AsReader P31N

SDK リファレンス ガイド V1.2

Asterisk, Inc.

変更履歴

No.	バージョン	変更内容	日付
1	1.0	新規作成	2019/05/05
2	1.1	１、付録Ⅱを更新 ２、SetStaticIP メソッドを追加	2019/06/18
3	1.2	GetSdkVersion メソッドを追加	2019/06/24

目　次

前提条件

Windows バージョン：

開発ソフトウェア：VisualStdio バージョン 2012+

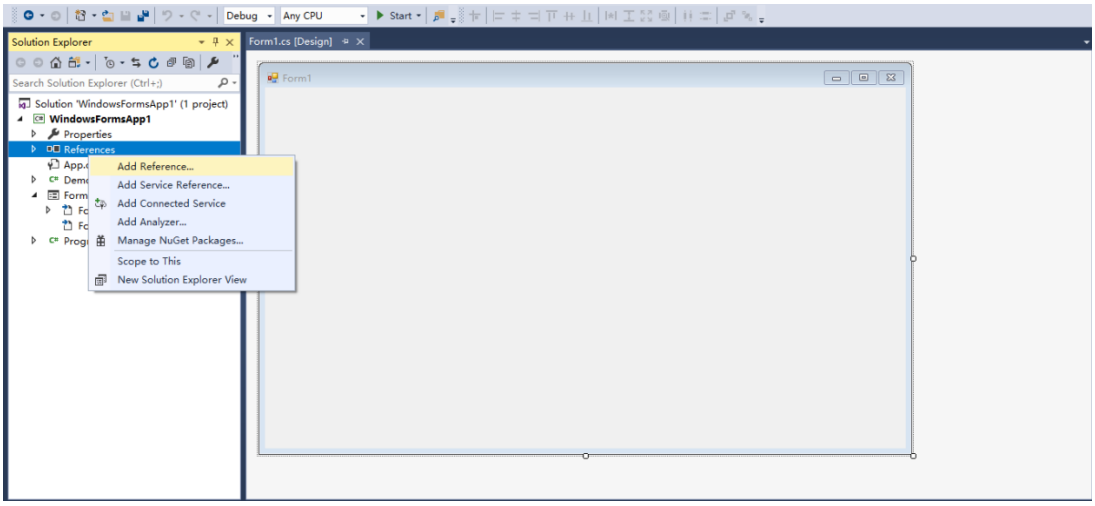
開発言語：C#

1 SDK の使用

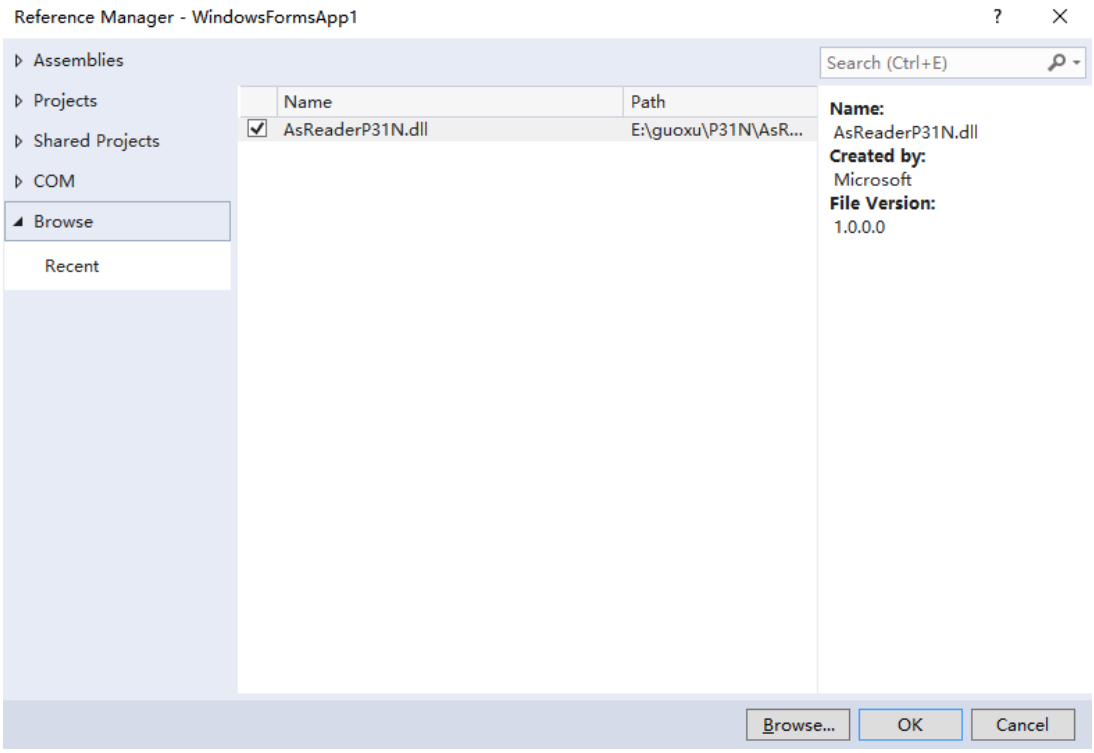
1.1 SDK 追加

1. プロジェクト（Windows フォームアプリケーション）を新規します。
   AsReaderP31N.dll をプロジェクトファイルにコピーします。

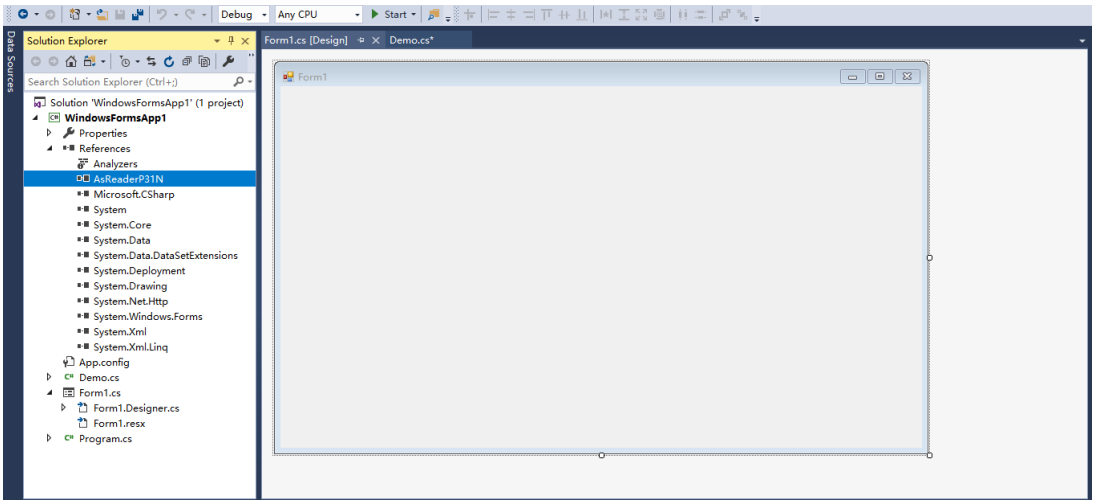
2. プロジェクトのインポートに AsReaderP31N.dll を追加します。
   インポート一覧で右クリックしてインポートを追加します。



3. 「ブラウズ」を選択し、プロジェクトパスに AaReaderP31N.dll を選択し、「確定」をクリックします。



4. 追加完了後、下図通り：

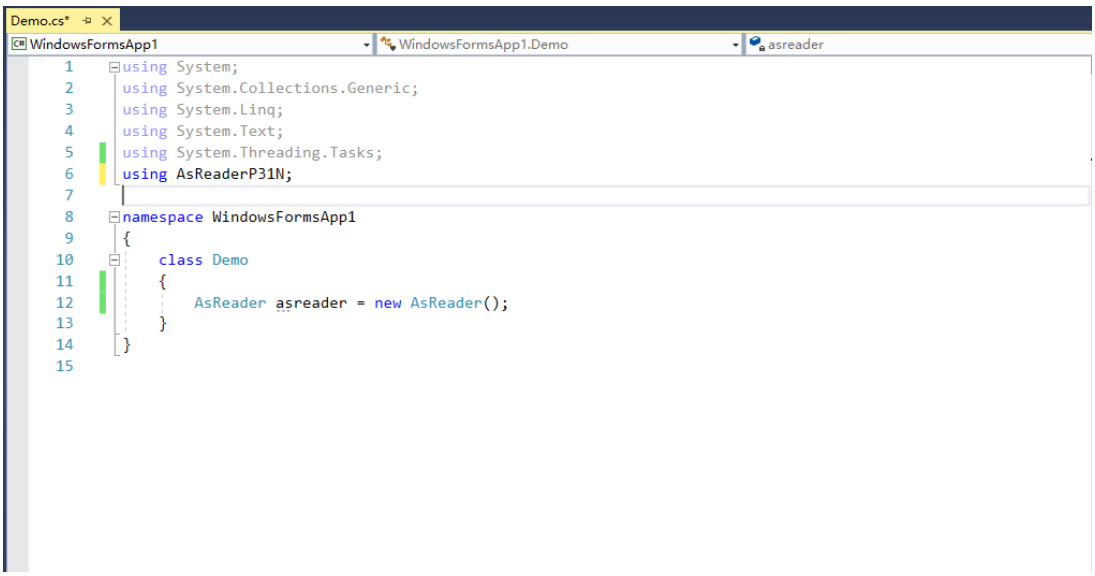


1.2 SDK インポート

1. 命名スペースをインポートします。

```
using AsReaderP31N;
```

2. オブジェクトをインスタンスします。



3. ConnectWithVCP メソッド（章 2.1.1 をご参照）を実行し、リーダライターを接続します。

注意：実行に成功した場合、０を返す。
実行に失敗した場合、１を返す。

```
asreader.ConnectWithVCP("COM1");
```

2 API 説明

2.1 AsReader Class

AsReader Class は RFID のインベントリー、読み取り、書き込み、ロックなどの API インターフェースを提供します。

2.1.1 ConnectWithVCP

関数名	UInt32 ConnectWithVCP(string comPort)		
引数名	IN/OUT	型	説明
comPort	IN	string	デバイスのポート番号
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： ポートを開き、ホストとデバイスとの通信接続をさせ、デバイスの構造体を初期化し、ホストはデバイスにすべての処理関数を登録します。			
例： ConnectWithVCP(“COM1”);			

2.1.2 ConnectWithTCP

関数名	UInt32 ConnectWithTCP(string ipAddress,string tcpPort)		
引数名	IN/OUT	型	説明
ipAddress	IN	string	デバイスの ip アドレス
tcpPort	IN	string	デバイスに接続するポート
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： TCP/IP とポートを通じて、ホストとデバイスとの通信接続をさせ、デバイス構造体を初期化し、ホストはデバイスにすべての処理関数を登録します。			

例： ConnectWithTCP(“192.168.1.100”、“9600”); 注意：TCP 通信ポート番号：5000 　　UDP 通信ポート番号：50001

2.1.3 Disconnect

関数名	UInt32 Disconnect()		
引数名	IN/OUT	型	説明
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： ホストとデバイスの接続を切断し、デバイスをリセットします。			
例：ポストとリーダライターの接続を切断します。 Disconnect();			

2.1.4 StartInventory

関数名	UInt32 StartInventory(bool an1,bool an2,bool an3,bool an4, bool an5,bool an6,bool an7,bool an8)		
引数名	IN/OUT	型	説明
an1	IN	bool	true：アンテナポート ON；false：アンテナポート OFF
an2	IN	bool	true：アンテナポート ON；false：アンテナポート OFF
an3	IN	bool	true：アンテナポート ON；false：アンテナポート OFF
an4	IN	bool	true：アンテナポート ON；false：アンテナポート OFF
an5	IN	bool	true：アンテナポート ON；false：アンテナポート OFF
an6	IN	bool	true：アンテナポート ON；false：アンテナポート OFF
an7	IN	bool	true：アンテナポート ON；false：アンテナポート OFF
an8	IN	bool	true：アンテナポート ON；false：アンテナポート OFF
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： 指定されるデバイスのアンテナポートをオンにし、タグインベントリーを行います。			
例：アンテナ 1 をオンにして、インベントリーを行います。 StartInventory(true,false,false,false,false,false,false,false);			

2.1.5 StartInventory

関数名	UInt32 StartInventory(InventoryType inventoryType)		
引数名	IN/OUT	型	説明
inventoryType	IN	enum	表示内容をオンにします。章 2.2.1.1 をご参照。
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： デバイスがタグのデータをインベントリーします。			
例：タグの EPC バンドデータのみインベントリーします。 StartInventory(ONLY_PC_EPC)；			

2.1.6 StopInventory

関数名	UInt32 StopInventory()		
引数名	IN/OUT	型	説明
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： デバイスがタグのインベントリーを停止します。			
例：インベントリーを停止します。 StopInventory()；			

2.1.7 SetSelectMask

関数名	UInt32 SetSelectMask(MemBankType memBank,TargetType target,ActionType action uint startAddressWord,byte[] selectMask)		
引数名	IN/OUT	型	説明
memBank	IN	enum	マスク比較を行うタグのメモリバンク 章 2.2.1.6 をご参照
target	IN	enum	Selection Mask を適用するタグセッション 章 2.2.1.4 をご参照
action	IN	enum	タグがマークされた後のアクション 章 2.2.1.5 をご参照
startAddressWord	IN	UInt32	選択したタグのメモリバンクのオフセット（スタートアドレス）単位：word
selectMask	IN	byte	マスクの値を選択[0-16] 単位：word
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： 選択されたマスクのパラメータを設定します。 複数のタグがある場合、この方法により特定のタグに対してインベントリー、読み取り、書き込み、ロックなどの操作をすることができます。			
例： タグをフィルターする条件： タグのメモリバンク： EPC セッション設定： SESSION_S0 アクション設定： ACTION_ASLINVA_DSLINVB オフセット：2 マスクを選択する内容は下記通り： byte[] selectMask= {0x12,0x34,0x56,0x78,0x12,0x34,0x56x,0x78,0x12,0x34,0x56,0x78}; SetSelectMask(Mem_EPC,SESSION_S0,ACTION_ASLINVA_DSLINVB,0x02,selectMask);			

2.1.8 WriteMemory

関数名	UInt32 WriteMemory(MemBankType memBank,uint startAddressWord, uint accessPassword,byte[] writeData,byte[] epcData)		
引数名	IN/OUT	型	説明
memBank	IN	enum	章 2.2.1.6 をご参照
startAddressWord	IN	uint	ターゲットを書き込みバンクのオフセット

			単位：Word
accessPassword	IN	uint	ターゲットタグのアクセスパスワード。(パスワードを設定してない場合、0 である)
writeData	IN	byte	書き込みデータ
epcData	IN	byte	ターゲットタグの EPC 値
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： epcData でタグを選定し、タグのターゲットバンクにデータを書き込みます。 タグのメモリバンクに書き込むデータの長さは 32 ワード（Words）を超えないことです。すなわち 64 バイト（Bytes）です。			
例：タグを書き込みます。 メモリバンク：EPC オフセット：2 アクセスパスワード：0x12345678； 書き込むデータ：byte[] writedata = {0x12,0x34}； ターゲットタグの EPC 値： byte[] epcData= {0x12,0x34,0x56,0x78,0x12,0x34,0x56x,0x78,0x12,0x34,0x56,0x78}； WriteMemory(Mem_EPC,0x02,0x12345678,writeData,epcData)；			

2.1.9 ReadMemory

関数名	UInt32 ReadMemory(MemBankType memBank,uint startAddressWord, uint lengthWord, uint accessPassword,byte[] epcData)		
引数名	IN/OUT	型	説明
memBank	IN	enum	章 2.2.1.6 をご参照
startAddressWord	IN	unit	ターゲット読み取りバンクのオフセット 単位：Word
lengthWord	IN	unit	読み取りの長さ、単位：Word
accessPassword	IN	unit	ターゲットタグのアクセスパスワード。(パスワードがない場合、0 である)
epcData	IN	byte	ターゲットタグの EPC 値
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： epcData でタグを選定し、タグのターゲットバンクのデータを読み取ります。 タグのメモリバンクに読み取ったデータの長さは 32 ワード（Words）を超えないことです。すなわり 64 バイト（Bytes）です。			
例：タグを読み取ります。 ターゲットバンク：EPC			

オフセット：2 長さ：2 アクセスパスワード：0x12345678 ターゲットタグの EPC 値： byte[] epcData= {0x12,0x34,0x56,0x78,0x12,0x34,0x56x,0x78,0x12,0x34,0x56,0x78}; ReadMemory(Mem_EPC,0x02,0x02,0x12345678,epcData);

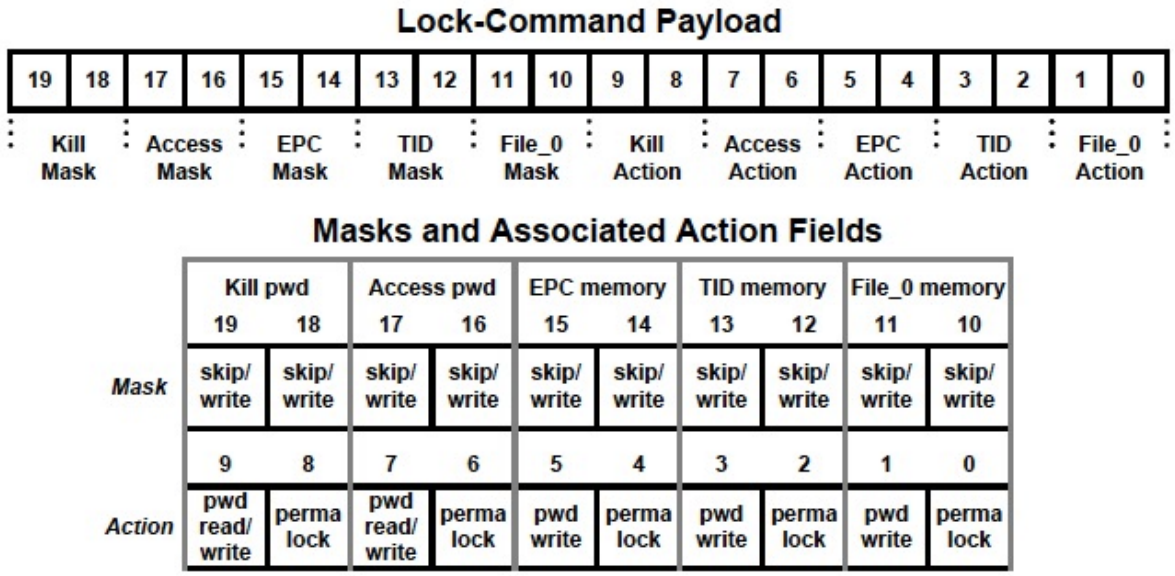
2.1.10 Kill

関数名	UInt32 Kill(uint KillPassword,byte[] epcData)		
引数名	IN/OUT	型	説明
KillPassword	IN	uint	ターゲットタグのキルパスワード
epcData	IN	byte	ターゲットタグの EPC 値
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： epcData でターゲットタグを選定し、タグをキルします。 タグをキルする前にキルパスワードを RESERVED バンクに書き込む必要があります。 オフセット 00 から 2 Word を書き込みます。 注意：タグをキルすると、回復できなくなります。			
例：キルパスワード：0x12345678 ターゲットタグの EPC 値： byte[] epcData= {0x12,0x34,0x56,0x78,0x12,0x34,0x56x,0x78,0x12,0x34,0x56,0x78}; Kill(0x12345678,epcData);			

2.1.11 LockMemory

関数名	UInt32 LockMemory(TagMask tagMask,TagAction tagAction, uint accessPassword,byte[] epcData)		
引数名	IN/OUT	型	説明
tagMask	IN	TagMask	ロック操作のマスク（付録Ⅲをご参照）
tagAction	IN	TagMask	ロック操作のデータ（付録Ⅲをご参照）
accessPassword	IN	uint	ターゲットタグのアクセスパスワード（パスワードを設定してない場合、0 である。）
epcData	IN	byte	ターゲットタグの EPC 値
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明：			

タグのメモリバンクに対してロック（Lock）、永久ロック（PermaLock）、アンロック（Unlock）永久アンロック（PermaUnlock）をします。
タグをロックする前に、アクセスパスワードを RESERVED バンクに書き込む必要があります。
オフセット 0X20 から 2 Word を書き込みます。
ロック操作パラメータ target の高 4 桁は保留桁であり、残りの 20 桁はロック操作の Payload であり、Mask と Action を含めて、高いから低い順に 10 桁ずつあります。
マスクは Mask 桁が 1 の Action だけに有効です。各データエリアの Action は 2 bits、00~11 があり、オープン、永久オープン、ロック、永久ロックの順番です。
Mask と Action の各桁の意味は以下通り：



例：タグの EPC バンクをロックします。
ターゲットタグのアクセスパスワード：12345678。
ターゲットタグの EPC 値：

```
byte[] epcData= {0x12,0x34,0x56,0x78,0x12,0x34,0x56x,0x78,0x12,0x34,0x56,0x78};
LockMemory(0x008020,0x12345678,epcData);
```

2.1.12 SetRegion

関数名	UInt32 SetRegion(RegionType region)		
引数名	IN/OUT	型	説明
region	IN	enum	当面のリージョン（章 2.2.1.1 をご参照）
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： リージョンを設定します。			

例：リージョンを REGION_CHINA1 に設定します。 SetRegion(REGION_CHINA1)；

2.1.13 GetRegion

関数名	UInt32 GetRegion(ref Region region)		
引数名	IN/OUT	型	説明
region	OUT	enum	当面のリージョン（章 2.2.1.2 をご参照）
戻り値	OUT	UInt32	実行成功：０を返す 実行失敗：１を返す
メソッド説明： リージョンを取得します。			
例：リージョンを取得します。 GetRegion(Value)；			

2.1.14 GetTxPower

関数名	UInt32 GetTxPower(ref uint power)		
引数名	IN/OUT	型	説明
power	OUT	uint	発射パワーのパラメータ値［13-24］
戻り値	OUT	UInt32	実行成功：０を返す 実行失敗：１を返す
メソッド説明： リーダライターの発射パワーを取得します。			
例：リーダライターの発射パワーを取得します。 GetTxPower(power)；			

2.1.15 SetTxPower

関数名	UInt32 SetTxPower(uint txPower)		
引数名	IN/OUT	型	説明
txPower	IN	uint	発射パワーのパラメータ値［13-24］
戻り値	OUT	UInt32	実行成功：０を返す 実行失敗：１を返す
メソッド説明： リーダライターの発射パワーを設定します。			

例：発射パワーを 24dBm に設定します。 SetTxPower(24)；

2.1.16 SetTxPower

関数名	UInt32 SetTxPower(uint txPower_an1, uint txPower_an2, uint txPower_an3, uint txPower_an4, uint txPower_an5, uint txPower_an6, uint txPower_an7, uint txPower_an8)		
引数名	IN/OUT	型	説明
txPower_an1	IN	uint	アンテナ１のパワーを設定［13-24］
txPower_an2	IN	uint	アンテナ２のパワーを設定［13-24］
txPower_an3	IN	uint	アンテナ３のパワーを設定［13-24］
txPower_an4	IN	uint	アンテナ４のパワーを設定［13-24］
txPower_an5	IN	uint	アンテナ５のパワーを設定［13-24］
txPower_an6	IN	uint	アンテナ６のパワーを設定［13-24］
txPower_an7	IN	uint	アンテナ７のパワーを設定［13-24］
txPower_an8	IN	uint	アンテナ８のパワーを設定［13-24］
戻り値	OUT	UInt32	実行成功：０を返す 実行失敗：１を返す
メソッド説明： リーダライターのアンテナパワーを設定します。			
例：リーダライターアンテナ１のパワーを 24dBm に設定します。 SetTxPower(24,0,0,0,0,0,0,0)；			

2.1.17 GetTxPower

関数名	UInt32 GetTxPower(ref uint power_an1, ref uint power_an2, ref uint power_an3, ref uint power_an4, ref uint power_an5, ref uint power_an6, ref uint power_an7, ref uint power_an8)		
引数名	IN/OUT	型	説明
Power_an1	OUT	uint	アンテナ１のパワーを取得［13-24］
Power_an2	OUT	uint	アンテナ２のパワーを取得［13-24］
Power_an3	OUT	uint	アンテナ３のパワーを取得［13-24］
Power_an4	OUT	uint	アンテナ４のパワーを取得［13-24］
Power_an5	OUT	uint	アンテナ５のパワーを取得［13-24］
Power_an6	OUT	uint	アンテナ６のパワーを取得［13-24］
Power_an7	OUT	uint	アンテナ７のパワーを取得［13-24］
Power_an8	OUT	uint	アンテナ８のパワーを取得［13-24］

戻り値	OUT	UInt32	実行成功：0を返す 実行失敗：1を返す
メソッド説明： リーダライターのアンテナパワーを取得します。			
例：リーダライターのアンテナパワーを取得します。 GetTxPower(power_an1, power_an2, power_an3, power_an4, power_an5, power_an6, power_an7, power_an8)；			

2.1.18 SetSession

関数名	UInt32 SetSession(SessionType session)		
引数名	IN/OUT	型	説明
session	IN	enum	RFID Session（章2.2.1.3をご参照）
戻り値	OUT	UInt32	実行成功：0を返す 実行失敗：1を返す
メソッド説明： RFID Sessionを設定します。			
例： RFID Sessionを Session_S0に設定します。 SetSession(Session_S0)；			

2.1.19 GetSession

関数名	UInt32 GetSession(ref SessionType session)		
引数名	IN/OUT	型	説明
session	OUT	enum	RFID Session（章2.2.1.3をご参照）
戻り値	OUT	UInt32	実行成功：0を返す 実行失敗：1を返す
メソッド説明： RFID Sessionを取得します。			
例：RFID Sessionを取得します。 GetSession(session)；			

2.1.20 SetChannel

関数名	UInt32 SetChannel(ChannelType channel)		
引数名	IN/OUT	型	説明
channel	IN	enum	RFIDのチャンネル（章2.2.1.7をご参照）

			チャンネル番号の範囲はリージョンによって異なります。
戻り値	OUT	UInt32	実行成功：0を返す 実行失敗：1を返す
メソッド説明： リーダライターのRFIDチャンネルを設定します。			
例：RFIDのチャンネルをCHANNEL_24に設定します。 SetChannel(CHANNEL_24)；			

2.1.21 GetChannel

関数名	UInt32 GetChannel(ref ChannelType channel)		
引数名	IN/OUT	型	説明
channel	OUT	enum	RFIDのチャンネル（章2.2.1.7をご参照） チャンネル番号の範囲はリージョンによって異なります。
戻り値	OUT	UInt32	実行成功：0を返す 実行失敗：1を返す
メソッド説明： RFIDのチャンネルを取得します。			
例：RFIDのチャンネルを取得します。 GetChannel(channel)；			

2.1.22 SetGain

関数名	UInt32 SetGain(GainType gain)		
引数名	IN/OUT	型	説明
gain	IN	enum	PAゲインモード（章2.2.1.8をご参照）
戻り値	OUT	UInt32	実行成功：0を返す 実行失敗：1を返す
メソッド説明： PAゲインモードを設定します。			
例： PAゲインモードをHIGH_GAINに設定します。 SetGain(HIGH_GAIN)；			

2.1.23 GetGain

関数名	UInt32 GetGain(ref GainType gain)		
引数名	IN/OUT	型	説明
gain	OUT	enum	PA ゲインモード（章 2.2.1.8 をご参照）
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： PA ゲインモードを取得します。			
例： PA ゲインモードを取得します。 GetGain(gain)；			

2.1.24 SetQuery

関数名	UInt32 SetQuery(QueryType dr, QueryType m,QueryType trext, QueryType sel, QueryType session, QueryType target, QType q)		
引数名	IN/OUT	型	説明
dr	IN	enum	TRcal 分周比が周波数を設定します。（章 2.2.1.10 をご参照）
m	IN	enum	エンコードタイプ。（章 2.2.1.11 をご参照）
trext	IN	enum	プリアンブルの前にパイロットトーンを付加するかどうかを選択します。（章 2.2.1.12 をご参照）
sel	IN	enum	クエリに応答するタグを選択します。（章 2.2.1.13 をご参照）
session	IN	enum	セッションを選択します。（章 2.2.1.3 をご参照）
target	IN	enum	インベントリーフラグ A またはフラグ B を選択します。（章 2.2.1.9 をご参照）
q	IN	enum	ラウンドのスロット数を設定します。スロット数＝2 の Q 乗（章 2.2.1.14 をご参照）
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： Query パラメータを設定します。			

例：分周比を DR_8 に設定、エンコードモード＝M1、パイロットトーンを付加しない、Se＝SEL_ALL、セッション＝SESSION_S0、セッションフラッグ＝TARGET_A、スロット：スロット＝16。 SetQuery(DR_8,M1,NO_Pilot_Tone,SEL_ALL,SESSION_S0,TARGET_A,Q4)；

2.1.25 GetQuery

関数名	UInt32 GetQuery(ref QueryType dr, ref QueryType m, ref QueryType trext, ref QueryType sel, ref QueryType session, ref QueryType target, ref QType q)		
引数名	IN/OUT	型	説明
dr	OUT	enum	TRcal 分周比が周波数を設定します。（章 2.2.1.10 をご参照）
m	OUT	enum	エンコードタイプ。（章 2.2.1.11 をご参照）
trext	OUT	enum	プリアンブルの前にパイロットトーンを付加するかどうかを選択します。（章 2.2.1.12 をご参照）
sel	OUT	enum	クエリに応答するタグを選択します。（章 2.2.1.13 をご参照）
session	OUT	enum	セッションを選択します。（章 2.2.1.3 をご参照）
target	OUT	enum	インベントリーフラグ A またはフラグ B を選択します。（章 2.2.1.9 をご参照）
q	OUT	enum	ラウンドのスロット数を設定します。スロット数＝2 の Q 乗（章 2.2.1.14 をご参照）
戻り値	OUT	UInt32	実行成功：0 を返す 実行失敗：1 を返す
メソッド説明： タグクエリーのすべてのパラメータ値を取得します。			
例：タグクエリーのすべてのパラメータ値を取得します。 GetQuery(dr,m,trext,sel,session,target,q)；			

2.1.26 SetAntiCollisionMode

関数名	UInt32 SetAntiCollisionMode(AntiCollisionMode antiCollisionMode)		
引数名	IN/OUT	型	説明
antiCollisionMode	IN	enum	アンチコリジョンモード（章 2.2.1.15 をご参照）

戻り値	OUT	UInt32	実行成功：0を返す 実行失敗：1を返す
メソッド説明： アンチコリジョンモードを設定します。			
例：アンチコリジョンモードをFixedQに設定します。 SetAntiCollisionMode(FixedQ)；			

2.1.27 GetAntiCollisionMode

関数名	UInt32 GetAntiCollisionMode(ref AntiCollisionMode antiCollisionMode)		
引数名	IN/OUT	型	説明
antiCollisionMode	OUT	enum	アンチコリジョンモード （章2.2.1.15をご参照）
戻り値	OUT	UInt32	実行成功：0を返す 実行失敗：1を返す
メソッド説明： デバイスが使用中のアンチコリジョンモードを取得します。			
例：デバイス使用中のアンチコリジョンモードを取得します。 GetAntiCollisionMode(antiCollisionMode)；			

2.1.28 SetCWOn

関数名	UInt32 SetCWOn()		
引数名	IN/OUT	型	説明
戻り値	OUT	UInt32	実行成功：0を返す 実行失敗：1を返す
メソッド説明： CW搬送波をオンにします。 CW搬送波：順方向リンク変調がない場合、コンフィグされたチャンネルまたは周波数でレーダーの信号を発送します。			
例：CW搬送波をオンにします。 SetCWOn()；			

2.1.29 SetCWOff

関数名	UInt32 SetCWOff()		
引数名	IN/OUT	型	説明
戻り値	OUT	UInt32	実行成功：０を返す 実行失敗：１を返す
メソッド説明： CW 搬送波をオフにします。 CW 搬送波：順方向リンク変調がない場合、コンフィグされたチャンネルまたは周波数でレーダーの信号を発送します。			
例：CW 搬送波をオフにします。 SetCWOff();			

2.1.30 SetReadTime

関数名	UInt32 SetReadTime(uint time_an1,uint time_an2,uint time_an3,uint time_an4,uint time_an5,uint time_an6,uint time_an7,uint time_an8)		
引数名	IN/OUT	型	説明
time_an1	IN	uint	アンテナ１の読取タイム。(10~40000,1=1ms)
time_an2	IN	uint	アンテナ２の読取タイム。(10~40000,1=1ms)
time_an3	IN	uint	アンテナ３の読取タイム。(10~40000,1=1ms)
time_an4	IN	uint	アンテナ４の読取タイム。(10~40000,1=1ms)
time_an5	IN	uint	アンテナ５の読取タイム。(10~40000,1=1ms)
time_an6	IN	uint	アンテナ６の読取タイム。(10~40000,1=1ms)
time_an7	IN	uint	アンテナ７の読取タイム。(10~40000,1=1ms)
time_an8	IN	uint	アンテナ８の読取タイム。(10~40000,1=1ms)
戻り値	OUT	UInt32	実行成功：０を返す 実行失敗：１を返す
メソッド説明：各アンテナの読取タイムを設定します。			
例；各アンテナの読取タイムを 1000ms に設定します。 SetReadTime(1000,1000,1000,1000,1000,1000,1000,1000)；			

2.1.31 GetReadTime

関数名	UInt32 GetReadTime(ref uint time_an1,ref uint time_an2,ref uint time_an3,ref uint time_an4,ref uint time_an5,ref uint time_an6,ref uint time_an7,ref uint time_an8)		
引数名	IN/OUT	型	説明
time_an1	OUT	uint	アンテナ１の読取タイム。(10~40000, 1=1ms)
time_an2	OUT	uint	アンテナ２の読取タイム。(10~40000, 1=1ms)
time_an3	OUT	uint	アンテナ３の読取タイム。(10~40000, 1=1ms)
time_an4	OUT	uint	アンテナ４の読取タイム。(10~40000, 1=1ms)
time_an5	OUT	uint	アンテナ５の読取タイム。(10~40000, 1=1ms)
time_an6	OUT	uint	アンテナ６の読取タイム。(10~40000, 1=1ms)
time_an7	OUT	uint	アンテナ７の読取タイム。(10~40000, 1=1ms)
time_an8	OUT	uint	アンテナ８の読取タイム。(10~40000, 1=1ms)
戻り値	OUT	UInt32	実行成功：０を返す 実行失敗：１を返す
メソッド説明：各アンテナの読取タイムを取得します。			
例：各アンテナの読取タイムを取得します。 GetReadTime(time_an1,time_an2,time_an3,time_an4,time_an5,time_an6,time_an7,time_an8); 			

2.1.32 GetDeviceInformation

関数名	void GetDeviceInformation(ref DeviceInformation deviceInformation)		
引数名	IN/OUT	型	説明
deviceInformation	OUT	DeviceInformation	デバイス情報を取得 付録Ⅱをご参照
戻り値	OUT	Void	
メソッド説明： デバイス情報を取得します。			
例：デバイス情報を取得します。 DeviceInformation mode; GetDeviceInformation(mode);			

2.1.33 SetStaticIP

関数名	bool SetStaticIP(string ip,string subnet,string gateway,string dnsServer)		
引数名	IN/OUT	型	説明
ip	IN	string	IP アドレス
subnet	IN	string	サブネットマスク
gateway	IN	string	ゲートウェイ
dnsServer	IN	string	DNS サーバー
戻り値	OUT	bool	
メソッド説明： 静的 IP アドレスを設定します。			
例： SetStaticIP(“192.168.3.1”,” 255.255.255.0”,” 192.168.3.1”,” 192.168.3.1”)；			

2.1.34 GetSdkVersion

関数名	void GetSdkVersion(ref string sdkVersion)		
引数名	IN/OUT	型	説明
sdkVersion	OUT	string	SDK バージョンを取得
戻り値	OUT	Void	
メソッド説明： SDK バージョンを取得します。			
例： string sdkVersion; GetSdkVersion(sdkVersion)；			

2.1.35 SetDelegate

関数名	Void SetDelegate(CallBackReadTagDatareadTagData,CallBackErrorCode errorCode,CallBackSuccessCode successCode)		
引数名	IN/OUT	型	説明
readTagData	IN	CallBackReadTagData	データ処理のコールバック関数
errorCode	IN	CallBackErrorCode	実行にエラーが出る場合のコールバック関数
successCode	IN	CallBackSuccessCode	成功に実行した場合のコールバック関数
戻り値	OUT	Void	

メソッド説明： デリゲート関数を設定します。
例： CallBackReadTagData Rec = null; CallBackErrorCode Rec1 = null; CallBackSuccessCode Rec2 = null; Void test(InventoryResult ReadTagStruct); Void test1(uint error); Void test2(uint success); Rec = test; Rec1 = test1; Rec2 = test2 SetDelegate(Rec,Rec1,Rec2);
注意：データを処理するデリゲートと実行にエラーが出る場合の出力用デリゲートを定義します。 public delegate void CallBackReadTagData(InventoryResult tagcallbackdata); public delegate void CallBackErrorCode (uint error); public delegate void CallBackErrorCode (uint success); InventoryResult 付録 I をご参照 error 章 2.2.1.16 をご参照。 success 章 2.2.1.17 をご参照。

2.2 Types Class

Types クラスはリージョン、RFID モード、セッションモード、サーチモード、Selection Mask のタグセッション、タグのセッション状態、メモリバンク、パラメータ設定を定義します。

2.2.1 列挙型

2.2.1.1 InventoryType

インベントリー引数	パラメータ
PC_EPC_RSSI	1
PC_EPC_TID	2
ONLY_PC_EPC	3

2.2.1.2 RegionType

リージョン	パラメータ
REGION_US	0x21
REGION_US_Narrow	0x22
REGION_Europe	0x31
REGION_JAPAN	0x41
REGION_CHINA1	0x51
REGION_CHINA2	0x52
REGION_BRAZIL	0x61

2.2.1.3 SessionType

セッションモード

セッションモード	パラメータ
Session_S0	0x0
Session_S1	0x1
Session_S2	0x2
Session_S3	0x3

2.2.1.4 TargetType

Selection Mask を適用するタグのセッションを設定します。

Target	パラメータ
SESSION_S0	0x0
SESSION_S1	0x1
SESSION_S2	0x2
SESSION_S3	0x3
SL_FLAG	0x4

2.2.1.5 ActionType

タグのセッション状態を設定します。

Selection Mask 使う時に、Selection Mask にマッチング、マッチングしない場合、タグの Session と Session Flag の変化をそれぞれに表示します。

Action	パラメータ
ACTION_ASLINVA_DSLINVB	0x0
ACTION_ASLINVA_NOTHING	0x1
ACTION_NOTHING_DSLINVB	0x2
ACTION_NSLINVS_NOTHING	0x3
ACTION_DSLINVB_ASLINVA	0x4
ACTION_DSLINVB_NOTHING	0x5
ACTION_NOTHING_ASLINVA	0x6
ACTION_NOTHING_NSLINVS	0x7

2.2.1.6 MemBankType

Selection Mask 比較を行うタグのメモリバンクを設定します。

MemBank	パラメータ
MEM_RESERVED	0x0
MEM_EPC	0x1
MEM_TID	0x2
MEM_USER	0x3

2.2.1.7 ChannelType

RF チャンネル	パラメータ
CHANNEL_24	24
CHANNEL_25	25
CHANNEL_26	26
CHANNEL_27	27
CHANNEL_28	28
CHANNEL_29	29
CHANNEL_30	30
CHANNEL_31	31
CHANNEL_32	32

2.2.1.8 GainType

PA ゲインモード	パラメータ
HIGH_GAIN	0x00
LOW_GAIN	0x01

2.2.1.9 TargetABType

セッションフラッグ	パラメータ
TARGET_A	0x00
TARGET_B	0x01

2.2.1.10 DRType

分周比	パラメータ
DR_8	0x00
DR_64_3	0x01

2.2.1.11 MType

エンコードタイプ	パラメータ
M1	0x00
M2	0x01
M4	0x02
M8	0x03

2.2.1.12 TRextType

パイロットトーン付加するかどうか	パラメータ
NO_Pilot_Tone	0x00
Use_Pilot_Tone	0x01

2.2.1.13 SelType

クエリに応答するタグを選択	パラメータ
SEL_ALL	0x00
SEL_SL_N	0x02

SEL_SL	0x03

2.2.1.14 QType

Q 値	パラメータ
Q0	0
Q1	1
Q2	2
Q3	3
Q4	4
Q5	5
Q6	6
Q7	7
Q8	8
Q9	9
Q10	10
Q11	11
Q12	12
Q13	13
Q14	14
Q15	15

2.2.1.15 AntiCollisionMode

アンチコリジョン	パラメータ
FixedQ	0x0
DynamicQ	0x01

2.2.1.16 ErrorCode

エラーコード	パラメータ
OTHER_ERROR	0x0
NOT_SUPPORTED	0x1
INSUFFICIENT_PRIVILEGES	0x2
MEMORY_OVERRUN	0x3
MEMORY_LOCKED	0x4
CRYPTO_SUITE_ERROR	0x5
COMMAND_NOT_ENCAPSULATED	0x6

RESPONSEBUFFER_OVERFLOW	0x7
SECURITY_TIMEOUT	0x8
INSUFFICIENT_POWER	0xB
NON_SPECIFIC_ERROR	0xF
SENSOR_SCHEDULING_CONFIGURATION	0x11
TAG_BUSY	0x12
MEASUREMENT_TYPE_NOT_SUPPORTED	0x13
NO_TAG_DETECTED	0x80
HANDLE_ACQUISITION_FAILURE	0x81
ACCESS_PASSWORD_FAILURE	0x82
KILL_PASSWORD_FAILURE	0x83
CRC_ERROR	0x90
RX_TIMEOUT	0x91
REGISTRY_UPDATE_FAILURE	0xA0
REGISTRY_ERASE_FAILURE	0xA1
REGISTRY_WRITE_FAILURE	0xA2
REGISTRY_NOT_EXIST	0xA3
UART_FAILURE	0xB0
SPI_FAILURE	0xB1
I2C_FAILURE	0xB2
GPIO_FAILURE	0xB3
NOT_SUPPORTED_COMMAND	0xE0
UNDEFINED_COMMAND	0xE1
INVALID_PARAMETER	0xE2
TOO_HIGH_PARAMETER	0xE3
TOO_LOW_PARAMETER	0xE4
FAILURE_AUTOMATIC_READ_OPERATION	0xE5
NOT_AUTOMATIC_READ_MODE	0xE6
FAILURE_TO_GET_LAST_RESPONSE	0xE7
FAILURE_TO_CONTROL_TEST	0xE8
FAILURE_TO_RESET_READER	0xE9
RFID_BLOCK_CONTROL_FAILURE	0xEA
PR9200_BUSY	0xEB
COMMAND_FAILURE	0xF0
VERIFY_FAILURE	0xF1
ABNORMAL	0xFC
ERROR_NONE	0xFF

2.2.1.17 SuccessCode

成功コード	パラメータ
SET_READER_POWER_CONTROL	0x0
GET_READER_INFORMATION	0x03
GET_REGION	0x06
SET_REGION	0x07
SET_SYSTEM_RESET	0x08
GET_TYPE_C_AI_SELECT_PARAMETERS	0xB
SET_TYPE_C_AI_SELECT_PARAMETERS	0xC
GET_TYPE_C_AI_QUERY_RELATED_PARAMETERS	0xD
SET_TYPE_C_AI_QUERY_RELATED_PARAMETERS	0xE
GET_CURRENT_RF_CHANNEL	0x11
SET_CURRENT_RF_CHANNEL	0x12
GET_FH_AND_LBT_PARAMETERS	0x13
SET_FH_AND_LBT_PARAMETERS	0x14
GET_TX_POWER_LEVEL	0x15
SET_TX_POWER_LEVEL	0x16
RF_CW_SIGNAL_CONTROL	0x17
GET_MULTIPLE_POWER	0x18
SET_MULTIPLE_POWER	0x19
SET_ANTENNA	0x1B
GET_READ_TIME	0x1E
SET_READ_TIME	0x1F
READ_TYPE_C_UII	0x22
READ_TYPE_C_UII_RSSI	0x23
READ_TYPE_C_USER_DATA	0x24
READ_TYPE_C_UII_TID	0x25
START_AUTO_READ	0x27
STOP_AUTO_READ	0x28
READ_TYPE_C_TAG_DATA	0x29
READ_TYPE_C_TAG_DATA2	0x2A
GET_SESSION	0x2E
SET_SESSION	0x2F
GET_FREQUENCY_HOPPING_TABLE	0x30
SET_FREQUENCY_HOPPING_TABLE	0x31
GET_MODULATION	0x32
SET_MODULATION	0x33
GET_ANTICOLLISION_MODE	0x34
SET_ANTICOLLISION_MODE	0x35
START_AUTO_READ2	0x36

STOP_AUTO_READ2	0x37
START_AUTO_READ_RSSI	0x38
STOP_AUTO_READ_RSSI	0x39
START_AUTO_READ_EX2	0x3A
WRITE_TYPE_C_TAG_DATA	0x46
BLOCKWRITE_TYPE_C_TAG_DATA	0x47
BLOCKERASE_TYPE_C_TAG_DATA	0x48
NXP_READPROTECT	0x52
NXP_RESET_READPROTECT	0x53
NXP_CHANGE_EAS	0x54
NXP_EAS_ALARM	0x55
NXP_CALIBRATE	0x56
ISP_DATA	0x57
KILL_RECOM_TYPE_C_TAG	0x65
SET_GAIN	0x66
Get_GAIN	0x67
LOCK_TYPE_C_TAG	0x82
BLOCKPERMALOCK_TYPE_C_TAG	0x83
SET_MODEM_REGISTER	0xA6
SET_RF_REGISTER	0xA7
GET_MODEM_REGISTER	0xA8
GET_RF_REGISTER	0xA9
ISP_DOWNLOAD	0xB1
ISP_DUMP	0xB3
SET_ANTENNA_PATH	0xB4
SET_LEAKAGE_CAL_MODE	0xB5
GET_IAP_VERSION	0xB6
GET_TEMPERATURE	0xB7
GET_RSSI	0xC5
SCAN_RSSI	0xC6
UPDATE_REGISTRY	0xD2
ERASE_REGISTRY	0xD3
GET_REGISTRY_ITEM	0xD4
SET_REGISTRY_ITEM	0xD5
SET_OPTIMUM_FREQUENCY_HOPPING_TABLE	0xE4
GET_FREQUENCY_HOPPING_MODE	0xE5
SET_FREQUENCY_HOPPING_MODE	0xE6
GET_TX_LEAKAGE_RSSI_LEVEL_FOR_SMART_HOPPING_MODE	0xE7
SET_TX_LEAKAGE_RSSI_LEVEL_FOR_SMART_HOPPING_MODE	0xE8

START_READ_WITH_FAST_LEAKAGE_CAL	0xEC
REQUEST_FAST_LEAKAGE_CAL	0xED

付録Ⅰ：

1. InventoryResult

タグをインベントリーする時に表示される項目
rssi
channel
phase
antenna
TagData tagData

2. TagData

タグのデータ
pc
epc
tid
data

付録Ⅱ：

デバイス情報
macAddress
version
status
ipAddress
subnetMask
gateway
dns
dhcp
password

付録Ⅲ：

TagMask
userMemoryBit1
userMemoryBit2
tidMemoryBit1
tidMemoryBit2
epcMemoryBit1
epcMemoryBit2
accessMemoryBit1
accessMemoryBit2
killMemoryBit1
killMemoryBit2

TagAction
userMemoryBit1
userMemoryBit2
tidMemoryBit1
tidMemoryBit2
epcMemoryBit1
epcMemoryBit2
accessMemoryBit1
accessMemoryBit2
killMemoryBit1
killMemoryBit2