CASIO®

NFCFelica ライブラリマニュアル

このマニュアルは、NFCFelica ライブラリの
仕様について記載します。

Ver. 1.01

ご注意

- このソフトウェアおよびマニュアルの、一部または全部を無断で使用、複製することはできません。
- このソフトウェアおよびマニュアルは、本製品の使用許諾契約書のもとでのみ使用することができます。
- このソフトウェアおよびマニュアルを運用した結果の影響については、一切の責任を負いかねますのでご了承ください。
- このソフトウェアの仕様、およびマニュアルに記載されている事柄は、将来予告なしに変更することがあります。
- このマニュアルの著作権はカシオ計算機株式会社に帰属します。
- 本書中に含まれている画面表示は、実際の画面とは若干異なる場合があります。予めご了承ください。

変更履歴

バージョン	変更日付	ページ	内容
0.90	2009.07		開発推進版作成
1.00	2010.05		正版作成
1.01	2012.01		対象機種に DT-X8 および IT-9000 を追加
			関数を追加
	2012.02		FeliCa のスペルを修正
			概要を修正
			関数呼び出し手順を追加

目次

1. 概要

NFC（Near Field Communication）Felica ライブラリは、FeliCa カード（※）との通信を行う関数を提供します。
※ FeliCa は、ソニー株式会社が開発した非接触 IC カードの技術方式です。FeliCa は、ソニー株式会社の登録商標です。

NFC ライブラリを使用して FeliCa カードにアクセスする場合、業務アプリケーションは自分で FeliCa コマンドを作成し、FeliCa カードに送信する必要があります。NFCFelica ライブラリは、業務アプリケーションの代わりに FeliCa コマンドの作成を行なうことで、FeliCa カードへのアクセスをサポートします。

対象の IC カードが FeliCa に限定される場合においては、NFC ライブラリを使用するよりも、NFCFelica ライブラリを使用する方が効率的にアプリケーションを開発することができます。

NFCFelica ライブラリを使用することにより、機種を意識することなく、アプリケーションのソースコード互換性を高めることかできます。

NFCFelica ライブラリでは、機種を問わず、すべての関数を用意し、アプリケーションから見た「仮想マシン」としての振る舞いを提供します。
NFCFelica ライブラリの各関数は、アプリケーションからの要求に対して、対象のデバイス機能が制御できない場合は、「未サポートエラー」を返します。また搭載デバイスの機能差によって利用できないパラメータを設定した場合は、「パラメータエラー」を返します。

※ NFCFelica ライブラリは、アプリケーションのソースコード互換性の向上を目的としたライブラリであり、搭載デバイスの機能互換性を保障するものではありません。
「未サポートエラー」および「パラメータエラー」を正しく判定し、操作者に対して機能が未サポートである旨を通知する、あるいは処理そのものを無効としてください。

2. 動作環境

NFCFelica ライブラリの動作環境を以下に示します。

対象機種

- DT-5300
- DT-X8
- IT-9000

対象 OS

- Microsoft WindowsCE 6.0
- Microsoft WindowsMobile 6.5

開発環境とプログラミング言語

開発環境	Visual C++	Visual Basic, Visual C#
Microsoft Visual Studio 2005 + SP1	○	○
Microsoft Visual Studio 2008 + SP1	○	○

提供ファイル

ファイル	Visual C++	Visual Basic, Visual C#
NFCFelicaLib.h	○	-
NFCFelicaLib.lib	○	-
NFCFelicaLib.dll	○	○
NFCFelicaLibNet.dll	-	○

使用方法

Visual C++の場合

- プログラムソース内に NFCFelicaLib.h と NFCLib.h をインクルードし、リンカの依存ファイルとして NFCFelicaLib.lib を指定してください。
- NFCFelicaLib.dll は本体に内蔵されています。

Visual Basic または Visual C#の場合

- NFCFelicaLibNet.dll をプロジェクトの参照に追加してください。
- NFCFelicaLib.dll は本体に内蔵されています。
- NFCFelicaLibNet.dll を実行モジュールと同じフォルダーにコピーしてください。

名前空間とクラス

クラスライブラリ NFCFelicaLibNet.dll では、関数および定数の参照用として、下記のクラスが用意されています。

名前空間	クラス名	内容
CaLib	NFCFelicaLibNet.Api	関数参照用クラス
	NFCFelicaLibNet.Def	定数参照用クラス

クラス定義の詳細については、Microsoft Visual Studio で NFCFelicaLibNet.dll を参照設定し、オブジェクトブラウザで確認してください。

3. 関数

NFCFelica ライブラリの提供する関数は、以下のとおりです。

関数名	機能	DT-5300	DT-X8	IT-9000
NFCFelicaOpen	通信許可状態の設定	○	○	○
NFCFelicaClose	通信禁止状態の設定	○	○	○
NFCFelicaPolling	通信範囲内の FeliCa カードの検索および UID の取得	○	○	○
NFCFelicaStopPolling	通信範囲内の FeliCa カード検索を停止	○	○	○
NFCFelicaGetCardResponse	起動した FeliCa カードの詳細情報を取得	○	○	○
NFCFelicaRadioOff	電波送信を停止	○	○	○
NFCFelicaRead	指定したアドレスのデータ読み出し（16 バイト）	○	○	○
NFCFelicaWrite	指定したアドレスにデータ書き込み（16 バイト）	○	○	○
NFCFelicaSetEventNotification	電波自動停止のタイミング通知方法の設定	○	○	○
NFCFelicaGetEventNotification	電波自動停止のタイミング通知方法の取得	○	○	○
NFCFelicaSetAutoRadioOff	電波自動停止までの時間の設定	○	○	○
NFCFelicaGetAutoRadioOff	電波自動停止までの時間の取得	○	○	○
NFCFelicaSetPollingMode	IC カードの検索方法を設定	-	○	○
NFCFelicaGetPollingMode	IC カードの検索方法を取得	-	○	○
NFCFelicaGetCardResponseEx	起動した IC カードの詳細情報を取得	-	○	○
NFCFelicaSamOpen	SAM カードコントローラの電源の ON	-	-	○
NFCFelicaSamClose	SAM カードコントローラの電源の OFF	-	-	○
NFCFelicaSamPowerUp	指定したスロット番号の SAM カードの電源の ON	-	-	○
NFCFelicaSamPowerDown	指定されたスロット番号の SAM カードの電源の OFF	-	-	○
NFCFelicaSamAuthentication	NFCFelica ライブラリと SAM カードとの相互認証※	-	-	○
NFCFelicaAuthentication	FeliCa カードと SAM カードとの相互認証※	-	-	○
NFCFelicaReadWithEncryption	FeliCa カードのセキュリティ領域のデータの読み出し※	-	-	○
NFCFelicaWriteWithEncryption	FeliCa カードのセキュリティ領域にデータの書き込み※	-	-	○

○ 関数サポート

－ 関数未サポート

※ FeliCa 用 SAM カード（RC-S251）が必要です

■関数呼び出し手順

アプリケーション起動時

1. NFCFelicaOpen 関数により、NFC デバイスの電源を ON にします。※1
2. NFCFelicaSamOpen 関数により、SAM スロットの電源を ON にします。※2
3. NFCFelicaSamPowerUp 関数により、SAM スロットに挿入した SAM カード（RC-S251）の電源を ON にします。※3
4. NFCFelicaSamAuthentication 関数により、NFCFelica ライブラリと SAM カードとの相互認証を行ないます。

※ 2 から 4 の手順は FeliCa カードのセキュリティ領域にアクセスする場合にのみ行ないます。

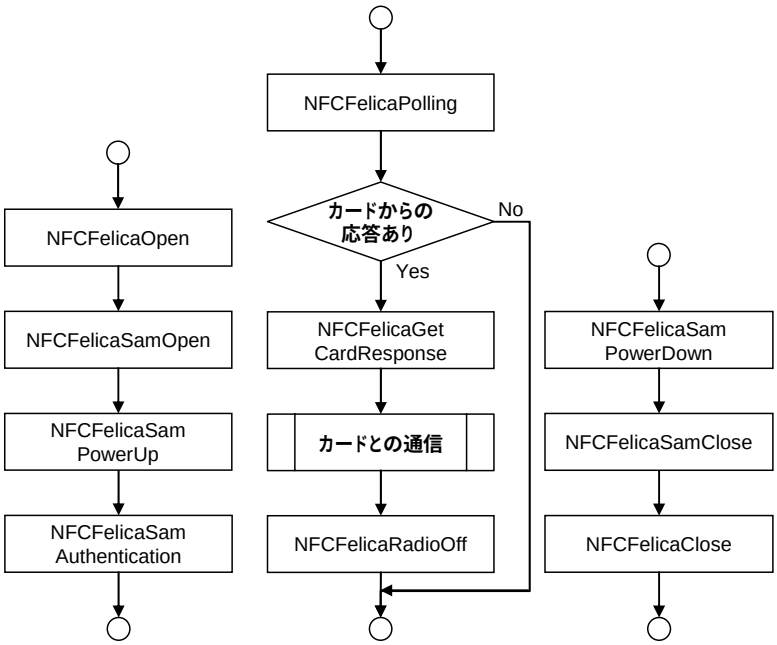
FeliCa カードとの通信時

1. 通信処理開始時に、NFCFelicaPolling 関数により電波送信を開始し、通信可能範囲内にある FeliCa カードを検索します。
2. FeliCa カードから応答がある場合、NFCFelicaGetCardResponse 関数により、応答情報を取得します。
3. FeliCa カードとの通信を行います。(次ページを参照)
4. FeliCa カードとの通信が終了した場合は、NFCFelicaRadioOff 関数により、電波出力を停止します。

アプリケーション終了時

1. NFCFelicaSamPowerDown 関数により、SAM スロットに挿入した SAM カードの電源を OFF にします。
2. NFCFelicaSamClose 関数により、SAM スロットの電源を OFF にします。
3. NFCFelicaClose 関数により、NFC デバイスの電源を OFF にします。

※ 1 から 2 の手順は FeliCa カードのセキュリティ領域にアクセスする場合にのみ行ないます。



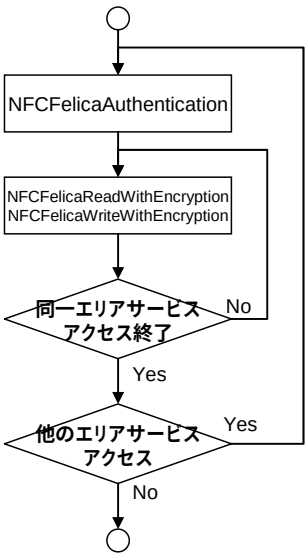
※1 FeliCa カードと通信していないとき、NFC デバイスはスタンバイモードとなるため、ほとんど電力を使用しません。

※2 SAM カード(RC-S251)の電源が OFF のとき、SAM コントローラはスタンバイモードとなるため、ほとんど電力を使用しません。

※3 SAM カード(RC-S251)の電源が ON のときは約 15mA の電力を消費します。電力の使用を抑えたい場合、通常は SAM カードの電源を OFF にし、SAM カードにアクセスするときだけ電源を ON にしてください。

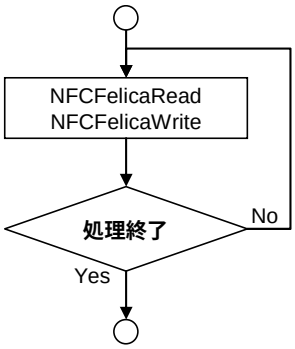
FeliCa カードとの通信(セキュリティ領域にアクセスする場合)

1. NFCFelicaAuthentication 関数により、FeliCa カードと SAM カード（RC-S251）の相互認証を行ないます。相互認証に成功すると、セキュリティ領域にアクセスできるようになります。
2. NFCFelicaReadWithEncryption 関数または NFCFelicaWriteWithEncryption 関数を実行し、データアクセスを実行します。
3. 同一エリア･サービス内の他のブロックにアクセスする場合は、2.に戻って処理を繰り返します。
4. 他のエリアまたはサービスにアクセスする場合は 1.に戻って処理を繰り返します。



FeliCa カードとの通信(非セキュリティ領域にアクセスする場合)

1. NFCFelicaRead 関数または NFCFelicaWrite 関数を実行します。



3.1 NFCFelicaOpen

NFC ドライバを通信許可状態（Open 状態）にし、NFC デバイスの電源を ON にします。

```
[C++]
int NFCFelicaOpen(
    HWND  hWnd
)
```

```
 [Visual Basic]
Public Shared Function NFCFelicaOpen(
    ByVal hWnd As IntPtr _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaOpen(
    IntPtr hWnd
)
```

解説

本関数は、NFC ドライバを通信許可状態（Open 状態）にし、NFC デバイスの電源を ON にします。
この状態はNFCFelicaClose関数を実行するまで有効です。
Open 状態時に、NFCFelicaPolling関数を実行すると、通信を開始します。

パラメータ

hWnd
アプリケーションのウィンドウハンドルを指定します。
電波自動停止が有効、かつ、イベント通知方法がメッセージの場合、指定したウィンドウハンドルに対して、メッセージを送信します。
NULL を指定した場合は、BROADCAST に対してメッセージを送信します。

戻り値

以下の値を返します。

NFC_OK	:	正常終了
NFC_PON	:	オープン済み※
NFC_NOT_DEVICE	:	NFC ドライバエラー DeviceEmulator では発生しません
NFC_ERROR_MODULE	:	モジュール未応答エラー DeviceEmulator では発生しません

※ 本関数を同一プロセス内で 2 回以上呼び出した場合は正常終了を返します

3.2 NFCFelicaClose

NFC ドライバを通信禁止状態（Close 状態）にし、NFC デバイスの電源を OFF にします。

```
[C++]
int NFCFelicaClose()
```

```
[Visual Basic]
Public Shared Function NFCFelicaClose() As Int32
```

```
[C#]
public static Int32 NFCFelicaClose()
```

解説

本関数は、NFC ドライバを通信禁止状態（Close 状態）にし、NFC デバイスの電源を OFF にします。

パラメータ

なし

戻り値

以下の値を返します。

NFC_OK	： 正常終了
NFC_NOT_DEVICE	： NFC ドライバエラー DeviceEmulator では発生しません

3.3 NFCFelicaPolling

通信可能範囲内にある FeliCa カードを検索します。

```
[C++]
int NFCFelicaPolling(
    DWORD  dwTimeout,
    BOOL   (*fpCallBack)(void),
    DWORD  dwSystemCode,
    DWORD  dwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaPolling( _
    ByVal dwTimeout     As Int32, _
    ByVal fpCallBack    As IntPtr, _
    ByVal dwSystemCode  As Int32, _
    ByVal dwReserved    As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaPolling(
    Int32      dwTimeout,
    IntPtr     fpCallBack,
    Int32      dwSystemCode,
    Int32      dwReserved
)
```

解説

本関数は、通信可能範囲内にある FeliCa カードを検索します。
FeliCa カードを発見した場合は、その FeliCa カードを起動し、データ通信可能な状態にします。
本関数は FeliCa カードを発見する、指定したタイムアウト時間経過する、または、指定したコールバック関数が FALSE を返すまで、通信範囲内の IC カードを検索します。
DeviceEmulator ではパラメータチェックのみを行います。

パラメータ

dwTimeout
FeliCa カードが起動するまでのタイムアウト時間を 100～60,000（msec 単位）の範囲で指定します。
また、0 を指定した場合は、タイムアウトなしで FeliCa カードを検索します。

fpCallBack
FeliCa カードの検索を続行するかどうかを判定するコールバック関数を指定します。
コールバック関数が TRUE を返す場合は処理を続行し、FALSE を返す場合は処理を停止します。
また、NULL を指定した場合は、常に続行します。

dwSystemCode
起動する FeliCa カードのシステムコードを指定します。
すべてのシステムコードの FeliCa カードを起動する場合は 0xFFFF を指定してください。

dwReserved

現在のバージョンではこの引数を使用しません。0 を指定してください。

戻り値

以下の値を返します。

NFC_OK	:	正常終了
NFC_NOT_DEVICE	:	NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	:	未オープンエラー
NFC_PRM	:	パラメータエラー
NFC_ERROR_TIMEOUT	:	タイムアウトエラー DeviceEmulator では発生しません
NFC_ERROR_CALLBACK	:	コールバック関数エラー DeviceEmulator では発生しません
NFC_ERROR_MODULE	:	モジュール未応答エラー DeviceEmulator では発生しません
NFC_ERROR_STOP	:	停止関数による中断エラー DeviceEmulator では発生しません

3.4 NFCFelicaStopPolling

通信可能範囲内にある FeliCa カードの検索を停止します。

```
[C++]
int NFCFelicaStopPolling()
```

```
[Visual Basic]
Public Shared Function NFCFelicaStopPolling() As Int32
```

```
[C#]
public static Int32 NFCFelicaStopPolling()
```

解説

本関数は、通信可能範囲内にある IC カードの検索を停止します。
コールバック関数を指定しないでNFCFelicaPolling関数を実行した場合は、本関数を実行することにより検索を停止することができます。

パラメータ

なし

戻り値

以下の値を返します。

NFC_OK	:	正常終了
NFC_NOT_DEVICE	:	NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	:	未オープンエラー

3.5 NFCFelicaGetCardResponse

起動した FeliCa カードの応答情報を取得します。

```
[C++]
int NFCFelicaGetCardResponse(
    BYTE   *pIDm,
    BYTE   *pPMm,
    DWORD  *pSystemCode,
    DWORD  dwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaGetCardResponse( _
    ByVal pIDm         As Byte(), _
    ByVal pPMm         As Byte(), _
    ByRef pSystemCode  As Int32, _
    ByVal dwReserved   As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaGetCardResponse(
    Byte[]     pIDm,
    Byte[]     pIDm,
    ref Int32  pSystemCode,
    Int32      dwReserved
)
```

解説

NFCFelicaPolling関数成功後に本関数を実行すると、起動した FeliCa カードの応答情報を取得します。
応答情報は FeliCa カード起動成功時にドライバに記憶し、本関数によりドライバにある応答情報を取得します。
DeviceEmulator では、パラメータチェックのみを行います。

パラメータ

pIDm
起動に成功した FeliCa カードの IDm を取得します。
8 バイト領域のポインタを指定してください。

pPMm
起動に成功した FeliCa カードの PMm を取得します。
8 バイト領域のポインタを指定してください。

pSystemCode
起動に成功した FeliCa カードのシステムコードを取得します。

dwReserved

現在のバージョンではこの引数を使用しません。0 を指定してください。

戻り値

以下の値を返します。

NFC_OK	： 正常終了
NFC_NOT_DEVICE	： NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	： 未オープンエラー
NFC_PRM	： パラメータエラー

3.6 NFCFelicaRadioOff

NFC モジュールの電波送信を停止します。

```
[C++]
int NFCFelicaRadioOff()
```

```
[Visual Basic]
Public Shared Function NFCFelicaRadioOff() As Int32
```

```
[C#]
public static Int32 NFCFelicaRadioOff()
```

解説

本関数は、NFC モジュールの電波送信を停止します。

パラメータ

なし

戻り値

以下の値を返します。

NFC_OK	:	正常終了
NFC_NOT_DEVICE	:	NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	:	未オープンエラー
NFC_ERROR_MODULE	:	モジュール未応答エラー DeviceEmulator では発生しません

3.7 NFCFelicaRead

起動した FeliCa カードのデータを読み出します。

```
[C++]
int NFCFelicaRead(
    DWORD  dwServiceCode,
    DWORD  dwBlockNumber,
    BYTE   *pData,
    DWORD  dwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaRead( _
    ByVal dwServiceCode   As Int32, _
    ByVal dwBlockNumber   As Int32, _
    ByVal pData           As Byte(), _
    ByVal dwReserved      As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaRead(
    Int32      dwServiceCode,
    Int32      dwBlockNumber,
    Byte[]     pData,
    Int32      dwReserved
)
```

解説

本関数は、起動した FeliCa カードに対して、指定したサービスコードおよびブロック番号のデータを読み出します。
認証なしでアクセス可能な領域についてのみデータの読み出しが可能です。
DeviceEmulator では、パラメータチェックのみを行います。

パラメータ

dwServiceCode
読み出す位置のサービスコードを指定します。(範囲 0x0000～0xFFFF)

dwBlockNumber
読み出す位置のブロック番号を指定します。(範囲 0 以上)

pData
読み出したブロックデータを取得します。
16 バイト領域のポインタを指定してください。

dwReserved
現在のバージョンではこの引数を使用しません。0 を指定してください。

戻り値

以下の値を返します。

NFC_OK	:	正常終了
NFC_NOT_DEVICE	:	NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	:	未オープンエラー
NFC_PRM	:	パラメータエラー
NFC_ERROR_TIMEOUT	:	タイムアウトエラー DeviceEmulator では発生しません
NFC_NOT_ACTIVATION	:	カード未起動エラー DeviceEmulator では発生しません
NFC_ERROR_MODULE	:	モジュール未応答エラー DeviceEmulator では発生しません
NFC_ERROR_SUSPEND	:	本体 OFF 発生エラー DeviceEmulator では発生しません
NFC_ERROR_AUTOOFF	:	電波自動停止エラー DeviceEmulator では発生しません

3.8 NFCFelicaWrite

起動した FeliCa カードにデータを書き込みます。

```
[C++]
int NFCFelicaWrite(
    DWORD  dwServiceCode,
    DWORD  dwBlockNumber,
    BYTE   *pData,
    DWORD  dwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaWrite( _
    ByVal dwServiceCode   As Int32, _
    ByVal dwBlockNumber   As Int32, _
    ByVal pData           As Byte(), _
    ByVal dwReserved      As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaWrite(
    Int32      dwServiceCode,
    Int32      dwBlockNumber,
    Byte[]     pData,
    Int32      dwReserved
)
```

解説

本関数は、起動した FeliCa カードに対して、指定したサービスコードおよびブロック番号のデータを書き込みます。
認証なしでアクセス可能な領域についてのみデータの読み出しが可能です。
DeviceEmulator では、パラメータチェックのみを行います。

パラメータ

dwServiceCode
書き込む位置のサービスコードを指定します。(範囲 0x0000～0xFFFF)

dwBlockNumber
書き込む位置のブロック番号を指定します。(範囲 0 以上)

pData
書き込むブロックデータを指定します。
16 バイト領域のポインタを指定してください。

dwReserved
現在のバージョンではこの引数を使用しません。0 を指定してください。

戻り値

以下の値を返します。

NFC_OK	:	正常終了
NFC_NOT_DEVICE	:	NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	:	未オープンエラー
NFC_PRM	:	パラメータエラー
NFC_ERROR_TIMEOUT	:	タイムアウトエラー DeviceEmulator では発生しません
NFC_NOT_ACTIVATION	:	カード未起動エラー DeviceEmulator では発生しません
NFC_ERROR_MODULE	:	モジュール未応答エラー DeviceEmulator では発生しません
NFC_ERROR_SUSPEND	:	本体 OFF 発生エラー DeviceEmulator では発生しません
NFC_ERROR_AUTOOFF	:	電波自動停止エラー DeviceEmulator では発生しません

3.9 NFCFelicaSetEventNotification

電波自動停止のタイミング通知方法を設定します。

```
[C++]
int NFCFelicaSetEventNotification(
    DWORD dwMode
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaSetEventNotification(
    ByVal dwMode As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaSetEventNotification(
    Int32 dwMode
)
```

解説

本関数は、電波自動停止のタイミング通知方法を設定します。

■ ウィンドウメッセージ通知
WM_NFC_AUTORADIOOFF(WM_USER + 0x580)のウィンドウメッセージを指定したウィンドウハンドルに対して送信します。

■ イベント通知
電波自動停止時に発行されるイベントは“NFCEventAutoRadioOff”です。WindowsCE では、名前は Unicode のため、プログラム上では TEXT(“NFCEventAutoRadioOff”)と指定します。

パラメータ

dwMode
電波自動停止のタイミング通知方法を指定します。

NFC_DISABLE	： 通知無効（デフォルト）
NFC_MESSAGE	： ウィンドウメッセージ通知
NFC_EVENT	： イベント通知

戻り値

以下の値を返します。

NFC_OK	： 正常終了
NFC_NOT_DEVICE	： NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	： 未オープンエラー
NFC_PRM	： パラメータエラー

3.10 NFCFelicaGetEventNotification

電波自動停止のタイミング通知方法を取得します。

```
[C++]
int NFCFelicaGetEventNotification(
    DWORD *pMode
)
```

```
[Visual Basic]
Public Shared Function NFCFeliacGetEventNotification(
    ByRef pMode As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaGetEventNotification(
    ref Int32 pMode
)
```

解説

本関数は、電波自動停止のタイミング通知方法を取得します。

パラメータ

pMode

電波自動停止のタイミング通知方法を取得します。取得する値の詳細については、NFCFelicaSetEventNotification関数を参照してください。

戻り値

以下の値を返します。

NFC_OK	： 正常終了
NFC_NOT_DEVICE	： NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	： 未オープンエラー
NFC_PRM	： パラメータエラー

3.11 NFCFelicaSetAutoRadioOff

電波自動停止までの時間を設定します。

```
[C++]
int NFCFelicaSetAutoRadioOff(
    DWORD dwTimeout
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaSetAutoRadioOff(
    ByVal dwTimeout As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaSetAutoRadioOff(
    Int32 dwTimeout
)
```

解説

本関数は、電波自動停止までの時間を設定します。

パラメータ

Timeout

電波自動停止までの時間を 100～60,000（msec 単位）の範囲で指定します（デフォルト：1,000）。
また、0 を指定した場合は、電波自動停止が無効となります。

戻り値

以下の値を返します。

NFC_OK	： 正常終了
NFC_NOT_DEVICE	： NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	： 未オープンエラー
NFC_PRM	： パラメータエラー

3.12 NFCFelicaGetAutoRadioOff

電波自動停止までの時間を取得します。

```
[C++]
int NFCFelicaGetAutoRadioOff(
    DWORD *pTimeout
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaGetAutoRadioOff(
    ByRef pTimeout As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaGetAutoRadioOff(
    ref Int32 pTimeout
)
```

解説

本関数は、電波自動停止までの時間を取得します。

パラメータ

Timeout

電波自動停止までの時間を取得します。取得する値の詳細については、NFCFelicaSetAutoRadioOff 関数を参照してください。

戻り値

以下の値を返します。

NFC_OK	: 正常終了
NFC_NOT_DEVICE	: NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	: 未オープンエラー
NFC_PRM	: パラメータエラー

3.13 NFCFelicaSetPollingMode

IC カードの検索方式を設定します。

```
[C++]
int NFCFelicaSetPollingMode(
    DWORD  dwMode,
    DWORD  dwNum,
    DWORD  dwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaSetPollingMode( _
    ByVal dwMode     As Int32, _
    ByVal dwNum      As Int32, _
    ByVal dwReserved As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaSetPollingMode(
    Int32  dwMode,
    Int32  dwNum,
    Int32  dwReserved
)
```

解説

本関数は、IC カードの検索方式を設定します。

パラメータ

dwMode

IC カードの検索方式を指定します。

NFC_PLMODE_NORMAL	： 通常起動（デフォルト）
NFC_PLMODE_MULTISTEP	： 多段起動
NFC_PLMODE_MULTISTEP2	： 多段起動 2
NFC_PLMODE_PACKAGE	： 一括起動

dwNum

多段起動時の段数を指定します。設定範囲は検索方式により異なります。

NFC_PLMODE_NORMAL 指定時	： 0 を指定してください
NFC_PLMODE_MULTISTEP 指定時	： 2～100
NFC_PLMODE_MULTISTEP2 指定時	： 2～100
NFC_PLMODE_PACKAGE 指定時	： 2 を指定してください

dwReserved

現在のバージョンではこの引数を使用しません。0 を指定してください。

戻り値

以下の値を返します。

NFC_OK　　：正常終了
NFC_NOT_DEVICE　　：NFC ドライバエラー
　　DeviceEmulator では発生しません
NFC_POF　　：未オープンエラー
NFC_PRM　　：パラメータエラー

補足

■ IC カードの検索方式
通常起動　　：1 回の検索で 1 枚の IC カードを起動します
多段起動/　　：段数に指定した回数まで異なる IC カードを連続して起動します
多段起動 2　　（1 回の検索で 1 枚しか起動することができません）※
一括起動　　：1 回の検索で同一タイプの複数枚の IC カードを起動します
　　段数に指定した回数まで検索を行います
※ 注意
IC カードを 1 つ起動するたびに、起動した IC カードの Uid をドライバに記録し、その記録した IC カードと重複する IC カードの二重起動を防止します。この記録は、指定した枚数の IC カードを起動したとき、タイムアウト時間を経過したとき、コールバック関数が FALSE を返したとき、およびNFCFelicaStopPolling関数を実行したときにクリアします。

3.14 NFCFelicaGetPollingMode

IC カードの検索方式を取得します。

```
[C++]
int NFCFelicaGetPollingMode(
    DWORD  *pdwMode,
    DWORD  *pdwNum,
    DWORD  *pdwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaGetPollingMode( _
    ByRef pdwMode     As Int32, _
    ByRef pdwNum      As Int32, _
    ByRef pdwReserved As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaGetPollingMode(
    ref Int32  pdwMode,
    ref Int32  pdwNum,
    ref Int32  pdwReserved
)
```

解説

本関数は、IC カードの検索方式を取得します。

パラメータ

pdwMode
IC カードの検索方式を取得します。取得する値の詳細については、NFCFelicaSetPollingMode関数を参照してください。

pdwNum
多段起動時の段数を取得します。取得する値の詳細については、NFCFelicaSetPollingMode関数を参照してください。

pdwReserved
現在のバージョンではこの引数を使用しません。NULL を指定してください。

戻り値

以下の値を返します。

NFC_OK	: 正常終了
NFC_NOT_DEVICE	: NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	: 未オープンエラー
NFC_PRM	: パラメータエラー

3.15 NFCFelicaGetCardResponseEx

起動した IC カードの応答情報を取得します。

```
[C++]
int NFCFelicaGetCardResponseEx(
    BYTE   *pbyIDm,
    BYTE   *pbyPMm,
    DWORD  *pdwSystemCode,
    DWORD  *pdwDiscoveredNum,
    DWORD  *pdwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaGetCardResponseEx( _
    ByVal pbyIDm          As Byte(), _
    ByVal pbypMm          As Byte(), _
    ByRef pdwSystemCode   As Int32, _
    ByRef pdwDiscoveredNum As Int32, _
    ByRef pdwReserved     As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaGetCardResponseEx(
    Byte[]     pbyIDm,
    Byte[]     pbyPMm,
    ref Int32  pdwSystemCode,
    ref Int32  pdwDiscoveredNum,
    ref Int32  pdwReserved
)
```

解説

NFCFelicaSetPollingMode関数で一括起動モードに設定した状態で、NFCFelicaPolling関数成功後に本関数を実行すると、起動した複数枚の IC カードの応答情報を取得します。
応答情報は IC カード起動成功時にドライバに記憶し、本関数によりドライバにある応答情報を取得します。
DeviceEmulator では、パラメータチェックのみを行います。

パラメータ

pbyIDm

起動に成功した IC カードの IDm を取得します。
バッファサイズは（8×NFCFelicaSetPollingMode関数の dwNum）以上確保してください。

8バイト	8バイト	…	8バイト
1枚目のIDm	2枚目のIDm	…	n枚目のIDm

pbyPMm

起動に成功した IC カードの PMm を取得します。
バッファサイズは（8×NFCFelicaSetPollingMode関数の dwNum）以上確保してください。

8バイト	8バイト	…	8バイト
1枚目のPMm	2枚目のPMm	…	n枚目のPMm

pdwSystemCode

起動に成功した IC カードのシステムコードを取得します。
サイズは（4×NFCFelicaSetPollingMode関数の dwNum）以上確保してください。

4バイト	4バイト	…	4バイト
1枚目の システムコード	2枚目の システムコード	…	n枚目の システムコード

pdwDiscoveredNum

NFCFelicaPolling関数で起動に成功した IC カードの枚数を取得します。
NFCFelicaRead関数の dwTargetNo に指定可能な値の最大は、（本パラメータで取得した値-1）となります。

例） 本パラメータで 3 を取得した場合
NFCFelicaRead関数の dwTargetNo に指定可能な値は 0～2 となります。

dwReserved

現在のバージョンではこの引数を使用しません。NULL を指定してください。

戻り値

以下の値を返します。

NFC_OK	： 正常終了
NFC_NOT_DEVICE	： NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	： 未オープンエラー
NFC_PRM	： パラメータエラー
NFC_ERROR_INVALID_ACCESS	： 電波自動停止タイマー動作中に実行エラー DeviceEmulator では発生しません

3.16 NFCFelicaSamOpen

SAM カードコントローラの電源を ON にします。

```
[C++]
int NFCFelicaSamOpen()
```

```
[Visual Basic]
Public Shared Function NFCFelicaSamOpen() As Int32
```

```
[C#]
public static Int32 NFCFelicaSamOpen()
```

解説

本関数は、SAM カードコントローラの電源を ON にし、SAM ドライバを通信許可状態（Open 状態）にします。

パラメータ

なし

戻り値

以下の値を返します。

NFC_OK	:	正常終了
NFC_PON	:	オープン済み※
NFC_NOT_DEVICE	:	SAM ドライバエラー DeviceEmulator では発生しません
NFC_ERROR_MODULE	:	モジュール未応答エラー DeviceEmulator では発生しません

※ 本関数を同一プロセス内で 2 回以上呼び出した場合は正常終了を返します

3.17 NFCFelicaSamClose

SAM カードコントローラの電源を OFF にします。

```
[C++]
int NFCFelicaSamClose()
```

```
[Visual Basic]
Public Shared Function NFCFelicaSamClose() As Int32
```

```
[C#]
public static Int32 NFCFelicaSamClose()
```

解説

本関数は、SAM カードコントローラの電源を OFF にし、SAM ドライバを通信禁止状態（Close 状態）にします。

パラメータ

なし

戻り値

以下の値を返します。

NFC_OK	:	正常終了
NFC_NOT_DEVICE	:	SAM ドライバエラー DeviceEmulator では発生しません

3.18 NFCFelicaSamPowerUp

指定したスロット番号の SAM カードの電源を ON にします。

```
[C++]
int NFCFelicaSamPowerUp(
    DWORD  dwSlotNumber,
    DWORD  dwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaSamPowerUp( _
    ByVal dwSlotNumber As Int32, _
    ByVal dwReserved   As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaSamPowerUp(
    Int32  dwSlotNumber,
    Int32  dwReserved
)
```

解説

本関数は、指定したスロット番号の SAM カードの電源を ON にします。

パラメータ

dwSlotNumber
通信対象の SAM カードを挿入したカードスロット番号を指定します。(1～カードスロット数)

dwReserved
現在のバージョンではこの引数を使用しません。0 を指定してください。

戻り値

以下の値を返します。

NFC_OK	: 正常終了
NFC_NOT_DEVICE	: NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	: 未オープンエラー
NFC_PRM	: パラメータエラー
NFC_ERROR_NOCARD	: SAM カード未挿入エラー
NFC_ERROR_MODULE	: モジュール未応答エラー
NFC_ERROR_RESPONSE	: カード異常応答発生エラー

3.19 NFCFelicaSamPowerDown

指定したスロット番号の SAM カードの電源を OFF にします。

```
[C++]
int NFCFelicaSamPowerDown(
    DWORD  dwSlotNumber,
    DWORD  dwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaSamPowerDown( _
    ByVal dwSlotNumber As Int32, _
    ByVal dwReserved   As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaSamPowerDown(
    Int32  dwSlotNumber,
    Int32  dwReserved
)
```

解説

本関数は、指定したスロット番号の SAM カードの電源を OFF にします。

パラメータ

dwSlotNumber
通信対象の SAM カードを挿入したカードスロット番号を指定します。(1～カードスロット数)

dwReserved
現在のバージョンではこの引数を使用しません。0 を指定してください。

戻り値

以下の値を返します。

NFC_OK	: 正常終了
NFC_NOT_DEVICE	: NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	: 未オープンエラー
NFC_PRM	: パラメータエラー
NFC_ERROR_NOCARD	: SAM カード未挿入エラー
NFC_ERROR_MODULE	: モジュール未応答エラー
NFC_ERROR_RESPONSE	: カード異常応答発生エラー

3.20 NFCFelicaSamAuthentication

NFCFelica ライブラリと SAM カードとの相互認証を実行します。

```
[C++]
int NFCFelicaSamAuthentication(
    DWORD  dwSlotNumber,
    DWORD  dwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaSamAuthentication( _
    ByVal dwSlotNumber As Int32, _
    ByVal dwReserved   As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaSamAuthentication(
    Int32  dwSlotNumber,
    Int32  dwReserved
)
```

解説

本関数は、NFCFelica ライブラリと SAM カード（RC-S251）との相互認証を実行します。
相互認証に成功すると、NFCFelicaAuthentication関数、NFCFelicaReadWithEncryption関数、およびNFCFelicaWriteWithEncryption関数が使用可能になります。
また、本関数実行後に再度本関数を実行すると、SAM カードとの通信回数をリセットします。

パラメータ

dwSlotNumber
通信対象の SAM カードを挿入したカードスロット番号を指定します。（1～カードスロット数）

dwReserved
現在のバージョンではこの引数を使用しません。0 を指定してください。

戻り値

以下の値を返します。

NFC_OK	： 正常終了
NFC_NOT_DEVICE	： NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	： 未オープンエラー
NFC_PRM	： パラメータエラー
NFC_ERROR_NOCARD	： SAM カード未挿入エラー
NFC_ERROR_ACTIVATION	： SAM カード未起動エラー
NFC_ERROR_TIMEOUT	： タイムアウトエラー
NFC_ERROR_SUSPEND	： 本体電源 OFF エラー
NFC_ERROR_RESPONSE	： カード異常応答発生エラー

注意

本関数で相互認証の成功後、端末を電源 OFF すると、電源 ON 時に状態がﾘｾｯﾄされ、相互認証前の状態に戻ります。

補足

■ 暗号化

本関数は、NFCFelica ﾗｲﾌﾞﾗﾘと SAM カード（RC-S251）と暗号化なしの相互認証を行います。そのため、SAM カードのコミュニケーション設定において、暗号化方式を”Disable CBC off”に設定してください。詳細は SAM カードのマニュアルを参照してください。

■ 通信回数

SAM カード（RC-S251）との通信回数が 65535 回を超えると、SAM カードとの通信ができなくなります。そのため、NFCFelicaAuthentication関数、NFCFelicaReadWithEncryption関数、および NFCFelicaWriteWithEncryption関数が NFC_ERROR_COUNT を返します。その場合、本関数を実行して通信回数をｸﾘｱしてください。

3.21 NFCFelicaAuthentication

FeliCa カードと SAM カードとの相互認証との相互認証を行います。

```
[C++]
int NFCFelicaAuthentication(
    DWORD  dwSlotNumber,
    DWORD  dwSystemCode,
    WORD   wSystemKeyVer,
    BYTE   byAreaNum,
    BYTE   *pbyAreaCode,
    BYTE   byServiceNum,
    BYTE   *pbyServiceCode,
    DWORD  dwTargetNo,
    DWORD  dwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaAuthentication( _
    ByVal  dwSlotNumber   As Int32, _
    ByVal  dwSystemCode   As Int32, _
    ByVal  wSystemKeyVer  As UInt16, _
    ByVal  byAreaNum      As Byte, _
    ByVal  pbyAreaCode    As Byte(), _
    ByVal  byServiceNum   As Byte, _
    ByVal  pbyServiceCode As Byte(), _
    ByVal  dwTargetNo     As Int32, _
    ByVal  dwReserved     As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaAuthentication(
    Int32  dwSlotNumber,
    Int32  dwSystemCode,
    UInt16 wSystemKeyVer,
    byte   byAreaNum,
    Byte[] pbyAreaCode,
    byte   byServiceNum,
    Byte[] pbyServiceCode,
    Int32  dwTargetNo,
    Int32  dwReserved
)
```

解説

本関数は、FeliCa カードと SAM カードとの相互認証との相互認証を行います。
相互認証に成功すると、FeliCa カードのセキュリティ領域にアクセスすることができます。

パラメータ

dwSlotNumber
通信対象の SAM カードを挿入したカードスロット番号を指定します。(1～カードスロット数)

dwSystemCode
NFCFelicaGetCardResponse関数またはNFCFelicaGetCardResponseEx関数で取得したシステムコードを指定します。

wSystemKeyVer
システム鍵バージョンを指定します。(リトルエンディアン)

byAreaNum
エリア数を指定します。(1～8)

pbyAreaCode
エリアコード/エリア鍵バージョンリスト(リトルエンディアン)を指定します。

byServiceNum
サービス数を指定します。(1～8)

pbyServiceCode
サービスコード/サービス鍵バージョンリスト(リトルエンディアン)を指定します。

dwTargetNo
通信したい IC カードに対応したカード番号を指定します。
通常は 0 を指定してください。NFCFelicaPolling関数で複数の IC カードの起動に成功した状態で 2 枚目以降の IC カードと通信する場合は 1 以上の値を指定してください。

dwReserved
現在のバージョンではこの引数を使用しません。0 を指定してください。

戻り値

以下の値を返します。

NFC_OK	: 正常終了
NFC_NOT_DEVICE	: NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	: 未オープンエラー
NFC_PRM	: パラメータエラー
NFC_ERROR_NOCARD	: SAM カード未挿入エラー
NFC_ERROR_ACTIVATION	: SAM カード未起動エラー
NFC_ERROR_TIMEOUT	: タイムアウトエラー
NFC_ERROR_SUSPEND	: 本体電源 OFF エラー
NFC_ERROR_RESPONSE	: カード異常応答発生エラー
NFC_ERROR_COUNT	: SAM 通信回数オーバー

使用例

SAM スロット番号:1
システムコード/システム鍵バージョン:0018h/0001h
エリアコード/エリア鍵バージョン:0000h/0001h
サービスコード/サービス鍵バージョン:1020h/0001h

```
dwSlotNumber    = 0x00000001;
```

```
dwSystemCode    = 0x00001800;
wSystemVer      = 0x0100;
byAreaNum       = 0x01;
pbyAreaCode     = {0x00, 0x00, 0x01, 0x00};    // エリアコード 2 バイト
                                               // 鍵バージョン 2 バイト
byServiceNum    = 0x01;
pbyServiceCode  = {0x20, 0x10, 0x01, 0x00};    // サービスコード 2 バイト
                                               // 鍵バージョン 2 バイト
dwTargetNo      = 0x00000000;
dwReserved      = 0x00000000;
```

3.22 NFCFelicaReadWithEncryption

FeliCa カードのセキュリティ領域のデータを読み出します。

```
[C++]
int NFCFelicaReadWithEncryption(
    DWORD  dwSlotNumber,
    BYTE   byBlockNum,
    BYTE   *pbyBlockList,
    BYTE   *pbyData,
    BYTE   *pbyActualNum,
    DWORD  dwTargetNo,
    DWORD  dwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaReadWithEncryption( _
    ByVal  dwSlotNumber   As Int32, _
    ByVal  byBlockNum     As Byte, _
    ByVal  pbyBlockList   As Byte(), _
    ByVal  pbyData        As Byte(), _
    ByRef  pbyServiceCode As Byte, _
    ByVal  dwTargetNo     As Int32, _
    ByVal  dwReserved     As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaReadWithEncryption(
    Int32    dwSlotNumber,
    Byte     byBlockNum,
    Byte[]   pbyBlockList,
    Byte[]   pbyData,
    ref Byte pbyActualNum,
    DWORD    dwTargetNo,
    DWORD    dwReserved
)
```

解説

本関数は、FeliCa カードのセキュリティ領域のデータを読み出します。

パラメータ

dwSlotNumber
通信対象の SAM カードを挿入したカードスロット番号を指定します。(1～カードスロット数)

byBlockNum
読み出し対象ブロック数を指定します。(1～12)

pbyBlockList
読み出し対象ブロックリストを指定します。36 バイトまで指定できます。

pbyData
読み出したブロックデータを取得します。(16 バイト×ブロック数)

pbyActualNum
読み出したブロック数を取得します。

dwTargetNo
通信したい IC カードに対応したカード番号を指定します。
通常は 0 を指定してください。NFCFelicaPolling関数で複数の IC カードの起動に成功した状態で 2 枚目以降の IC カードと通信する場合は 1 以上の値を指定してください。

dwReserved
現在のバージョンではこの引数を使用しません。0 を指定してください。

戻り値

以下の値を返します。

NFC_OK	: 正常終了
NFC_NOT_DEVICE	: NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	: 未オープンエラー
NFC_PRM	: パラメータエラー
NFC_ERROR_NOCARD	: SAM カード未挿入エラー
NFC_ERROR_ACTIVATION	: SAM カード未起動エラー
NFC_ERROR_TIMEOUT	: タイムアウトエラー
NFC_ERROR_SUSPEND	: 本体電源 OFF エラー
NFC_ERROR_RESPONSE	: カード異常応答発生エラー
NFC_ERROR_COUNT	: SAM 通信回数オーバー

使用例

0 ブロック目に対し、1 ブロック分 (16 バイト) のデータを読み出す場合

```
/* Input   */
dwSlotNumber = 0x00000001;
byBlockNum   = 0x01;
pbyBlockList = {0x80, 0x00}    // ブロック番号 0 の場合
dwTargetNo   = 0x00000000;
dwReserved   = 0x00000000;

/* Output  */
pbyData      = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07,
                0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F};
pbyActualNum = 0x01;
```

3.23 NFCFelicaWriteWithEncryption

FeliCa カードのセキュリティ領域にデータを書き込みます。

```
[C++]
int NFCFelicaWriteWithEncryption(
    DWORD  dwSlotNumber,
    BYTE   byBlockNum,
    BYTE   *pbyBlockList,
    BYTE   *pbyData,
    DWORD  dwTargetNo,
    DWORD  dwReserved
)
```

```
[Visual Basic]
Public Shared Function NFCFelicaWriteWithEncryption( _
    ByVal  dwSlotNumber   As Int32, _
    ByVal  byBlockNum     As Byte, _
    ByVal  pbyBlockList   As Byte(), _
    ByVal  pbyData        As Byte(), _
    ByVal  dwTargetNo     As Int32, _
    ByVal  dwReserved     As Int32 _
) As Int32
```

```
[C#]
public static Int32 NFCFelicaWriteWithEncryption(
    Int32   dwSlotNumber,
    Byte    byBlockNum,
    Byte[]  pbyBlockList,
    Byte[]  pbyData,
    DWORD   dwTargetNo,
    DWORD   dwReserved
)
```

解説

本関数は、FeliCa カードのセキュリティ領域にデータを書き込みます。

パラメータ

dwSlotNumber
通信対象の SAM カードを挿入したカードスロット番号を指定します。(1～カードスロット数)

byBlockNum
書き込み対象ブロック数を指定します。(1～8)

pbyBlockList
書き込み対象ブロックリストを指定します。24 バイトまで指定できます。

pbyData
書き込むブロックデータを取得します。(16 バイト×ブロック数)

dwTargetNo

通信したい IC カードに対応したカード番号を指定します。
通常は 0 を指定してください。NFCFelicaPolling関数で複数の IC カードの起動に成功した状態で 2 枚目以降の IC カードと通信する場合は 1 以上の値を指定してください。

dwReserved

現在のバージョンではこの引数を使用しません。0 を指定してください。

戻り値

以下の値を返します。

NFC_OK	: 正常終了
NFC_NOT_DEVICE	: NFC ドライバエラー DeviceEmulator では発生しません
NFC_POF	: 未オープンエラー
NFC_PRM	: パラメータエラー
NFC_ERROR_NOCARD	: SAM カード未挿入エラー
NFC_ERROR_ACTIVATION	: SAM カード未起動エラー
NFC_ERROR_TIMEOUT	: タイムアウトエラー
NFC_ERROR_SUSPEND	: 本体電源 OFF エラー
NFC_ERROR_RESPONSE	: カード異常応答発生エラー
NFC_ERROR_COUNT	: SAM 通信回数オーバー

使用例

0 ブロック目に対し、1 ブロック分（16 バイト）のデータを書き込む場合

```
dwSlotNumber = 0x00000001;
byBlockNum   = 0x01;
pbyBlockList = {0x80, 0x00}    // ブロック番号 0 の場合
pbyData      = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07,
                0x08, 0x09, 0x0A, 0x0B, 0x0C, 0x0D, 0x0E, 0x0F};
dwTargetNo   = 0x00000000;
dwReserved   = 0x00000000;
```

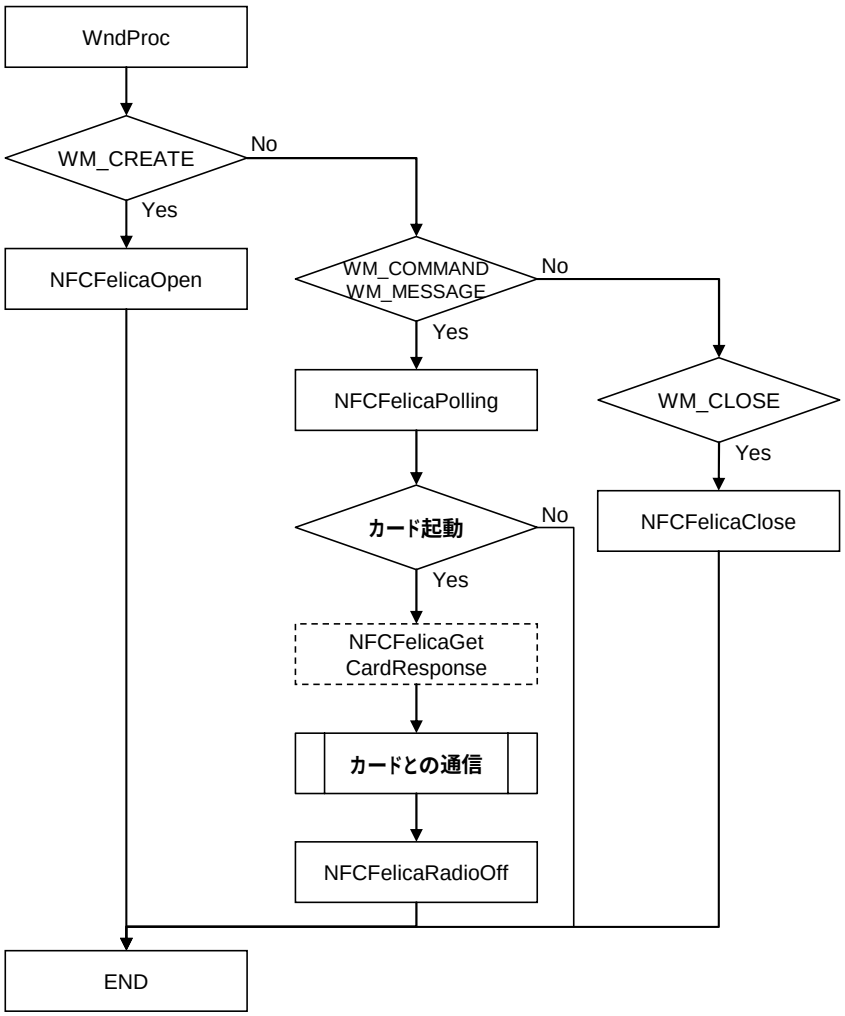
4. プログラミング上の注意点

4.1 電波停止の通知について

ウィンドウメッセージ通知を使用する場合

電波を手動で停止する場合

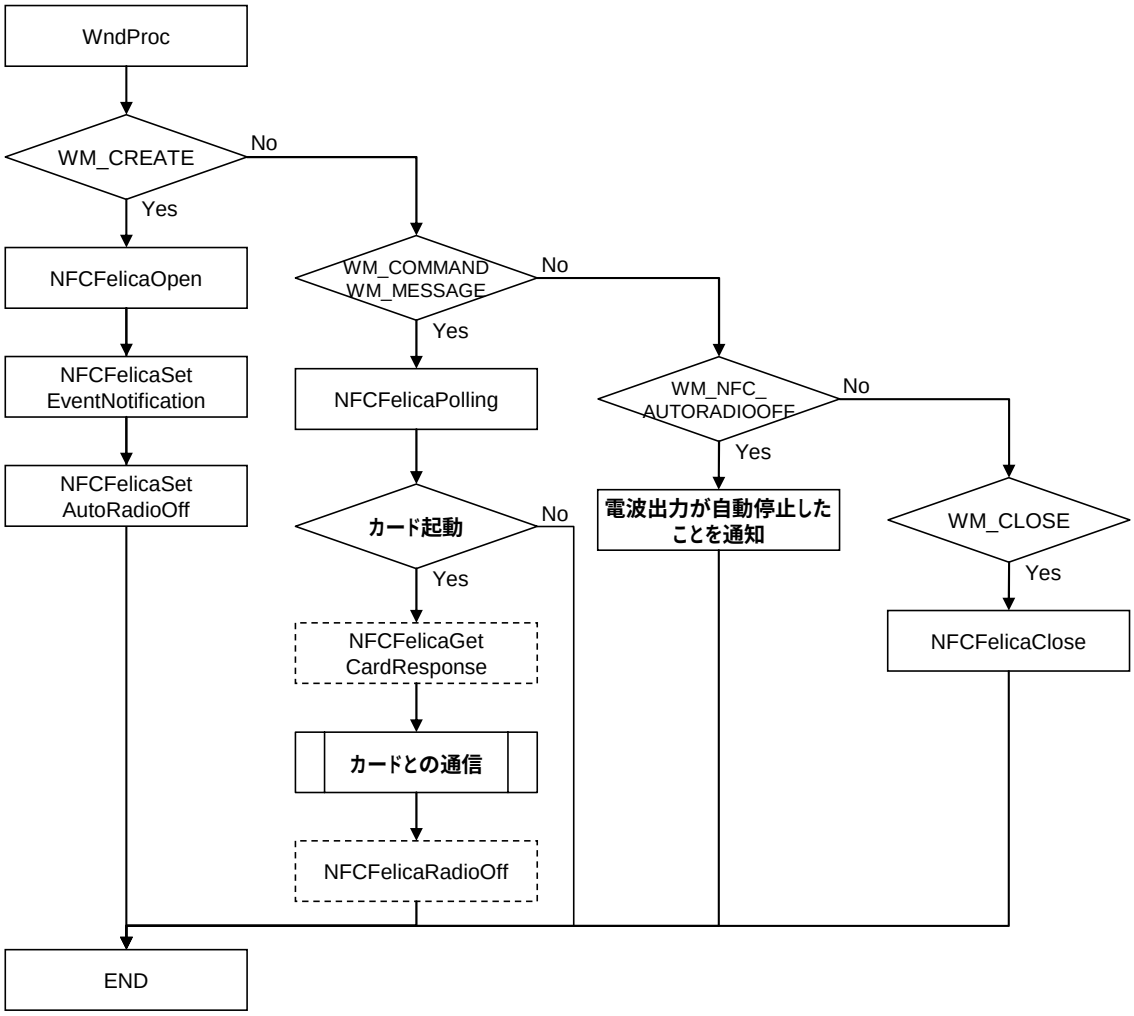
1. WM_CREATE メッセージを受け取った場合は、NFCFelicaOpen関数を実行し、読み取り待機状態にします。
2. WM_COMMAND、WM_KEYDOWN 等のメッセージを受け取った場合は、NFCFelicaPolling関数により、通信可能範囲内にある FeliCa カードを検索/起動します。
3. FeliCa カードの起動に成功、かつ、そのカードの詳細な情報が必要な場合は、NFCFelicaGetCardResponse関数により、応答情報を取得します。(任意)
4. FeliCa カードとの通信を行います※。
5. IC カードとの通信が終了した場合は、NFCFelicaRadioOff関数により、電波出力を停止します。
6. WM_CLOSE メッセージを受け取った場合は、NFCFelicaClose関数により、読み取り禁止状態にします。



※ FeliCa カードとの通信については、「FeliCa カードとの通信について」を参照してください。

電波を自動で停止し、停止タイミングを通知する場合

1. WM_CREATE メッセージを受け取った場合は、NFCFelicaOpen関数を実行し、読み取り待機状態にします。
2. NFCFelicaSetEventNotification関数により、ウィンドウメッセージ通知を有効に設定します。
3. NFCFelicaSetAutoRadioOff関数により、電波自動停止を有効に設定します。
4. WM_COMMAND、WM_KEYDOWN 等のメッセージを受け取った場合は、NFCFelicaPolling関数により、通信可能範囲内にある IC カードを検索/起動します。
5. FeliCa カードの起動に成功、かつ、そのカードの詳細な情報が必要な場合は、NFCFelicaGetCardResponse関数により、応答情報を取得します。(任意)
6. FeliCa カードとの通信を行います※。
7. FeliCa カードとの通信が終了した場合は、NFCFelicaRadioOff関数により、電波出力を停止します。(停止しない場合でも、通信を一定時間行わないと自動的に電波出力を停止します)
8. 電波出力の自動停止が発生したタイミングで WM_NFC_AUTORADIOOFF(WM_USER + 0x580)メッセージを受け取ることができます。このとき、電波出力が自動停止したことをユーザに通知することが可能です。
9. WM_CLOSE メッセージを受け取った場合は、NFCFelicaClose関数により、読み取り禁止状態にします。

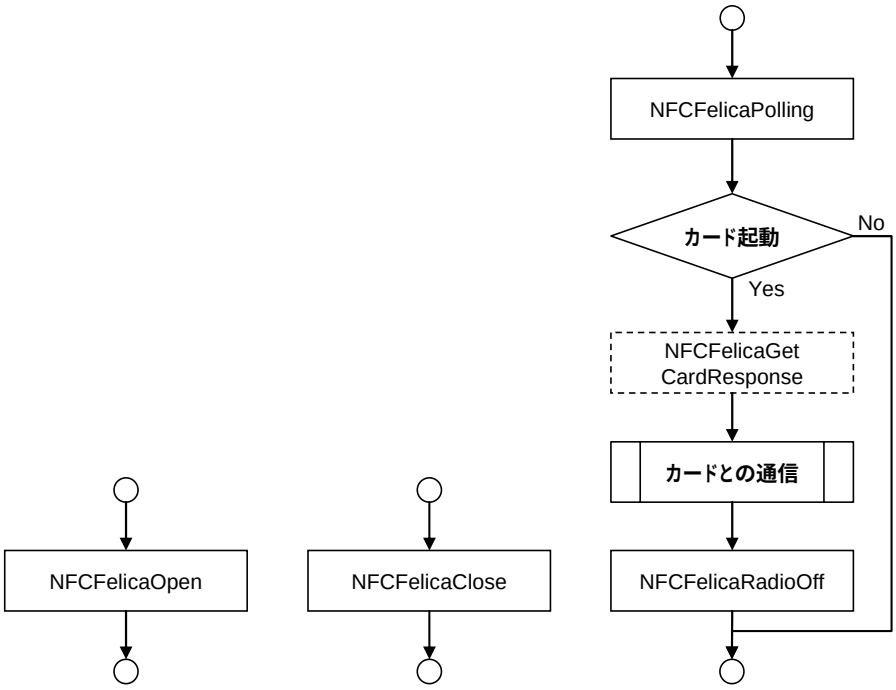


※ FeliCa カードとの通信については、「FeliCa カードとの通信について」を参照してください。

イベント通知を使用する場合

電波を手動で停止する場合

1. アプリケーション開始時に、NFCFelicaOpen関数により、読み取り待機状態にします。
2. 通信処理開始時に、NFCFelicaPolling関数により、通信可能範囲内にある FeliCa カードを検索/起動します。
3. FeliCa カードの起動に成功、かつ、そのカードの詳細な情報が必要な場合は、NFCFelicaGetCardResponse関数により、応答情報を取得します。（任意）
4. FeliCa カードとの通信を行います※。
5. FeliCa カードとの通信が終了した場合は、NFCFelicaRadioOff関数により、電波出力を停止します。
6. アプリケーション終了時に、NFCFelicaClose関数により、読み取り禁止状態にします。



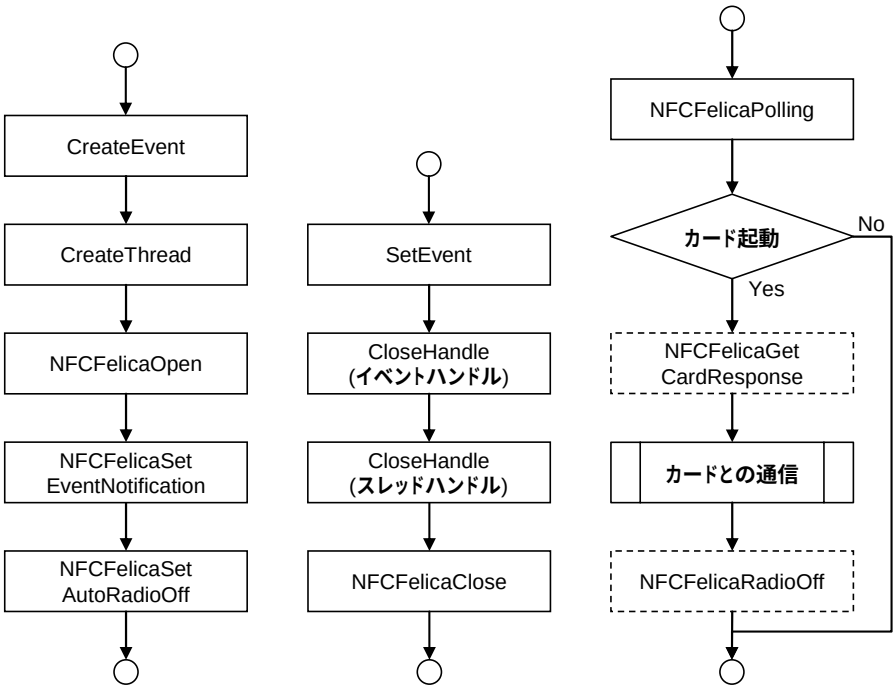
アプリケーション起動時　アプリケーション終了時　FeliCa カードとの通信実行時

※ FeliCa カードとの通信については、「FeliCa カードとの通信について」を参照してください。

電波を自動で停止し、停止タイミングを通知する場合

■ メインスレッド

1. アプリケーション開始時に、CreateEvent 関数により、電波自動停止タイミング通知イベントハンドルを作成します。
2. CreateThread 関数により、電波自動停止を監視するスレッドを作成します。
3. NFCFelicaOpen関数により、読み取り待機状態にします。
4. NFCFelicaSetEventNotification関数により、イベント通知を有効に設定します。
5. NFCFelicaSetAutoRadioOff関数により、電波自動停止を有効に設定します。
6. 通信処理開始時に、NFCFelicaPolling関数により、通信可能範囲内にある FeliCa カードを検索/起動します。
7. FeliCa カードの起動に成功、かつ、そのカードの詳細な情報が必要な場合は、NFCFelicaGetCardResponse関数により、応答情報を取得します。(任意)
8. FeliCa カードとの通信を行います※。
9. FeliCa カードとの通信が終了した場合は、NFCFelicaRadioOff関数により、電波出力を停止します。(停止しない場合でも、通信を一定時間行わないと自動的に電波出力を停止します)
10. アプリケーション終了時に、SetEvent 関数により、電波自動停止を監視するスレッドに対して通知を行います。
11. イベントハンドルとスレッドハンドルをクローズします。
12. NFCFelicaClose関数により、読み取り禁止状態にします。

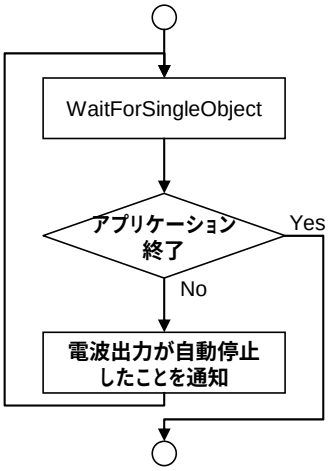


アプリケーション起動時　アプリケーション終了時　FeliCa カードとの通信実行時

※ FeliCa カードとの通信については、「FeliCa カードとの通信について」を参照してください。

■ NFCFelica スレッド

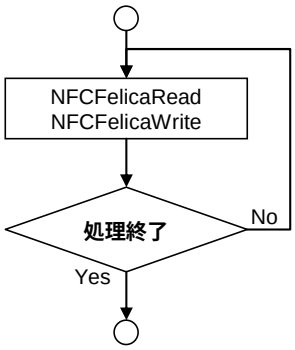
13. WaitForSingleObject 関数により、電波自動停止タイミング通知イベントハンドルに対して待機します。
14. アプリケーション終了時に通知イベントを受け取った場合、電波自動停止の監視を終了します。
15. 上記以外時に通知イベントを受け取った場合、電波出力が自動停止したことを通知することが可能です。



4.2 FeliCa カードとの通信について

以下は「電波停止の通知について」におけるカードとの通信部分の手順です。

1. NFCFelicaRead関数またはNFCFelicaWrite関数を実行します。

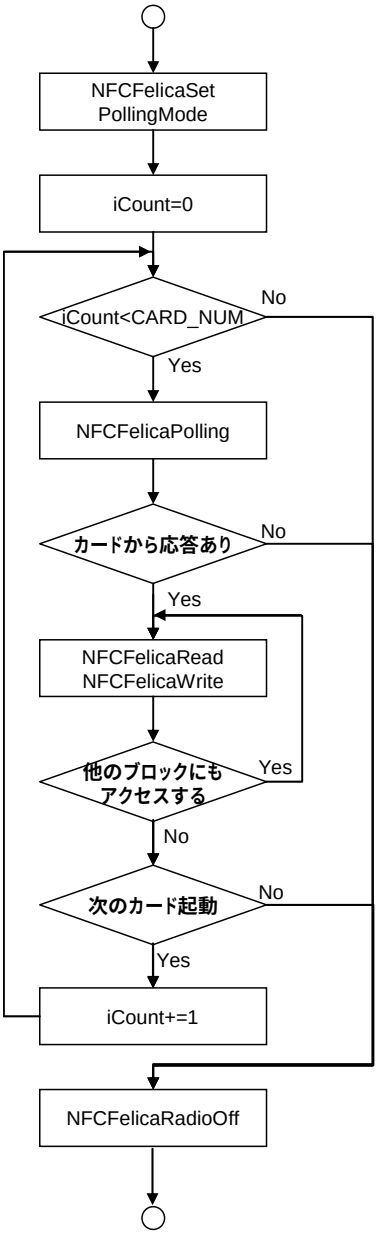


4.3 検索方式について

多段起動を使用する場合

FeliCa カードと通信する場合

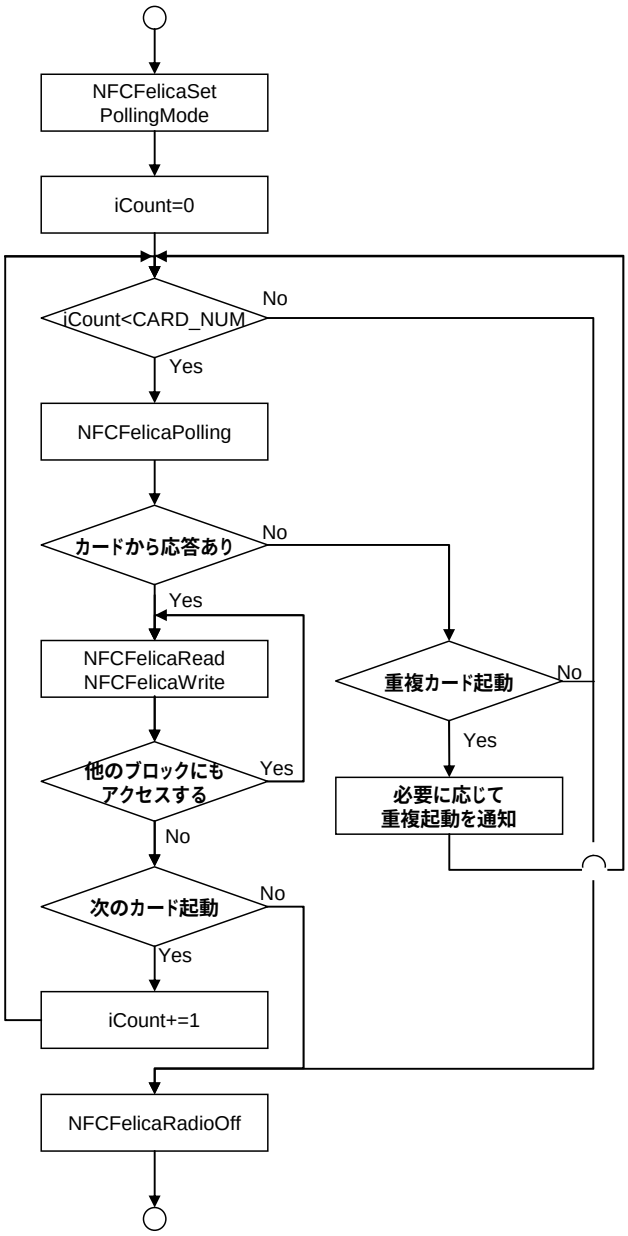
1. NFCFelicaSetPollingMode関数により、検索方式に多段起動（NFC_PLMODE_MULTISTEP）を、段数に連続起動するカード枚数 CARD_NUM を指定します。
2. iCount=0 をセットします。
3. iCount<CARD_NUM の場合、次の処理に進みます。
   CARD_NUM は連続起動する IC カードの枚数を表します。
4. NFCFelicaPolling関数により通信範囲内の IC カードを検索します。
5. IC カードから応答があった場合は、NFCFelicaRead関数またはNFCFelicaWrite関数により、データアクセスを実行します。
6. 他のブロックにアクセスする場合は、5.に戻って処理を繰り返します。
7. 次のカードを起動する場合、iCount に 1 を加算し、3.に戻って同様の処理を繰り返します。
8. 3.において、iCount が CARD_NUM より大きい場合、ループ処理を終了します。
9. NFCFelicaRadioOff関数により、電波を停止します。（電波を自動で停止する場合は、本手順は必要ありません。）



多段起動 2 を使用する場合

FeliCa カードと通信する場合

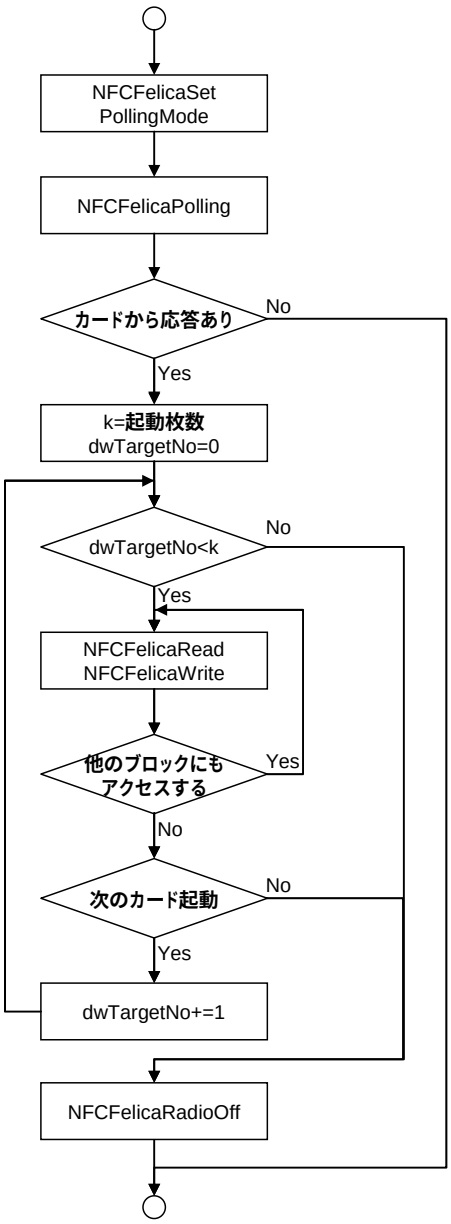
1. NFCFelicaSetPollingMode関数により、検索方式に多段起動（NFC_PLMODE_MULTISTEP2）を、段数に連続起動するカード枚数 CARD_NUM を指定します。
2. iCount=0 をセットします。
3. iCount<CARD_NUM の場合、次の処理に進みます。CARD_NUM は連続起動する IC カードの枚数を表します。
4. NFCFelicaPolling関数により通信範囲内のカードを検索します。
5. カードの起動に失敗し、NFCFelicaPolling 関数の戻り値が重複起動を表す場合、必要に応じて LED 等により重複起動を通知します。その後、3.に戻って処理を繰り返します。
6. IC カードから応答があった場合は、NFCFelicaRead関数または NFCFelicaWrite関数により、データアクセスを実行します。
7. 他のブロックにアクセスする場合は、6.に戻って処理を繰り返します。
8. 次のカードを起動する場合、iCount に 1 を加算し、3.に戻って同様の処理を繰り返します。
9. 3.において、iCount が CARD_NUM より大きい場合、ループ処理を終了します。
10. NFCFelicaRadioOff関数により、電波を停止します。（電波を自動で停止する場合は、本手順は必要ありません。）



一括起動を使用する場合

FeliCa カードと通信する場合

1. NFCFelicaSetPollingMode関数により、検索方式に一括起動（NFC_PLMODE_PACKAGE）を、段数に一括起動する枚数を指定します。
2. NFCFelicaPolling関数により通信範囲内のカードを検索します。
3. IC カードが起動したら次の処理に進みます。
4. k に起動した枚数を、dwTargetNo に 0 をセットします。（NFCFelicaRead関数やNFCFelicaWrite関数の引数）
5. dwTargetNo が k よりも小さい場合、次の処理に進みます。
6. NFCFelicaRead関数またはNFCFelicaWrite関数により、データアクセスを実行します。
7. 他のブロックにアクセスする場合は、6.に戻って処理を繰り返します。
8. 次のカードと通信する場合、dwTargetNo に 1 加算し、5.に戻って同様の処理を繰り返します。
9. 5.において、dwTargetNo が k よりも大きい場合、ループ処理を終了します。
10. NFCFelicaRadioOff関数により、電波を停止します。



カシオ計算機お問い合わせ窓口

製品に関する最新情報

●製品サポートサイト（カシオペア・ハンディターミナル）

http://casio.jp/support/ht/

製品の取扱い方法のお問い合わせ

●情報機器コールセンター

0570-022066
市内通話料金でご利用いただけます。
携帯電話･PHS 等をご利用の場合、**048-233-7241**

カシオ計算機株式会社
〒151-8543　東京都渋谷区本町 1-6-2
TEL 03-5334-4638(代)