byte Address メモリカードのメモリアドレス位置

MEM_L メモリに書き入れていないデータの長さ

MEM_D メモリカードに書き入れていないデータ

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1 、SW2 = 90 00h（エラーなしの場合）

7.2.3.4. VERIFY_PASSWORD

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU							
CLA	INS	P1	P2	Lc	Pw(0)	Pw(1)	Pw(2)
FFh	20h	00h		03h			

その中：

Pw(0),Pw(1),Pw(2) メモリカードに送信していないシークレットコード

P2 = 0000 00rpb
その中、2ビットの“rp”は比較されていないパスワードを示す
r = 0 ：パスワードを書く
r = 1 ：パスワードを読み取る
p：パスワード設定番号
rp = 01：セキュリティコード。

応答データフォーマット（RDR_to_PC_DataBlock中のabDataデータフィールド）

SW1	SW2 ErrorCnt
90h	

その中：

SW1 = 90h

SW2 (ErrorCnt) = エラー　カウンター。FFh は検証が正しいことを示している。00H 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。

7.2.3.5. INITIALIZE_AUTHENTICATION

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	**INS**	**P1**	**P2**	**Lc**	**Q(0)**	**Q(1)**	**…**	**Q(7)**
FFh	84h	00h	00h	08h				

その中：

Q(0),Q(1)…Q(7) ホスト挑戦、8 バイト

応答データフォーマット（RDR_to_PC_DataBlock中のabDataデータフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.3.6. VERIFY_AUTHENTICATION

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	**INS**	**P1**	**P2**	**Lc**	**Ch(0)**	**Ch(1)**	**…**	**Ch(7)**
FFh	82h	00h	00h	08h				

その中：

Ch(0),Ch(1)…Ch(7) ホスト挑戦、8 バイト

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.4. メモリカード – ATMEL AT88C1608

7.2.4.1. SELECT_CARD_TYPE

このコマンドはカードリーダーに挿入されて、選択したカードにパワーダウン/アップを実行します。同時にリセットを実行する時に使われます。それはまた、16 バイトのページの書き込みページサイズを選択する。
***注釈**:SCardConnect( ) API によって確立されたロジックなスマートカードリーダー通信後に使用しかできません。SCardConnect( ) API についての詳しい説明は PC/SC 基準を参照してください。*

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**カードタイプ**
FFh	A4h	00h	00h	01h	04h

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)

7.2.4.2. READ_MEMORY_CARD

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU				
CLA	**INS**	**Zone Address**	**Byte Address**	**MEM_L**
FFh				

その中:

INS = B0h:ユーザーゾーンを読み取る
= B1h:コンフィギュレーション·ゾーン
またはヒューズを読み取る

Zone Address = 0000 0A10A9A8b、その中 A10 はゾーンアドレスの MSB です
= ヒューズを読み取る必要がありません。

Byte Address = A7A6A5A4 A3A2A1A0b はメモリカードのメモリアドレス位置である
= 1000 0000b:ヒューズを読み取る

MEM_L メモリカード中の読み取られていないデータの長さ

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

BYTE 1	…	…	BYTE N	SW1	SW2

その中:

BYTE x メモリカードから読み出されたデータ

SW1、SW2 = 90 00h(エラーなしの場合)

7.2.4.3. WRITE_MEMORY_CARD

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	INS	Zone Address	Byte Address	MEM_L	バイト 1	…	…	Byte n
FFh								

その中：

INS = D0h：ユーザーゾーンを書く
= D1h：コンフィギュレーション·ゾーンまたはヒューズを書く
Zone Address = 0000 0A10A9A8b、その中 A10 はゾーンアドレスの MSB です
= ヒューズの書き込みと関係ない
Byte Address = A7A6A5A4 A3A2A1A0b はメモリカードのメモリアドレス位置である
= 1000 0000b：ヒューズを書く
MEM_L メモリに書き入れていないデータの長さ
Byte x メモリカードに書き入れていないデータ

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.4.4. VERIFY_PASSWORD

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	INS	P1	P2	Lc	数据			
FFh	20h	00h	00h	04h	RP	Pw(0)	Pw(1)	Pw(2)

その中：

Pw(0),Pw(1),Pw(2) メモリカードに送信していないシークレットコード
RP = 0000 rp2p1p0b
その中、4 ビットの“rp2p1p0”は比較されていないパスワードを示す
r = 0 ：パスワードを書く
r = 1 ：パスワードを読み取る
p2p1p0 ：パスワードセット番号。
(rp2p1p0 = 0111：セキュリティコード)

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

SW1	SW2 ErrorCnt
90h	

その中：

SW1 = 90h
SW2 (ErrorCnt) = エラー カウンター。FFh は検証が正しいことを示している。00H 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。

7.2.4.5. INITIALIZE_AUTHENTICATION

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	**INS**	**P1**	**P2**	**Lc**	**Q(0)**	**Q(1)**	**…**	**Q(7)**
FFh	84h	00h	00h	08h				

その中：

Byte Address メモリカードのメモリアドレス位置
Q(0),Q(1)…Q(7) ホスト挑戦、8 バイト

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.4.6. VERIFY_AUTHENTICATION

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	**INS**	**P1**	**P2**	**Lc**	**Q1(0)**	**Q1(1)**	**…**	**Q1(7)**
FFh	82h	00h	00h	08h				

その中：

Byte Address メモリカードのメモリアドレス位置
Q(0),Q(1)…Q(7) ホスト挑戦、8 バイト

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

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

7.2.5.1. SELECT_CARD_TYPE

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

***注釈**:SCardConnect( ) API によって確立されたロジックなスマートカードリーダー通信後に使用しかできません。SCardConnect( ) API についての詳しい説明は PC/SC 基準を参照してください。*

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**カードタイプ**
FFh	A4h	00h	00h	01h	05h

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)

7.2.5.2. READ_MEMORY_CARD

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU				
CLA	**INS**	**Byte Address**		**MEM_L**
		MSB	**LSB**	
FFh	B0h			

その中:

MSB Byte Address = 0000 00A9A8b はメモリカードのメモリアドレス位置である
LSB Byte Address = A7A6A5A4 A3A2A1A0b はメモリカードのメモリアドレス位置である
MEM_L メモリカード中の読み取られていないデータの長さ

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

BYTE 1	…	…	BYTE N	SW1	SW2

その中:

BYTE x メモリカードから読み出されたデータ
SW1 SW2 = 90 00h(エラーなしの場合)

7.2.5.3. READ_PRESENTATION_ERROR_COUNTER_MEMORY_CARD (SLE4428 及び SLE5528)

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。

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU				
CLA	**INS**	**P1**	**P2**	**MEM_L**
FFh	B1h	00h	00h	03h

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

ERRCNT	DUMMY 1	DUMMY 2	SW1	SW2

その中：

ERRCNT エラー カウンター。FFh は最後の検証が正しいことを示している。00h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）。他の値は最後の認証が失敗したことを示している。

DUMMY カードから読み出された 2 バイトのダミーデータ

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.5.4. READ_PROTECTION_BIT

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU				
CLA	**INS**	**Byte Address**		**MEM_L**
		MSB	**LSB**	
FFh	B2h			

その中：

MSB Byte Address = 0000 00A_9A_8b はメモリカードのメモリアドレス位置である

LSB Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b はメモリカードのメモリアドレス位置である

MEM_L カードから読み出される保護ビットの長さは 8 ビットの倍数です。最大数値は32です。MEM_L = 1 + INT((ビットのナンバー - 1)/8)

例えば、メモリ 0010H から始まる 8 保護ビットを読み取るために、以下の擬似 APDU を発行する必要がある：

FF B2 00 10 01h

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

PROT 1	…	…	PROT L	SW1	SW2

その中：

PROT y 保護ビットが含めているバイト

SW1、SW2 = 90 00h（エラーなしの場合）

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 の保護ビットです。

‘0’バイトが書き込み保護されている

‘1’バイトは書き込むことができる

7.2.5.5. WRITE_MEMORY_CARD

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	INS	Byte Address		MEM_L	バイト 1	....	....	Byte N
		MSB	LSB					
FFh	D0h							

その中：

MSB Byte Address = 0000 00A9A8b はメモリカードのメモリアドレス位置である
LSB Byte Address = A7A6A5A4 A3A2A1A0b はメモリカードのメモリアドレス位置である
MEM_L メモリに書き入れていないデータの長さ
Byte x メモリカードに書き入れていないデータ

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.5.6. WRITE_PROTECTION_MEMORY_CARD

コマンドで指定された各バイトは、内部でカードに指定されたアドレス中のデータと比べます。
一致した場合、対応している保護ビットが不可逆的に“0”にプログラムされている。
コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	INS	Byte Address		MEM_L	バイト 1	....	....	Byte N
		MSB	LSB					
FFh	D1h							

その中：

MSB Byte Address = 0000 00A9A8b はメモリカードのメモリアドレス位置である
LSB Byte Address = A7A6A5A4 A3A2A1A0b はメモリカードのメモリアドレス位置である
MEM_L メモリに書き入れていないデータの長さ
Byte x バイト値がバイトアドレスから始まるカード内のデータと比較される。BYTE 1 と Byte Address 中のデータを比べる；BYTE N と（Byte Address + N -1）中のデータが比べる。

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.5.7. PRESENT_CODE_MEMORY_CARD (SLE4428 及び SLE5528)

SLE4428 と SLE5528 に書き込む操作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する。

以下の操作を実行する:

1. 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。
2. 指定されたシークレットコードをカードに提出する。
3. プレゼンテーションエラーカウンタを消去しようとする。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU						
CLA	INS	P1	P2	MEM_L	CODE	
					バイト 1	バイト 2
FFh	20h	00h	00h	02h		

その中:

CODE 2 バイトのシークレットコード(PIN)

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

SW1	SW2 ErrorCnt
90h	

その中:

SW1 = 90h

SW2 (ErrorCnt) = エラー カウンター。FFh は認証が成功したことを示している。00h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。

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

7.2.6.1. SELECT_CARD_TYPE

このコマンドはカードリーダーに挿入されて、選択したカードにパワーダウン/アップを実行します。同時にリセットを実行する時に使われます。
注釈:SCardConnect( ) API によって確立されたロジックなスマートカードリーダー通信後に使用しかできません。SCardConnect( ) API についての詳しい説明は PC/SC 基準を参照してください。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**カードタイプ**
FFh	A4h	00h	00h	01h	06h

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)

7.2.6.2. READ_MEMORY_CARD

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU				
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**
FFh	B0h	00h		

その中:

Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b はメモリカードのメモリアドレス位置である
MEM_L メモリカード中の読み取られていないデータの長さ

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

BYTE 1	…	…	BYTE N	SW1	SW2

その中:

BYTE x メモリカードから読み出されたデータ
SW1、SW2 = 90 00h(エラーなしの場合)

7.2.6.3. READ_PRESENTATION_ERROR_COUNTER_MEMORY_CARD (SLE 4442 と SLE 5542)

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。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU				
CLA	**INS**	**P1**	**P2**	**MEM_L**
FFh	B1h	00h	00h	04h

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

ERRCNT	DUMMY 1	DUMMY 2	DUMMY 3	SW1	SW2

その中：

ERRCNT エラー カウンター。07h は最後の検証が正しいことを示している。 00H 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）。他の値は最後の認証が失敗したことを示している。

DUMMY カードから読み取られた3バイトのダミーデータ

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.6.4. READ_PROTECTION_BITS

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。

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU				
CLA	**INS**	**P1**	**P2**	**MEM_L**
FFh	B2h	00h	00h	04h

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

PROT 1	PROT 2	PROT 3	PROT 4	SW1	SW2

その中：

PROT y 保護ビットが含めているバイト

SW1 SW2 = 90 00h（エラーなしの場合）

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 の保護ビットです。

‘0’バイトが書き込み保護されている

‘1’バイトは書き込むことができる

7.2.6.5. WRITE_MEMORY_CARD

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**	**バイト 1**	**....**	**....**	**Byte N**
FFh	D0h	00h						

その中：

Byte Address = $A_7A_6A_5A_4\ A_3A_2A_1A_0$b はメモリカードのメモリアドレス位置である

MEM_L メモリに書き入れていないデータの長さ

Byte x メモリカードに書き入れていないデータ

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.6.6. WRITE_PROTECTION_MEMORY_CARD

コマンドで指定された各バイトは内部でカードに指定されたアドレス中のデータと比べる。一致した場合、対応している保護ビットが不可逆的に“0”にプログラムされている。

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	INS	P1	Byte Address	MEM_L	バイト 1	....	....	Byte N
FFh	D1h	00h						

その中：

Byte Address = 000A_4 $A_3A_2A_1A_0$b (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 メモリに書き入れていないデータの長さ
Bytex バイト値がバイトアドレスから始まるカード内のデータと比較される。BYTE 1 と Byte Address 中のデータを比べる；BYTE N と（Byte Address + N -1）中のデータが比べる。

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.6.7. PRESENT_CODE_MEMORY_CARD（SLE 4442 及び SLE 5542）

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。
以下の操作を実行する：

1. 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。
2. 指定されたシークレットコードをカードに提出する。
3. プレゼンテーションエラーカウンタを消去しようとする。

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU							
CLA	INS	P1	P2	MEM_L	CODE		
					バイト 1	バイト 2	バイト 3
FFh	20h	00h	00h	03h			

その中：

CODE 3 バイトのシークレットコード（PIN）

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

SW1	SW2 ErrorCnt
90h	

その中：

SW1 = 90h

SW2 (ErrorCnt) = エラー　カウンター。07h は検証が正しいことを示している。00H 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。

7.2.6.8. CHANGE_CODE_MEMORY_CARD（SLE 4442 と SLE 5542）

指定されたデータを新しいシークレットコードとして、カードに書き入れる時に、このコマンドを使用します。
PRESENT_CODE コマンドでカードに現在のシークレットコードを提出してから、このコマンドを実行します。

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU							
CLA	INS	P1	P2	MEM_L	CODE		
					バイト 1	バイト 2	バイト 3
FFh	D2h	00h	01h	03h			

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.7. メモリカード – SLE 4406/SLE 4436/SLE 5536/SLE 6636

7.2.7.1. SELECT_CARD_TYPE

このコマンドはカードリーダーに挿入されて、選択し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。

***注釈**:SCardConnect( ) API によって確立されたロジックなスマートカードリーダー通信後に使用しかできません。SCardConnect( ) API についての詳しい説明は PC/SC 基準を参照してください。*

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**カードタイプ**
FFh	A4h	00h	00h	01h	07h

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)

7.2.7.2. READ_MEMORY_CARD

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU				
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**
FFh	B0h	00h		

その中:

Byte Address メモリカードのメモリアドレス位置
MEM_L メモリに書き入れていないデータの長さ

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

BYTE 1	…	…	BYTE N	SW1	SW2

その中:

BYTE x メモリカードから読み出されたデータ
SW1 SW2 = 90 00h(エラーなしの場合)

7.2.7.3. WRITE_ONE_BYTE_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。

4 つの異なる書き込みモードは、このカードタイプで使用できる。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ます。

a. **書き込むコマンド**
   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、カードに個人化情報とカウンタ値を書き入れます。
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。この書き込みモードはカードにカウンタ値を更新するためにのみ使用することができる。

c. **書き入れる時、バックアップ機能を有効にする**（SLE 4436、SLE5536 及び 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ます。

d. **バックアップ機能の'Write with carry'コマンドを有効にする**（SLE 4436、SLE5536 及び 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。この書き込みモードはカードにカウンタ値を更新するためにのみ使用することができる。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バックアップモードは、書き込み動作に有効または無効されます。

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU						
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**	**MODE**	**BYTE**
FFh	D0h	00h		02h		

その中：

Byte Address メモリカードのメモリアドレス位置

MODE ライトモードとバックアップオプションを指定する
00h:書き入れる
01h:Write with carry
02h：書き入れる時、バックアップ機能を有効にする（SLE 4436、SLE5536 及び SLE6636 のみ）
03h：バックアップ機能の'Write with carry'コマンドを有効にする（SLE 4436、SLE5536 及び SLE6636 のみ）

BYTE カードに書き入れていないバイトの値

応答データフォーマット（*RDR_to_PC_DataBlock* 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.7.4. PRESENT_CODE_MEMORY_CARD

メモリカードにシークレットコードを提出して、カードの個人化モードを有効する時に、コマンドが以下の操作を実行する:

1. 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。
2. 指定されたシークレットコードをカードに提出する。

シークレットコードを提出すると、ACR3901U-S1 はプレゼンテーションカウンタを消去しようとしません。アプリケーションソフトウェアによって、独立な'Write with carry'コマンドを介して行われなければなりません。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU								
CLA	INS	P1	P2	MEM_L	CODE			
					ADDR	バイト 1	バイト 2	バイト 3
FFh	20h	00h	00h	04h	09h			

その中:

ADDR カードプレゼンテーションカウンタのビットアドレス
CODE 3 バイトのシークレットコード(PIN)

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)

7.2.7.5. AUTHENTICATE_MEMORY_CARD(SLE 4436、SLE 5536 及び SLE 6636)

SLE5536 または SLE6636 カードからカード認証証明書を読むには、ACR3901U-S1 は、次のアクションを実行します。

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. コマンドで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

認証プロセスが二段階で実行される:
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

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU												
CLA	INS	P1	P2	MEM_L	CODE							
					KEY	CLK_CNT	バイト 1	バイト 2	……	バイト 5	バイト 6
FFh	84h	00h	00h	08h							

その中:

KEY 認証証明書を計算するためのキー:
00h:Key 1、暗号ブロック連鎖付いていない
01h:Key 2、暗号ブロック連鎖付いていない
80h:key 1 暗号ブロック連鎖付き (SLE5536 と SLE6636 のみ)
81h:key 2 暗号ブロック連鎖付き (SLE5536 と SLE6636 のみ)

CLK_CNT CLK のパルス数は、認証証明書の各ビットの計算のためにカードに供給されます。標準値は 160(A0h)です。

BYTE 1...6 カードの乱数データ

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

SW1	SW2
61h	02h

その中:

SW1 SW2 = 61 02h (エラーなしの場合)、2 バイトの認証データが準備できていることを表す。
GET_RESPONSE コマンドによって、認証データを入手する。

ステップ 2:認証データを取り戻す(Get_Response)
コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU				
CLA	INS	P1	P2	MEM_L
FFh	C0h	00h	00h	02h

応答データフォーマット(RDR_to_PC_DataBlock 中の abData データフィールド)

CERT		SW1	SW2

その中:

CERT カードによって計算された認証データの 16 ビット。BYTE 1 の LSB はカードから読み出された始めの認証ビット。

SW1 SW2 = 90 00h(エラーなしの場合)

7.2.8. メモリカード – SLE 4404

7.2.8.1. SELECT_CARD_TYPE

このコマンドはカードリーダーに挿入されて、選択し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。

注釈:*SCardConnect( ) API によって確立されたロジックなスマートカードリーダー通信後に使用しかできません。SCardConnect( ) API についての詳しい説明は PC/SC 基準を参照してください。*

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**カードタイプ**
FFh	A4h	00h	00h	01	08h

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

SW1	SW2

その中:

SW1、SW2 = 90 00h(エラーなしの場合)

7.2.8.2. READ_MEMORY_CARD

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU				
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**
FFh	B0h	00h		

その中:

Byte Address メモリカードのメモリアドレス位置
MEM_L メモリカード中の読み取られていないデータの長さ

応答データフォーマット(*RDR_to_PC_DataBlock* 中の abData データフィールド)

BYTE 1	…	…	BYTE N	SW1	SW2

その中:

BYTE x メモリカードから読み出されたデータ
SW1 SW2 = 90 00h(エラーなしの場合)

7.2.8.3. 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。
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。

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	INS	P1	Byte Address	MEM_L	バイト 1	...	...	バイト N
FFh	D0h	00h						

その中：

Byte Address メモリカードのメモリアドレス位置
MEM_L メモリに書き入れていないデータの長さ
BYTE カードに書き入れていないバイトの値

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.8.4. 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。

エラーカウンターまたはユーザーゾーンを消去する時、**VERIFY_USER_CODE** パートの説明のように、VERIFY_USER_CODE コマンドを使ってください。

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU				
CLA	INS	P1	Byte Address	MEM_L
FFh	D2h	00h		00h

その中：

Byte Address ＝スクラッチパッドのメモリバイトアドレス位置
標準値は 02h です

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.8.5. VERIFY_USER_CODE

このコマンドはユーザーパスワード(2 バイト)を挿入されているカードに提出します。
ユーザーパスワードはカードのストレージのアクセス権限を有効するためです。
以下のアクションが実行される:

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を消去ことができる。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU						
CLA	INS	Error Counter LEN	Byte Address	MEM_L	CODE	
					バイト 1	バイト 2
FFh	20h	04h	08h	02h		

その中:

Error Counter LEN プレゼンテーションエラーカウンタの長さ、ビットの単位です。
Byte Address カード中のキーのバイトアドレス
CODE 2 バイトのユーザーシークレットコード

応答データフォーマット(RDR_to_PC_DataBlock 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)
= 63 00h(これ以上の再試行がない場合)

***注釈**:SW1 SW2 = 9000h 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。*

7.2.8.6. 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。メモリ エラー カウンター中のデータが消去されないことを注意してください。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU								
CLA	INS	Error Counter LEN	Byte Address	MEM_L	CODE			
					バイト 1	バイト 2	バイト 3	バイト 4
FFh	20h	40h	28h	04h				

応答データフォーマット(RDR_to_PC_DataBlock 中の abData データフィールド)

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）
= 63 00h（これ以上の再試行がない場合）

***注釈**：SW1 SW2 = 9000 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、アプリケーションゾーン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。*

7.2.9. メモリカード – AT88SC101/AT88SC102/AT88SC1003

7.2.9.1. SELECT_CARD_TYPE

このコマンドはカードリーダーに挿入されて、選択し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。
***注釈**:SCardConnect( ) API によって確立されたロジックなスマートカードリーダー通信後に使用しかできません。SCardConnect( ) API についての詳しい説明は PC/SC 基準を参照してください。*

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**カードタイプ**
FFh	A4h	00h	00h	01h	09h

応答データフォーマット(RDR_to_PC_DataBlock 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)

7.2.9.2. READ_MEMORY_CARD

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU				
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**
FFh	B0h	00h		

その中:

Byte Address メモリカードのメモリアドレス位置
MEM_L メモリカード中の読み取られていないデータの長さ

応答データフォーマット(RDR_to_PC_DataBlock 中の abData データフィールド)

BYTE 1	…	…	BYTE N	SW1	SW2

その中:

BYTE x メモリカードから読み出されたデータ
SW1 SW2 = 90 00h(エラーなしの場合)

7.2.9.3. 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。
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。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU								
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**	**バイト 1**	**....**	**....**	**Byte N**
FFh	D0h	00h						

その中：

Byte Address メモリカードのメモリアドレス位置
MEM_L メモリに書き入れていないデータの長さ
BYTE カードに書き入れていないバイトの値

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.9.4. ERASE_NON_APPLICATION_ZONE

このコマンドがアプリケーションゾーンにストアーされていない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：
ERASE_APPLICATION_ZONE_WITH_ERASE は ERASE_APPLICATION_ZONE_WITH_ERASE コマンドを定義している。
ERASE_APPLICATION_ZONE_WITH_WRITE_AND_ERASE は ERASE_APPLICATION_ZONE_WITH_WRITE_AND_ERASE コマンドを定義している。
VERIFY_SECURITY_CODE は VERIFY_SECURITY_CODE コマンドを定義している。
コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU				
CLA	INS	P1	Byte Address	MEM_L
FFh	D2h	00h		00h

その中：

Byte Address 消去していないワードのメモリバイトアドレスの場所

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

7.2.9.5. ERASE_APPLICATION_ZONE_WITH_ERASE

このコマンドは以下の状況に適用される：

1. AT88SC101：擦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。
2. AT88SC102：アプリケーションゾーン 1 のデータを消去する。
3. 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。
4. AT88SC1003：アプリケーションゾーン 1 のデータを消去する。
5. 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。
6. 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。
   a. 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。

コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU									
CLA	INS	Error Counter LEN	Byte Address	MEM_L	CODE				
					バイト 1	バイト 2	…	…	Byte N
FFh	20h	00h							

その中：

Error Counter LEN プレゼンテーションエラーカウンタの長さ、ビットの単位です。
ずっと 00h であるべき。

Byte Address カード中のアプリケーションゾーンのアドレス。
精確な数値が下のチャートを参照してください。

	Byte Address	LEN
AT88SC101：擦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
AT88SC1003：擦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

MEM_L 消去キーの長さ。
CODE N バイトの消去キー

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

__注釈__:SW1 SW2 = 9000h 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g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。

7.2.9.6. ERASE_APPLICATION_ZONE_WITH_WRITE_AND_ERASE

このコマンドは以下の状況に適用される：

1. 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。
2. 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。
3. 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 ヒューズが損害されていなくて、“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。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU								
CLA	INS	Error Counter LEN	Byte Address	MEM_L	CODE			
					バイト 1	バイト 2	バイト 3	バイト 4
FFh	20h	80h		04h				

その中:

Error Counter LEN プレゼンテーションエラーカウンタの長さ、ビットの単位です。
ずっと 80h であるべき。

Byte Address カード中のアプリケーションゾーンのアドレス。

	Byte Address
AT88SC101	96h
AT88SC102	9Ch
AT88SC1003	5Ch

CODE 4 バイトの消去キー

応答データフォーマット(RDR_to_PC_DataBlock 中の abData データフィールド)

SW1	SW2

その中:

SW1 SW2 = 90 00h(エラーなしの場合)
= 63 00h(これ以上の再試行がない場合)

***注釈**:SW1 SW2 = 9000h 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g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。*

7.2.9.7. 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を消去ことができる。

コマンドフォーマット(PC_to_RDR_XfrBlock 中の abData データフィールド)

Pseudo-APDU						
CLA	INS	Error Counter LEN	Byte Address	MEM_L	CODE	
					バイト 1	バイト 2
FFh	20h	08h	0Ah	02h		

その中:

Error Counter LEN プレゼンテーションエラーカウンタの長さ、ビットの単位です。
Byte Address カード中のキーのバイトアドレス
CODE 2 バイトのセキュリティコード

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1、SW2 = 90 00h（エラーなしの場合）
Fe= 63 00h（これ以上の再試行がない場合）

***注釈**:SW1 SW2 = 9000h を受信した後で、VERIFY_USER_CODE が正しいかどうかを確認するために、セキュリティコードの試みるカウンター（SCAC）を再度読み取ることができます。*
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。

7.2.9.8. BLOWN_FUSE

このコマンドは挿入されているカードのヒューズを変更する時に使われる。
ヒューズは EC_EN のヒューズ、EC2EN のヒューズ、メーカーのヒューズまたは発行者のヒューズ可能です。
***注釈**:ヒューズを変更することは不可逆過程です。*
コマンドフォーマット（PC_to_RDR_XfrBlock 中の abData データフィールド）

Pseudo-APDU								
CLA	INS	Error Counter LEN	Byte Address	MEM_L	CODE			
					Fuse Bit Addr (High)	Fuse Bit Addr (Low)	State of FUS Pin	State of RST Pin
FFh	05h	00h	00h	04h			01h	00h または 01h

その中：

Fuse Bit Addr (2 バイト) ヒューズのビットアドレス。精確な数値が下のチャートを参照してください。
State of FUS Pin FUS pin の状態。ずっと 01h であるべき。
State of RST Pin 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。

		Fuse Bit Addr (High)	Fuse Bit Addr (Low)	State of RST Pin
AT88SC101	メーカーのヒューズ	05h	80h	01h
	EC2EN のヒューズ	05h	C9h	01h
	発行者のヒューズ	05h	E0h	01h
AT88SC102	メーカーのヒューズ	05h	B0h	01h
	EC2EN のヒューズ	05h	F9h	01h
	発行者のヒューズ	06h	10h	01h
AT88SC1003	メーカーのヒューズ	03h	F8h	00h
	EC2EN のヒューズ	03h	FCh	00h
	発行者のヒューズ	03h	E0h	00h

応答データフォーマット（RDR_to_PC_DataBlock 中の abData データフィールド）

SW1	SW2

その中：

SW1 SW2 = 90 00h（エラーなしの場合）

付録 A. エラーコード

下記のテーブルは ACR3901U-S1 が返す可能なエラーコードをまとめています:

エラーコード	説明
01h	チェックサムが無効
02h	データの長さが無効
03h	コマンドフォーマット無効
04h	コマンド無効/未知のコマンド ID
05h	操作エラー
06h	認証必要/認証エラー
07h	電源不足
08h	認証失敗

表12 :エラーコード