SONY

FeliCa

SDK for NFC
ユーザーズマニュアル

FeliCaライブラリ編
(Starter Kit版)

Contactless IC Card
Software Development Kit

Sony Corporation Confidential

Version 1.2
No. M735-J01-20

はじめに

本文書は、SDK for NFC に含まれる FeliCa アクセスライブラリのうち、ポーリング（ターゲット捕捉）および FALP に関する関数の仕様について説明しています。

（このページは白紙です。）

目次

1. FeliCa ライブラリ 1

1.1. 関数一覧 1

1.2. 構造体一覧 3

1.2.1. structure_reader_writer_mode 3

1.2.2. structure_card_information 3

1.2.3. structure_device_information 3

1.2.4. structure_polling 4

1.3. FALP 機能 5

1.3.1. FALP 通信 5

1.3.2. FALP 関連 API 5

1.3.3. API 一覧と動作モード 6

2. API 仕様 7

2.1. ライブラリ管理関係の API 7

2.1.1. initialize_library 7

2.1.2. dispose_library 8

2.1.3. get_last_error_type 8

2.2. リーダ／ライタ制御関係の API 9

2.2.1. open_reader_writer 9

2.2.2. open_reader_writer_auto 9

2.2.3. close_reader_writer 11

2.2.4. transaction_lock 11

2.2.5. transaction_unlock 12

2.2.6. get_device_information 12

2.3. ライブラリの各種設定を行う API 14

2.3.1. set_time_out 14

2.3.2. get_time_out 14

2.3.3. set_lock_timeout 15

2.3.4. get_lock_timeout 15

2.3.5. set_polling_timeout 16

2.3.6. get_polling_timeout 16

2.4. ポーリングに関する API 17

2.4.1. polling_and_get_card_information 17

2.4.2. get_last_card_information 18

2.5. FALP に関する API 19
2.5.1. falp_open 19
2.5.2. falp_close 20
2.5.3. falp_connect 21
2.5.4. falp_listen 23
2.5.5. falp_stop_listen 24
2.5.6. set_falp_target_callback_parameters 25
2.5.7. falp_target_disconnect 26
2.5.8. falp_shutdown 27
2.5.9. falp_recv 28
2.5.10. falp_send 29
2.5.11. falp_wait_event 30
2.5.12. falp_dump_log_to_file 31
2.5.13. falp_stop_logging 31
2.5.14. falp_get_last_error_type 32
3. エラータイプ 33
3.1. get_last_error_type で取得されるエラー 33
3.2. falp_get_last_error_type()で取得されるエラー 34

1. FeliCa ライブラリ

1.1. 関数一覧

■ライブラリの管理に関するAPI

関数名	説明
initialize_library	ライブラリの初期化
dispose_library	ライブラリの解放
get_last_error_type	エラー情報の取得

■リーダ／ライタの制御に関するAPI

関数名	説明
open_reader_writer	リーダ／ライタのオープン
open_reader_writer_auto	リーダ／ライタの自動認識とオープン
close_reader_writer	リーダ／ライタのクローズ
transaction_lock	トランザクション排他ロックの設定
transaction_unlock	トランザクション排他ロックの解除
get_device_information	リーダ／ライタの種別と接続方式の取得

■ライブラリの各種設定を行うAPI

関数名	説明
set_lock_timeout	アクセス権獲得タイムアウト値の設定
get_lock_timeout	アクセス権獲得タイムアウト値の取得
set_polling_timeout	ポーリング専用タイムアウト値の設定
get_polling_timeout	ポーリング専用タイムアウト値の取得

■ポーリングに関するAPI

関数名	説明
polling_and_get_card_information	ポーリングとカード情報の取得（Polling コマンド）
get_last_card_information	カード情報の取得

■FALPに関するAPI

関数名	説明
falp_open	FALPモードへ移行
falp_close	FALPモードの終了
falp_connect	FALPイニシエータ転送中モードへ移行
falp_listen	ターゲット接続待ち開始
falp_stop_listen	ターゲット接続待ち停止
set_falp_target_callback_parameters	ターゲット接続時のメッセージ設定

falp_target_disconnect	FALPターゲットモードの終了
falp_shutdown	FALP転送終了モード、FALPターゲット転送終了モードへ移行
falp_recv	受信データ取得
falp_send	データ送信
falp_wait_event	FALPイベント取得
falp_dump_log_to_file	ロギング開始
falp_stop_logging	ロギング停止
falp_get_last_error_type	エラー情報取得

1.2. 構造体一覧

1.2.1. structure_reader_writer_mode

型（属性）	名称	意味
char*	port_name	ポートの名前
unsigned long	baud_rate	ボーレート 0x00: 固定
unsigned char	encryption_mode	暗号化モード 0x01: 固定
unsigned char*	kar	dummy
unsigned char*	kbr	dummy

1.2.2. structure_card_information

型（属性）	名称	意味
unsigned char*	card_idm	カードの IDm の格納領域（8バイト）
unsigned char*	card_pmm	カードの PMm の格納領域（8バイト）

1.2.3. structure_device_information

型（属性）	名称	意味
unsigned char	device_info_type	NFC ポート／パソリ
unsigned char	device_info_connect	NFC ポート／パソリ接続方式 0x00: 内蔵 0x01: 外付け

1.2.4. structure_polling

型（属性）	名称	意味
unsigned char* unsigned char	system_code time_slot	システムコード（2 バイト） タイムスロット 0x00, 0x01, 0x03, 0x07, 0x0f のいずれか

※ タイムスロットは、カードを複数枚同時に検出するための値です。複数枚のカードを同時検出する必要がない場合は、0x00を指定してください。

1.3.FALP 機能

1.3.1. FALP 通信

モバイル FeliCa OS Version 2.0 を搭載して FALP を実装している機器と、FALP 通信することができます。
FALP 通信を使用すると、双方向にデータの送受信ができます。
FALP の上で動作する各種プロトコルについては、それぞれのプロトコル策定者にお問い合わせください。

1.3.2. FALP 関連 API

FALP 通信開始メッセージを送信する側を「イニシエータ」、FALP 通信開始メッセージを受信する側を「ターゲット」と呼びます。また、イニシエータで通信する際の動作モードを「FALP モード」、ターゲットで通信する際の動作モードを「FALP ターゲットモード」とよびます。
FeliCa ライブラリは、以下 3 状態を有しています(各状態に、サブ状態があります)。

- 通常モード（ポーリングや読み書きを行う動作モード）
- FALP モード
- FALP ターゲットモード

FeliCa ライブラリは、上位アプリケーションからの要求・対向機器からの通信要求により、状態が変化します。
FeliCa ライブラリの動作モード遷移図を図 1-1に示します。

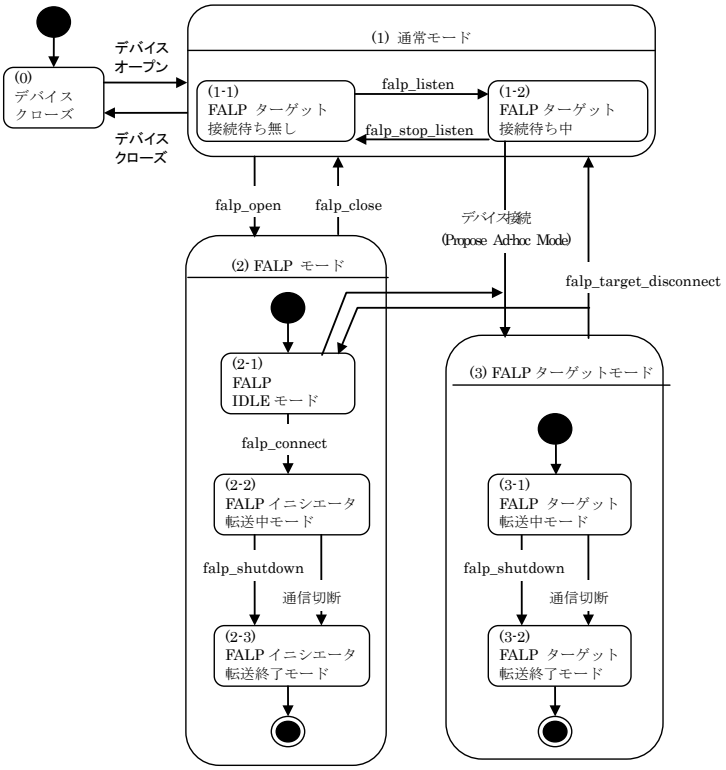

図 1-1:FeliCa ライブラリの内部状態遷移

1.3.3. API 一覧と動作モード

FALP 関連の API 関数と各動作モードの使用可能（○）・不可能（×）の関係を以下に示します。

表 1-1の動作モード番号は、図 1-1に出ているモード番号を示しています。

表 1-1:FALP 関連 API と動作モードごとの呼び出し可否

	(1) 通常モード		(2) FALPモード			(3) FALP ターゲットモード		説明
関数名	(1-1)	(1-2)	(2-1)	(2-2)	(2-3)	(3-1)	(3-2)	
open_reader_writer open_reader_writer_auto	×	×	×	×	×	×	×	デバイスクローズから通常モードへ移行
falp_open	○	○	×	×	×	×	×	FALPモードへ移行
falp_close	×	×	○	○	○	○	○	FALPモードの終了
falp_connect	×	×	○	×	×	×	×	FALPイニシエータ転送中モードへ移行
falp_listen	○	○	×	×	×	×	×	ターゲット接続待ち開始
falp_stop_listen	×	○	×	×	○	×	○	ターゲット接続待ち停止
set_falp_target_callback_parameters	○	×	×	×	×	×	×	ターゲット接続時のメッセージ設定
falp_target_disconnect	×	×	×	×	×	○	○	FALPターゲットモードの終了
falp_shutdown	×	×	×	○	○	○	○	FALP転送終了モード、FALPターゲット転送終了モードへ移行
falp_recv	×	×	×	○	○	○	○	受信データ取得
falp_send	×	×	×	○	×	○	×	データ送信
falp_wait_event	×	×	×	○	○	○	○	FALPイベント取得
falp_dump_log_to_file	○	○	○	○	○	○	○	ロギング開始
falp_stop_logging	○	○	○	○	○	○	○	ロギング停止
falp_get_last_error_type	○	○	○	○	○	○	○	エラー情報取得
dispose_library get_last_error	○	○	○	○	○	○	○	
close_reader_writer	○	×	×	×	×	×	×	デバイスクローズへ移行
他API	○	○	×	×	×	×	×	

2. API 仕様

各 API の解説にある Error types およびその原因は、各 API の呼び出しで発生する可能性のあるエラーの一例であり、ここで挙げられていないエラーが発生する場合もあります。

2.1. ライブラリ管理関係の API

2.1.1. initialize_library

ライブラリの初期化

bool initialize_library(void)

ライブラリの初期化を行います。get_last_error_type 関数以外は、本 API が実行されていることが実行条件となります。
タイムアウト値やリトライカウント値など、ライブラリの内部で保持している設定情報は、すべて初期化されます。

Parameters:
　　なし
Return values:
　　true　：成功
　　false　：失敗
Error Types:
　　FELICA_LIBRARY_ALREADY_INITIALIZED　：ライブラリはすでに初期化されています
　　FELICA_FALP_OPENED　：FALP モードのため処理できません

2.1.2. dispose_library

bool dispose_library(void)

ライブラリを解放します。リーダ／ライタがオープンされている場合には、自動的にクローズします。

Parameters:
　　なし
Return values:
　　true　：成功
　　false　：失敗
Error Types:
　　FELICA_LIBRARY_NOT_INITIALIZED　　：ライブラリが初期化されていません

2.1.3. get_last_error_type

エラー情報の取得

bool get_last_error_type　(enumeration_felica_error_type*　*error_type*　)

最新のエラー情報を取得します。各 API の実行が失敗(戻り値が false)だった場合に、本 API によりエラー情報を取得することができます。
エラーの詳細は、「3.1 get_last_error_type で取得されるエラー」を参照してください。

Parameters:
　　error_type ［OUT］アクセスライブラリのエラータイプ
　　　　本領域は、アプリケーション側で用意してください。
Return values:
　　true　：成功
　　false　：失敗
Error Types:
　　FELICA_ILLEGAL_ARGUMENT　　：関数の引数が不正です

2.2. リーダ／ライタ制御関係の API

2.2.1. open_reader_writer

リーダ／ライタのオープン

bool open_reader_writer(const structure_reader_writer_mode * *reader_writer_mode*)

リーダ／ライタをオープンします。
同時にオープン可能なリーダ／ライタは 1 つです。

Parameters:

reader_writer_mode [IN]リーダ／ライタモード情報

```
port_name;       // ポートの名前（"USB0"など）
dummy;           // 0 固定
encryption_mode; // 暗号化モード
                 // 0x01 固定
dummy;           // all 0
dummy;           // all 0
```

Return values:

true ：成功
false ：失敗

Error Types:

FELICA_LIBRARY_NOT_INITIALIZED	：ライブラリが初期化されていません
FELICA_FALP_OPENED	：FALP モードのため処理できません
FELICA_ILLEGAL_ARGUMENT	：関数の引数が不正です
FELICA_READER_WRITER_ALREADY_OPENED	：リーダ／ライタはすでにオープンされています
FELICA_READER_WRITER_OPEN_ERROR	：リーダ／ライタのオープンに失敗しました

2.2.2. open_reader_writer_auto

リーダ／ライタの自動認識とオープン

bool open_reader_writer_auto(void)

対象デバイスを走査し、リーダ／ライタを自動認識してオープンします。

走査対象デバイスは、レジストリに設定することができます(各項目最大 16 個まで可能、17 個目以降は無視します)。
レジストリに情報が格納されている場合は、こちらの値をデフォルト値として動作します。ただし、レジストリ値が不正な場合は、デフォルトの情報（USB0、COM1、COM2）を使用します。
レジストリ値は、FeliCa ライブラリロード時に読み込みます。

レジストリ情報の格納先は、
HKEY_LOCAL_MACHINE¥SOFTWARE¥Sony Corporation¥FeliCa Access Library¥ModeList
です。
レジストリへの格納情報・名前および設定値の例を示します。

デバイス走査リスト レジストリ格納情報

名前	型	値（設定例）
port_name_list	文字列(カンマ区切り)	USB0,USB1

PC 内蔵と外付けリーダ／ライタの両方が接続されている場合、外付けリーダ／ライタが優先してオープンされます。
本機能を無効にし、走査リスト順でオープンする場合は、以下のレジストリ値を変更してください。
キー：
HKEY_LOCAL_MACHINE¥SOFTWARE¥Sony Corporation¥FeliCa Access Library¥ModeList
値名：
priority_external (ON = 有効、OFF = 無効)

Parameters:
　　なし
Return values:
　　true　：成功
　　false　：失敗
Error Types:
　　FELICA_LIBRARY_NOT_INITIALIZED　　：ライブラリが初期化されていません
　　FELICA_FALP_OPENED　　：FALP モードのため処理できません
　　FELICA_READER_WRITER_OPEN_AUTO_ERROR　：リーダ／ライタを自動認識できません
　　FELICA_READER_WRITER_ALREADY_OPENED　：リーダ／ライタはすでにオープンされています
　　FELICA_READER_WRITER_OPEN_ERROR　　：リーダ／ライタのオープンに失敗しました

2.2.3. close_reader_writer

リーダ／ライタのクローズ

bool close_reader_writer(void)

リーダ／ライタをクローズします。

Parameters:
　　なし
Return values:
　　true　：成功
　　false　：失敗
Error Types:
　　FELICA_LIBRARY_NOT_INITIALIZED　　：ライブラリが初期化されていません
　　FELICA_FALP_OPENED　　：FALP モードのため処理できません
　　FELICA_READER_WRITER_NOT_OPENED　　：リーダ／ライタがオープンされていません

2.2.4. transaction_lock

トランザクション排他ロックの設定

bool transaction_lock(void)

リーダ／ライタを占有利用するためのアクセス権を獲得します。
本 API が成功すると、リーダ／ライタを共有オープンしている他のプロセスからのコマンドの割り込みは抑止されます。獲得したアクセス権は、transaction_unlock関数・close_reader_writer関数・プロセス終了のいずれかで解放されます。

Parameters:
　　なし
Return values:
　　true　：成功
　　false　：失敗
Error Types:
　　FELICA_LIBRARY_NOT_INITIALIZED　　：ライブラリが初期化されていません
　　FELICA_FALP_OPENED　　：FALP モードのため処理できません
　　FELICA_TRANSACTION_LOCK_ERROR　　：アクセス権獲得（ロック）に失敗しました

2.2.5. transaction_unlock

トランザクション排他ロックの解除

bool transaction_unlock(void)

リーダ／ライタを占有利用するためのアクセス権を解放します。
本 API が成功すると、リーダ／ライタを共有オープンしている他のプロセスがリーダ／ライタにアクセス可能になります。

Parameters:
　　なし
Return values:
　　true　：成功
　　false　：失敗
Error Types:
　　FELICA_LIBRARY_NOT_INITIALIZED　　：ライブラリが初期化されていません
　　FELICA_FALP_OPENED　　：FALP モードのため処理できません
　　FELICA_TRANSACTION_UNLOCK_ERROR　　：アクセス権解放（アンロック）に失敗しました

2.2.6. get_device_information

リーダ／ライタ種別と接続方式の取得

bool get_device_information(const structure_device_information * *device_information*)

オープン中のリーダ／ライタの種別と接続方式を取得します。種別ごとの接続方式は、下記のとおりです。

Parameters:
　　device_information ［OUT］デバイス情報取得結果格納構造体
　　本領域はアプリケーション側で用意してください。
　　device_info_type;　　// リーダ／ライタ種別（下記表参照）
　　device_info_connect;　// 接続方式(0x00:内蔵、0x01：外付け)

Return values:
　　true　：成功
　　false　：失敗

リーダ／ライタ種別	接続方式	対象デバイス
0x00	0x01	RC-S310（ED2）
0x01	0x01	RC-S310（ED2 以外）
0x02	0x01	RC-S320
0x03	0x00	RC-S600/U およびその互換品
0x04	0x0	周辺機器（ワイヤレスキーボード）
0x05	0x00	RC-S620/U およびその互換品
0x06	0x01	マルチサービスリーダ／ライタ

0x07	0x01	RC-S340
0x09	0x01	RC-S330 シリーズ
0x15	0x00	RC-S621/U、RC-S623/U およびその互換品
0x25	0x00	RC-S625/U およびその互換品
0x65	0x01	RC-S360
0x62	0x00	RC-S632 シリーズ(NFC Port-100)
0x63	0x00	RC-S634 シリーズ(NFC Port-100)
0x6A/6B/6C	0x01	RC-S380 シリーズ(NFC Port-100)

Error Types:

FELICA_LIBRARY_NOT_INITIALIZED	: ライブラリが初期化されていません
FLICA_FALP_OPENED	: FALP モードのため処理できません
FELICA_ILLEGAL_ARGUMENT	: 関数の引数が不正です
FELICA_GET_DEVICE_INFO_FAILED	: デバイス情報取得に失敗しました

2.3. ライブラリの各種設定を行う API

2.3.1. set_time_out

タイムアウト値の設定

bool set_time_out(unsigned long *time_out*)

リーダ／ライタからの応答を待つためのタイムアウト値を設定します。
デフォルトの値は、400ms です。
リーダ／ライタ管理コマンドや発行コマンドをご利用になる場合には、1000ms などこれよりも長めに設定してください。

＜ご注意＞
NFC ポート／パソリをご使用の場合、400ms 未満の設定は 400ms に切り上げられます（ワイヤレスキーボードをご使用の場合は 600ms）。また、ポーリング時のタイムアウト値は、「2.3.5 set_polling_timeout」にて設定してください。

Parameters:
　　time_out ［IN］タイムアウト値（ms）　（0≦タイムアウト値≦6553）
Return values:
　　true　：成功
　　false　：失敗
Error Types:
　　FELICA_LIBRARY_NOT_INITIALIZED　　：ライブラリが初期化されていません
　　FELICA_FALP_OPENED　　：FALP モードのため処理できません

2.3.2. get_time_out

タイムアウト値の取得

bool get_time_out(unsigned long * *time_out*)

設定されているタイムアウト値を取得します。

Parameters:
　　time_out ［OUT］タイムアウト値（ms）
　　　　本領域はアプリケーション側で用意してください。
Return values:
　　true　：成功
　　false　：失敗
Error Types:
　　FELICA_LIBRARY_NOT_INITIALIZED　　：ライブラリが初期化されていません
　　FELICA_FALP_OPENED　　：FALP モードのため処理できません
　　FELICA_ILLEGAL_ARGUMENT　　：関数の引数が不正です

2.3.3. set_lock_timeout

アクセス権獲得タイムアウト値の設定

bool set_lock_timeout(unsigned long *lock_timeout*)

リーダ／ライタを共有オープンしている状態で、他のプロセスがアクセス権を保有している場合、リーダ／ライタを使用可能な状態まで待ち合わせるタイムアウト間隔(ms 単位)を設定します。本 API で設定されたタイムアウトが経過してもアクセス権が獲得できない場合は、FeliCa ライブラリの関数はエラーとなり、FeliCa リーダ／ライタコントロールライブラリエラータイプには RW_LOCK_TIMEOUT がセットされます。FeliCa アクセスライブラリエラータイプは、コマンド種別で異なります。
lock_timeout に 0 を指定した場合は、無限待ちとなります。デフォルト値は 4000ms です。

Parameters:
lock_timeout [IN] アクセス権獲得タイムアウト値(ms)
0:無限待ち
Return values:
true ：成功
false ：失敗
Error Types:

FELICA_LIBRARY_NOT_INITIALIZED	：ライブラリが初期化されていません
FELICA_FALP_OPENED	：FALP モードのため処理できません
FELICA_SET_LOCK_TIMEOUT_ERROR	：アクセス権獲得タイムアウト設定に失敗しました

2.3.4. get_lock_timeout

アクセス権獲得タイムアウト値の取得

bool get_lock_timeout(unsigned long * *lock_timeout*)

アクセス権獲得タイムアウト値を取得します。

Parameters:
lock_timeout [OUT] 現在設定されているアクセス権獲得タイムアウト値(ms)
0:無限待ち
Return values:
true ：成功
false ：失敗
Error Types:

FELICA_LIBRARY_NOT_INITIALIZED	：ライブラリが初期化されていません
FELICA_FALP_OPENED	：FALP モードのため処理できません
FELICA_ILLEGAL_ARGUMENT	：関数の引数が不正です
FELICA_GET_LOCK_TIMEOUT_ERROR	：アクセス権獲得タイムアウト取得に失敗しました

2.3.5. set_polling_timeout

ポーリング専用タイムアウト値の設定

bool set_polling_timeout (unsigned long *polling_timeout*)

NFC ポート／パソリにおいて、Polling コマンドと Request Response コマンドで使用されるポーリング専用タイムアウト値(ms 単位)を設定します。デフォルト値は 100ms です。

＜ご注意＞
100ms 未満の設定は、100ms に切り上げられます（ワイヤレスキーボードをご使用の場合は 250ms）。また、ポーリング以外のタイムアウト値は、「2.3.1 set_time_out」関数にて設定してください。

Parameters:
polling_timeout [IN]ポーリング専用タイムアウト値(ms)
Return values:
true ：成功
false ：失敗
Error Types:

FELICA_LIBRARY_NOT_INITIALIZED	：ライブラリが初期化されていません
FELICA_FALP_OPENED	：FALP モードのため処理できません
FELICA_SET_POLLING_TIMEOUT_ERROR	：ポーリングタイムアウト設定に失敗しました

2.3.6. get_polling_timeout

ポーリング専用タイムアウト値の取得

bool get_polling_timeout (unsigned long * *polling_timeout*)

NFC ポート／パソリにおいて、Polling コマンドと Request Rresponse コマンドで使用されるポーリング専用タイムアウト値(ms 単位)を取得します。

Parameters:
polling_timeout [OUT] 現在設定されているポーリング専用タイムアウト値(ms)
Return values:
true ：成功
false ：失敗
Error Types:

FELICA_LIBRARY_NOT_INITIALIZED	：ライブラリが初期化されていません
FELICA_FALP_OPENED	：FALP モードのため処理できません
FELICA_ILLEGAL_ARGUMENT	：関数の引数が不正です
FELICA_GET_POLLING_TIMEOUT_ERROR	：ポーリングタイムアウト取得に失敗しました

2.4. ポーリングに関する API

2.4.1. polling_and_get_card_information

ポーリングとカード情報の取得（Polling コマンド）

```
bool polling_and_get_card_information (
        const structure_polling *       polling,
        unsigned char *                 number_of_cards,
        structure_card_information *    card_information
)
```

ポーリングを実行し、カードの情報を取得します。捕捉できるカードは 1 枚です。

Parameters:

polling	[IN]	ポーリングをするために必要な情報
		system_code; // システムコード(2 バイト)
		time_slot; // タイムスロット

number_of_cards	[OUT]	捕捉したカードの枚数（=1）

card_information	[OUT]	カードの情報
		本領域はアプリケーション側で用意してください。
		card_idm; // カードの IDm（8 バイト）
		card_pmm; // カードの PMm（8 バイト）

Return values:

true	：成功
false	：失敗

Error Types:

FELICA_LIBRARY_NOT_INITIALIZED	：ライブラリが初期化されていません
FELICA_READER_WRITER_NOT_OPENED	：リーダ／ライタがオープンされていません
FELICA_FALP_OPENED	：FALP モードのため処理できません
FELICA_ILLEGAL_ARGUMENT	：関数の引数が不正です
FELICA_POLLING_ERROR	：ポーリングに失敗しました

2.4.2. get_last_card_information

カード情報の取得

```
bool get_last_card_information (
        unsigned char                   card_index,
        structure_card_information *    card_information
)
```

最後に通信したカードの情報を取得します。

Parameters:

card_index	[IN]	カードのインデックス (=1)
card_information	[OUT]	インデックスで指定したカード情報を取得します。 本領域はアプリケーションで確保してください。 card_idm // カードの IDm (8 バイト) card_pmm // カードの PMm (8 バイト)

Return values:

true	: 成功
false	: 失敗

Error Types:

FELICA_LIBRARY_NOT_INITIALIZED	: ライブラリが初期化されていません
FELICA_FALP_OPENED	: FALP モードのため処理できません
FELICA_ILLEGAL_ARGUMENT	: 関数の引数が不正です
FELICA_INVALID_CARD_INDEX	: カードのインデックスが不正なためカードの情報を取得できませんでした
FELICA_CARD_INFORMATION_ACCESS_ERROR	: カードの情報の取得に失敗しました

2.5.FALP に関する API

2.5.1. falp_open

FALP モードへ移行します。

bool falp_open(void)

falp_open()を呼び出す前に、リーダ／ライタをオープンして対向機器とポーリングを行う必要があります。

FALP モードへの移行が成功すると、FALP 関連 API を利用して FALP 通信が可能になります。

Parameters:
　　なし
Return values:
　　true 成功
　　false 失敗
Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_READER_WRITER_NOT_OPENED	: リーダ／ライタがオープンされていません
FALP_START_POLLING_STARTED	: START POLLING が稼動しているため FALP を開始できません
FALP_ALREADY_OPENED	: FALP モードにすでに移行しています
FALP_PARAM_ERROR	: パラメータが不正です
FALP_MEMORY_ALLOCATION_ERROR	: メモリの確保ができません
FALP_TARGET_TRANSMIT_MODE	: FALP ターゲット転送モードです
FALP_TARGET_TRANSMIT_END_MODE	: FALP ターゲット転送終了モードです

2.5.2. falp_close

FALP モードを終了します。

bool falp_close(void)

FALP 通信中の場合は、速やかに FALP 通信を終了します。FALP 通信を、特定の条件で終了したい場合には、この API の前に falp_shutdown()を呼び出します。

Parameters:
　　なし
Return values:
　　true　成功
　　false　失敗
Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_READER_WRITER_NOT_OPENED	: リーダ／ライタがオープンされていません
FALP_NOT_OPENED	: FALP モードではありません
FALP_PARAM_ERROR	: パラメータが不正です

2.5.3. falp_connect

FALP 通信の接続を行います。

```
bool falp_connect (
        unsigned short              propose_time_out
        unsigned short              handshake_time_out
        const unsigned char *       appid
        unsigned char               appid_length
        const unsigned char *       data
        unsigned long *             data_length
)
```

指定されたタイムアウト時間まで、ブロッキングして接続の確立を待ちます。

propose_time_out は、リーダ／ライタの対向機器に FALP 通信を開始するためのコマンドを送信して、そのレスポンスを待つ時間です。推奨値は、2000ms です。推奨値は、すべての対向機器に対して動作を保証するものではありません。十分な動作検証を行ってください。

handshake_time_out は、FALP のハンドシェイク時に、対向機器からハンドシェイクの応答を待つ時間です。推奨値は、3000ms です。推奨値は、すべての対向機器に対して動作を保証するものではありません。十分な動作検証を行ってください。

appid は、アプリケーション識別子です。アプリケーション識別子は、サービスごとに規定される値です。アプリケーション識別子の値については、各サービス事業者にお問い合わせください。

data を指定した場合は、接続の確立後すぐにデータの送信を開始します。

〈ご注意〉

本関数の実行にはアクセス権が必要です。falp_open()の前に、transaction_lock()を実行し、アクセス権を獲得するようにしてください。

falp_connect 時にデータを送信する場合、データサイズが小さい(概ね数百バイト以下)場合、FALP_SHUTDOWN エラーが発生する場合があります。これは、falp_connect()処理中にデータ送信が完了してしまうためです。この現象に対処するには、上位アプリケーションで次のように対処してください。
falp_connect()で false が返却された場合、falp_get_last_error()でエラーコードを取得し、
・エラーコードが、FALP_SHUTDOWN であった場合は通信終了
・それ以外であればエラー
と判断する

Parameters:

propose_time_out	[IN] Propose Ad-hoc コマンドのタイムアウト値(ms) 0xff ≦ propose_time_out ≦ 0x8000
handshake_time_out	[IN] FALP のハンドシェイクのタイムアウト値(ms) 0xffff が指定された場合は無制限に待つ
appid	[IN] アプリケーション識別子
appid_length	[IN] アプリケーション識別子のサイズ(バイト)
data	[IN] 送信データ NULL が指定された場合は何も送信しない
data_length	[IN/OUT] 送信データのサイズ(バイト)

成功時は実際に送信したデータのサイズが返る
0 が指定された場合は何も送信しない

Return values:
true 成功
false 失敗

Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_READER_WRITER_NOT_OPENED	: リーダ／ライタがオープンされていません
FALP_NOT_OPENED	: FALP モードではありません
FALP_SHUTDOWN	: シャットダウンされています
FALP_PARAM_ERROR	: パラメータが不正です
FALP_ILLEGAL_STATE_ERROR	: 状態が不正です
FALP_HANDSHAKE_FAIL	: ハンドシェイクに失敗しました
FALP_PROPOSE_TIMEOUT	: Propose Ad-hoc の応答待ちでタイムアウトが発生しました
FALP_DEVICE_BUSY	: デバイスが他で使用されているか、アクセス権が獲得できません
FALP_FELICA_THRU_ERROR	: FALP 通信が失敗しました
FALP_DEVICE_COMMUNICATION_ERROR	: USB 通信が失敗しました
FALP_USB_DRIVER_INTERNAL_ERROR	: USB ドライバ内部で異常が発生しました
FALP_TARGET_NOT_FOUND	: ターゲットを捕捉することができませんでした
FALP_TARGET_TRANSMIT_MODE	: FALP ターゲット転送モードです
FALP_TARGET_TRANSMIT_END_MODE	: FALP ターゲット転送終了モードです

2.5.4. falp_listen

FALP ターゲット通信接続待ちを開始します。

```
bool falp_listen (
        const int                   num_of_app_ids
        const int                   app_id_size[]
        const unsigned char *       appid[]
        char                        result[]
)
```

指定アプリケーション識別子を対象として、FALP 通信接続待ちを開始します。

対向機器から接続要求を受信後、「FALP ターゲットモード」へ移行します。
この際、set_falp_target_callback_parameters()関数で登録したメッセージに対応する Window メッセージをポストします。

FALP 通信接続待ち中に本関数を呼び出すことにより、アプリケーション識別子の追加・削除が行えます。アプリケーション識別子は、変更後のものをすべて再登録してください。特定のアプリケーション識別子のみを指定して、アプリケーション識別子の追加・停止はできません。

<ご注意>
本関数実行前に、set_falp_target_callback_parameters()関数で、FALP 通信接続受信時のイベント通知コールバック用のメッセージを登録しておく必要があります。

指定したアプリケーション識別子が、すでに他アプリケーションにおいて指定されている場合、パラメータ result に 1（失敗）を格納します。

指定したアプリケーション識別子すべての登録に失敗した場合、戻り値は false（失敗）となります。１つでも成功した場合 true（成功）となります。

NFC Port-100 では本機能は利用できません。

Parameters:

num_of_ipp_ids	[IN]	アプリケーション識別子の数（n） 1 ≦ n ≦ 32
app_id_size	[IN]	各アプリケーション識別子のサイズ（m） 0 ≦ m ≦ 32
appid	[IN]	アプリケーション識別子
result	[OUT]	各アプリケーション識別子の登録結果 (0 : 成功、 1: 失敗)

Return values:

true 成功
false 失敗

Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_READER_WRITER_NOT_OPENED	: リーダ／ライタがオープンされていません
FALP_NOT_OPENED	: FALP モードではありません

FALP_PARAM_ERROR	： パラメータが不正です
FALP_CANNOT_START_TARGET_WAIT	： FALP ターゲット接続待ちを開始できません
FALP_MEMORY_ALLOCATION_ERROR	：メモリの確保ができません
FALP_INITIATOR_IDLE_MODE	：FALP IDLE モードです
FALP_INITIATOR_TRANSMIT_MOD	： FALP イニシエータ転送モードです
FALP_INITIATOR_TRANSMIT_END_MODE	： FALP イニシエータ転送終了モードです
FALP_TARGET_TRANSMIT_MODE	： FALP ターゲット転送モードです
FALP_TARGET_TRANSMIT_END_MODE	： FALP ターゲット転送終了モードです
FALP_CALL_BACK_PARAMETERS_NOT_SET	： FALP ターゲット用コールバックパラメータが設定されていません

2.5.5. falp_stop_listen

FALP ターゲット通信接続待ちを停止します。

bool falp_stop_stop_listen ()

FALP ターゲット通信接続待ちを停止します。
「FALP ターゲット転送終了モード」中に本関数を指示した場合、通常モードに遷移後に FALP 通信接続待ちを開始しません。

〈ご注意〉
本関数では、特定のアプリケーション識別子の追加・削除はできません。
この場合は falp_listen()関数をご使用ください。

NFC Port-100 では本機能は利用できません。

Parameters:
　　なし
Return values:
　　true 成功
　　false 失敗
Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	： FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	： FALP ライブラリの初期化に失敗しました
FALP_READER_WRITER_NOT_OPENED	： リーダ／ライタがオープンされていません
FALP_INITIATOR_IDLE_MODE	： FALP IDLE モードです
FALP_INITIATOR_TRANSMIT_MODE	： FALP イニシエータ転送モードです
FALP_TARGET_TRANSMIT_MODE	： FALP ターゲット転送モードです
FALP_NOT_TARGET_WAIT_MODE	： FALP ターゲット接続待ち要求を発行していません

2.5.6. set_falp_target_callback_parameters

FALP 通信接続受信時のイベント通知コールバック用メッセージを登録します。

```
bool falp_target_callback_parameters (
        const int                   handle
        const unsigned char *       msg
)
```

FALP 通信接続受信時のイベント通知コールバック用のメッセージを登録します。

Window プロシージャに渡される 2 個のパラメータ（WPARAM と LPARAM）にそれぞれ、接続されたアプリケーション ID のサイズ、接続されたアプリケーション識別子のアドレスが格納されます。

デバイスからの接続を待機している状態で異常が発生した場合もイベントを通知します。この際、アプリケーション識別子のサイズを 0 とし、アプリケーション識別子には FALP エラーコードが格納されます。エラーコードは enumeration_falp_error_type*の値です（リトルエンディアン）。

ターゲット接続待ち要求の発行（falp_listen()関数）前に、本関数を実行しておく必要があります。

<ご注意>
Window メッセージの定義には、Windows API の RegisterWindowMessage()関数を利用してください。メッセージ文字列長は最大 255 バイトです(終端 NULL 除く)。

イベント発生時、Window プロシージャに渡される 2 個のパラメータは、Windows API の WindowProc()関数においては第 3 引数(WPARAM wp) と第 4 引数(LPARAM lp)にあたります。

NFC Port-100 では本機能は利用できません。

Parameters:

	handle	[IN] ウィンドウハンドル
	msg	[IN] FALP 通信接続受信時のイベント通知コールバック用メッセージ

Return values:

true	成功
false	失敗

Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_PARAM_ERROR	: パラメータが不正です
FALP_INITIATOR_IDLE_MODE	: FALP IDLE モードです
FALP_INITIATOR_TRANSMIT_MODE	: FALP イニシエータ転送モードです
FALP_INITIATOR_TRANSMIT_END_MODE	: FALP イニシエータ転送終了モードです
FELICA_FALP_TARGET_MODE	: FALP ターゲットモードです
FALP_TARGET_WAIT_MODE	:すでに FALP ターゲット接続待ち要求が発行されています
FALP_MESSAGE_OF_TARGET_CONNECT_REGISTRATION_ERROR	: FALP ターゲット接続通知メッセージの登録に失敗しました

2.5.7. falp_target_disconnect

FALP ターゲットモードを終了します。

bool falp_target_disconnect ()

FALP ターゲットモードを終了します。
FALP ターゲット通信中モードの場合、直ちに FALP 通信を終了します。
FALP 通信を、特定の条件で終了したい場合には、この API の前に falp_shutdown()関数を呼び出すようにしてください。

〈ご注意〉
FALP モードを終了する場合（falp_open()関数を使用した場合）は、falp_close()関数を使用してください。

FALP ターゲット通信接続待ちを停止する場合は、falp_stop_listen()関数を使用してください。

NFC Port-100 では本機能は利用できません。

Parameters:
　　なし
Return values:
　　true 成功
　　false 失敗
Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_READER_WRITER_NOT_OPENED	: リーダ／ライタがオープンされていません
FALP_INITIATOR_IDLE_MODE	: FALP IDLE モードです
FALP_INITIATOR_TRANSMIT_MODE	: FALP イニシエータ転送モードです
FALP_INITIATOR_TRANSMIT_END_MODE	: FALP イニシエータ転送終了モードです
FALP_NOT_TARGET_MODE	: FALP ターゲットモードではありません

2.5.8. falp_shutdown

指定された条件で FALP 通信を終了します。

```
bool falp_shutdown (
        unsigned long           flag
)
```

FALP 通信の終了を確認するためには、falp_wait_event()を利用します。

FALP 通信の終了後に、再度、FALP 通信を行う場合は、一度 falp_close をよんで FALP モードを終了する必要があります。

flag には、以下のいずれかの値を設定します。

- FALP_SHUTDOWN_FLAG_SEND　　自分の送信バッファが空になったとき FALP 通信を終了します。以降、falp_send()を呼び出すことはできなくなります。
- FALP_SHUTDOWN_FLAG_RECV　　相手の送信バッファが空になったとき FALP 通信を終了します。
- FALP_SHUTDOWN_FLAG_BOTH　　FALP_SHUTDOWN_FLAG_SEND と FALP_SHUTDOWN_FLAG_RECV の両方の意味を持ちます。

Parameters:
　　flag　　[IN] シャットダウンフラグ
Return values:
　　true 成功
　　false 失敗
Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_READER_WRITER_NOT_OPENED	: リーダ／ライタがオープンされていません
FALP_NOT_OPENED	: FALP モードではありません
FALP_PARAM_ERROR	: パラメータが不正です
FALP_ILLEGAL_STATE_ERROR	: 状態が不正です
FALP_FELICA_THRU_ERROR	: FALP 通信が失敗しました
FALP_DEVICE_COMMUNICATION_ERROR	: USB 通信が失敗しました
FALP_USB_DRIVER_INTERNAL_ERROR	: USB ドライバ内部で異常が発生しました
RV_ILLEGAL_STATE_ERROR	: 状態が不正です

2.5.9. falp_recv

受信データを取得します。

```
bool falp_recv (
        unsigned char *         data
        unsigned long *         data_length
)
```

受信バッファからデータを読み込みます。受信バッファからデータを読み込むと、ただちに復帰します。受信バッファにデータがない場合は、data_length に 0 が返ります。

受信バッファにデータが格納されているかを調べるためには、falp_wait_event()を利用します。

Parameters:

data	[OUT] 受信データ格納領域 成功時は実際に取得したデータが返る
data_length	[IN/OUT] 受信データ格納領域の長さ(バイト) 成功時は実際に取得したデータの長さが返る

Return values:

- true 成功
- false 失敗

Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_READER_WRITER_NOT_OPENED	: リーダ／ライタがオープンされていません
FALP_NOT_OPENED	: FALP モードではありません
FALP_PARAM_ERROR	: パラメータが不正です
FALP_ILLEGAL_STATE_ERROR	: 状態が不正です
FALP_FELICA_THRU_ERROR	: FALP 通信が失敗しました
FALP_DEVICE_COMMUNICATION_ERROR	: USB 通信が失敗しました
FALP_USB_DRIVER_INTERNAL_ERROR	: USB ドライバ内部で異常が発生しました

2.5.10. falp_send

データを送信します。

```
bool falp_send (
        unsigned char *         data
        unsigned long *         data_length
)
```

送信バッファにデータを書き込みます。送信バッファにデータを書き込むと、ただちに復帰します。実際の送信処理は非同期に行われます。送信バッファに空きがない場合は、data_length に 0 が返ります。

送信バッファに空きができたか、または送信バッファのデータ送信が完了して送信バッファが空になったかを調べるためには、falp_wait_event()を利用します。

Parameters:

data	[IN] 送信データ格納領域
data_length	[IN/OUT] 送信データ格納領域の長さ(バイト) 成功時は送信バッファに書き込んだ長さが返る

Return values:

true 成功
false 失敗

Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_READER_WRITER_NOT_OPENED	: リーダ／ライタがオープンされていません
FALP_NOT_OPENED	: FALP モードではありません
FALP_PARAM_ERROR	: パラメータが不正です
FALP_ILLEGAL_STATE_ERROR	: 状態が不正です
FALP_SHUTDOWN	: シャットダウンされています

2.5.11. falp_wait_event

FALP イベントを取得します。

```
bool falp_wait_event(
        unsigned short          time_out
        unsigned long *         event
        unsigned long           event_mask
)
```

event_mask には、以下のイベントを指定します。複数のイベントを論理和で指定することも可能です。

・FALP_EVENT_SEND_READY　　送信バッファに空きがあります。

・FALP_EVENT_RECV_READY　　受信バッファにデータがあります。

・FALP_EVENT_SHUTDOWNED　　シャットダウンが完了しました。

・FALP_EVENT_SEND_EMPTY　　送信バッファが空になりました。
通信相手は全データを受信しています。

time_out 時間までに指定したイベントのいずれかを取得できた場合は、その論理和が event に返ります。

Parameters:

time_out	[IN]	タイムアウト(ms) 0xffff が指定された場合は無制限に待つ
event	[OUT]	取得したイベント
event_mask	[IN]	取得するイベント

Return values:

true 成功
false 失敗

Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_READER_WRITER_NOT_OPENED	: リーダ／ライタがオープンされていません
FALP_NOT_OPENED	: FALP モードではありません
FALP_PARAM_ERROR	: パラメータが不正です
FALP_ILLEGAL_STATE_ERROR	: 状態が不正です
FALP_TIMEOUT	: タイムアウトが発生しました
FALP_FELICA_THRU_ERROR	: FALP 通信が失敗しました
FALP_DEVICE_COMMUNICATION_ERROR	: USB 通信が失敗しました
FALP_USB_DRIVER_INTERNAL_ERROR	: USB ドライバ内部で異常が発生しました
RV_ILLEGAL_STATE_ERROR	: 状態が不正です

2.5.12. falp_dump_log_to_file

FALP 通信のログ出力を開始します。

```
bool falp_dump_log_to_file (
        const unsigned char *       file_name
)
```

file_name には、ファイル名に利用できる文字で構成された NULL 終端文字列を指定します。ファイルは追記で出力します。

Parameters:
　　file_name　　[IN] ログファイル名
Return values:
　　true 成功
　　false 失敗
Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_PARAM_ERROR	: パラメータが不正です
FALP_FELICA_THRU_ERROR	: FALP 通信が失敗しました
FALP_DEVICE_COMMUNICATION_ERROR	: USB 通信が失敗しました
FALP_USB_DRIVER_INTERNAL_ERROR	: USB ドライバ内部で異常が発生しました

2.5.13. falp_stop_logging

FALP 通信のログ出力を停止します。

```
bool falp_stop_logging(void )
```

Parameters:
　　なし
Return values:
　　true 成功
　　false 失敗
Error Types:

FALP_FELICA_LIBRARY_NOT_INITIALIZED	: FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR	: FALP ライブラリの初期化に失敗しました
FALP_FELICA_THRU_ERROR	: FALP 通信が失敗しました
FALP_DEVICE_COMMUNICATION_ERROR	: USB 通信が失敗しました
FALP_USB_DRIVER_INTERNAL_ERROR	: USB ドライバ内部で異常が発生しました

2.5.14. falp_get_last_error_type

FALP 関連 API のエラー情報を取得します。

```
bool falp_get_last_error_type (
        enumeration_falp_error_type *    error_type
)
```

最後に発生したエラータイプを返します。

Parameters:
error_type [OUT] エラータイプ
Return values:
true 成功
false 失敗
Error Types:
FALP_FELICA_LIBRARY_NOT_INITIALIZED : FeliCa ライブラリが初期化されていません
FALP_LIBRARY_INITIALIZE_ERROR : FALP ライブラリの初期化に失敗しました

3. エラータイプ

3.1. get_last_error_type で取得されるエラー

コード（10 進数）	コード（16 進数）	タイプ	説明
1002	0x3ea	FELICA_ILLEGAL_ARGUMENT	関数の引数が不正です
1005	0x3ed	FELICA_LIBRARY_NOT_INITIALIZED	ライブラリが初期化されていません
1006	0x3ee	FELICA_LIBRARY_ALREADY_INITIALIZED	ライブラリはすでに初期化されています
1027	0x403	FELICA_READER_WRITER_OPEN_ERROR	リーダ／ライタのオープンに失敗しました
1028	0x404	FELICA_READER_WRITER_OPEN_AUTO_ERROR	リーダ／ライタを自動認識できません
1029	0x405	FELICA_READER_WRITER_NOT_OPENED	リーダ／ライタがオープンされていません
1030	0x406	FELICA_READER_WRITER_ALREADY_OPENED	リーダ／ライタはすでにオープンされています
1036	0x40c	FELICA_POLLING_ERROR	ポーリングに失敗しました
1047	0x417	FELICA_CARD_INFORMATION_ACCESS_ERROR	カードの情報の取得に失敗しました
1048	0x418	FELICA_INVALID_CARD_INDEX	カードのインデックスが不正なためカードの情報を取得できませんでした
1065	0x429	FELICA_TRANSACTION_LOCK_ERROR	アクセス権獲得（ロック）に失敗しました
1067	0x42b	FELICA_SET_LOCK_TIMEOUT_ERROR	アクセス権獲得タイムアウト設定に失敗しました
1068	0x42c	FELICA_GET_LOCK_TIMEOUT_ERROR	アクセス権獲得タイムアウト取得に失敗しました
1069	0x42d	FELICA_GET_DEVICE_INFO_FAILED	デバイス情報取得に失敗しました
1071	0x42f	FELICA_SET_POLLING_TIMEOUT_ERROR	ポーリングタイムアウト設定に失敗しました
1072	0x430	FELICA_GET_POLLING_TIMEOUT_ERROR	ポーリングタイムアウト取得に失敗しました
1081	0x439	FELICA_FALP_OPENED	FALPモードのため処理できません

3.2. falp_get_last_error_type()で取得されるエラー

コード (10 進数)	コード (16 進数)	タイプ	説明
2000	0x7d0	FALP_ERROR_NOT_OCCURRED	エラーは発生していません
2001	0x7d1	FALP_TIMEOUT	タイムアウトが発生しました
2002	0x7d2	FALP_SHUTDOWN	シャットダウンされています
2003	0x7d3	FALP_HANDSHAKE_FAIL	ハンドシェイクに失敗しました
2004	0x7d4	FALP_PARAM_ERROR	パラメータが不正です
2005	0x7d5	FALP_MEMORY_ALLOCATION_ERROR	メモリの確保ができません
2006	0x7d6	FALP_OUT_OF_RANGE	範囲外の値が指定されました
2007	0x7d7	FALP_ILLEGAL_STATE_ERROR	状態が不正です
2008	0x7d8	FALP_FELICA_COMMAND_TIMEOUT	FeliCaコマンドがタイムアウトしました
2009	0x7d9	FALP_PACKET_FORMAT_ERROR	パケットフォーマットが不正です
2010	0x7da	FALP_PORT_BUSY	ポートビジーが発生しました
2011	0x7db	FALP_RW_LIBRARY_ERROR	FeliCaリーダ／ライタコントロールライブラリからエラーが返されました
2012	0x7dc	FALP_RW_RECEIVE_DATA_LENGTH_ERROR	受信したデータの長さが異常です
2013	0x7dd	FALP_RW_RECEIVE_DATA_CODE_ERROR	受信したコードが異常です
2014	0x7de	FALP_RW_GET_ATTRIBUE_RATE_ERROR	リーダ／ライタから無線通信速度が正しく取得できませんでした
2015	0x7df	FALP_FELICA_LIBRARY_NOT_INITIALIZED	FeliCaライブラリが初期化されていません
2016	0x7e0	FALP_LIBRARY_INITIALIZE_ERROR	FALPライブラリの初期化に失敗しました
2017	0x7e1	FALP_READER_WRITER_NOT_OPENED	リーダ／ライタがオープンされていません
2018	0x7e2	FALP_START_POLLING_STARTED	START POLLINGが稼動しているためFALPを開始できません
2019	0x7e3	FALP_ALREADY_OPENED	FALPモードにすでに移行しています
2020	0x7e4	FALP_NOT_OPENED	FALPモードではありません
2021	0x7e5	FALP_DEVICE_COMMUNICATION_ERROR	USB通信が失敗しました
2022	0x7e6	FALP_DEVICE_BUSY	デバイスが他で使用されているか、アクセス権が獲得できません
2023	0x7e7	FALP_DEVICE_COMMUNICATION_PARAMETER_ERROR	デバイスアクセス時のパラメータが不正です
2024	0x7e8	FALP_FELICA_COMMAND_ERROR	FeliCaコマンドレスポンスでエラーが通知されました
2025	0x7e9	FALP_FELICA_THRU_ERROR	FALP通信が失敗しました
2026	0x7ea	FALP_SYNTAX_ERROR	シンタックスエラーが発生しました
2027	0x7eb	FALP_COMMAND_CHECKSUM_ERROR	コマンドチェックサム(LCS,DCS)エラーが発生しました
2028	0x7ec	FALP_COMMAND_IDTR_ERROR	不正なトランザクションIDが検出されました
2029	0x7ed	FALP_READER_WRITER_MODE_ERROR	リーダ／ライタのモード状態によりコマンドが実行されませんでした
2030	0x7ee	FALP_ILLEGAL_COMMAND_PACKET	コマンドのヘッダ・フッタ部が不正です
2031	0x7ef	FALP_STATUS_FLAG_ERROR	ステータスコードが正常・タイムアウト以外です
2032	0x7f0	FALP_LOG_FILE_OPEN_ERROR	FALPログファイルがオープンできません
2033	0x7f1	FALP_LOG_START_ERROR	FALPログの取得開始に失敗しました
2034	0x7f2	FALP_UNKNOWN_ERROR	原因不明のエラーが発生しました
2035	0x7f3	FALP_USB_DRIVER_INTERNAL_ERROR	USBドライバ内部で異常が発生しました
2036	0x7f4	FALP_TARGET_NOT_FOUND	ターゲットを捕捉することができませんでした
2037	0x7f5	FALP_INITIATOR_IDLE_MODE	FALP IDLEモードです
2038	0x7f6	FALP_INITIATOR_TRANSMIT_MODE	FALP イニシエータ転送モードです
2039	0x7f7	FALP_INITIATOR_TRANSMIT_END_MODE	FALP イニシエータ転送終了モードです
2040	0x7f8	FALP_TARGET_TRANSMIT_MODE	FALP ターゲット転送モードです
2041	0x7f9	FALP_TARGET_TRANSMIT_END_MODE	FALP ターゲット転送終了モードです

2042	0x7fa	FALP_CALL_BACK_PARAMETERS_NOT_SET	FALP ターゲット用コールバックパラメータが設定されていません
2043	0x7fb	FALP_TARGET_WAIT_MODE	すでにFALPターゲット接続待ち要求が発行されています
2044	0x7fc	FALP_NOT_TARGET_WAIT_MODE	FALPターゲット接続待ち要求を発行していません
2045	0x7fd	FALP_MESSAGE_OF_TARGET_CONNECT_REGISTRATION_ERROR	FALPターゲット接続通知メッセージの登録に失敗しました
2046	0x7fe	FALP_NOT_TARGET_MODE	FALP ターゲットモードではありません
2047	0x7ff	FALP_CANNOT_START_TARGET_WAIT	FALP ターゲット接続待ちを開始できません
2048	0x800	FALP_PROPOSE_TIMEOUT	Propose Ad-hocの応答待ちでタイムアウトが発生しました
2049	0x801	FALP_MODE_TRANSMIT_ERROR	モード移行のためのデバイス設定処理でエラーが発生しました
2050	0x802	FALP_DEVICE_INITIALIZE_ERROR	デバイスの初期化でエラーが発生しました
2051	0x803	FALP_TEMPERATURE_ERROR	デバイスが温度異常です
2052	0x804	FALP_TEMPERATURE_FATAL_ERROR	温度異常によりデバイスが使用できない状態です
2053	0x805	FALP_SYSTEM_BUSY	システムビジーが発生しました

（このページは白紙です。）

SDK for NFC ユーザーズマニュアル
FeliCa ライブラリ編（Starter Kit 版）　Version 1.2

2011 年 11 月　　初版発行　　FeliCa 事業部

2013 年 6 月　　改訂

ソニー株式会社

No. M735-J01-20