

API を使った Windows アプリ開発方法 (CB ファミリ版)

SDK 内にある API(dll)を使って下の条件の Windows アプリを開発する方法を記します。

◆条件

- ◇開発ツール:Visual Studio 2015
- ◇開発言語:C#(.NET)
- ◇アプリケーション:Windows フォームアプリケーション
- ◇対象 RFID リーダ・ライタ:UP4-250-J2(CB ファミリ)、PC とは USB ケーブルで接続

このドキュメントでは、アンテナ4ch の内 1ch と 2ch を使って実際にタグ ID を取得するアプリを作成しながら開発手順を記しています。

■注意

このドキュメント内のソースコードでは、分かりやすさを優先しているためエラー処理を行っていません。

API 関数からの返り値や例外処理など、必要に応じて対処してください。

1. 作成するアプリ

1.1. 概要

下図はこれから作成するアプリでタグ ID を取得している例です。

現在の送信出力値を表示

送信出力(dBm) 設定

アンテナ1 20 アンテナ2 20

送信出力値を設定するボタン

タグの ID を取得するボタン

タグID取得

アンテナ1

tag count = 18

300ED89F335001551DE4F300
300ED89F33500086843B138E
300ED89F335001551DE4F29B
300ED89F335001551DE4F239
300ED89F335001551DE4F1A4
300ED89F33500086843B127C
300ED89F335001551DE4F1FC
300ED89F33500086843B1277
300ED89F335001551DE4F35A
300ED89F33500086843B128B
300ED89F335001551DE4F320
300ED89F33500086843B13CB
300ED89F33500086843B12F4
300ED89F33500086843B139B
300ED89F33500086843B1350
300ED89F335001551DE4F231
300ED89F335001551DE4F30E
300ED89F335001551DE4F212

アンテナ2

tag count = 39

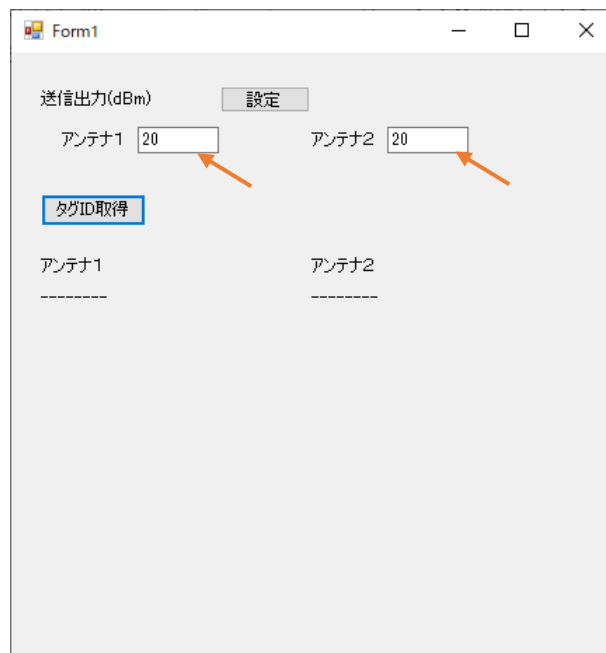
00B07A14541F97100800B707
300ED89F335001551DE4F212
00E08648044477D04801337C
30B48690AC59427E848000C7
00E086480C0D135008003624
00E086481032B5D514001596
00E0864814019CCF00002074
00B07A14B01CFC8F00004C23
00B07A13DC58258800005A45
00E086480453C94808019B83
00B07A14A4021910300033CF
00B07A147C60639048004830
00B07A1458087CD130011EC5
00E086480C2091D048007AE7
300ED89F335001551DE4F1A4
00B07A148838AFCF00000091
00E08648102BB44F0000916C
00E08648082C19C808002358
300ED89F335001551DE4F300
00B07A14580879D13000E9EA
00B07A144037D611300229F7
00E08648102E0E5008002419
00B07A14A0600B9140015BEA
300ED89F335001551DE4F35A
00B07A14A0600B11400138C4
00B07A14583176D12000079A
00B07A14A828D99130000960
00E0864810249D2A691A300C
00112233445566778899AABB
00B07A14581441512000280A
00B07A13DC5825880000090E
00B07A145804FED70C9D044B
00E08648082E815048002B99
00B07A14A41F52914000AABA
300ED89F335001551DE4F30E
00E086480C19998F00000006
0000AAAABBBBCCCCDDDEEEEE
00B07A14B40A2E914000D1CF
300ED89F335001551DE4F29B

取得したタグ数と ID を表示

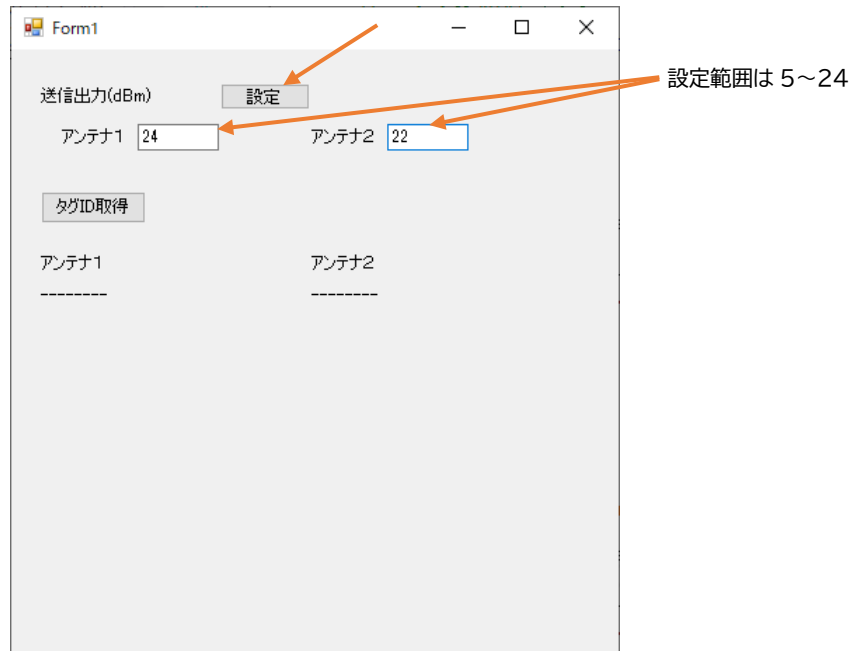
1.2. 操作方法

このアプリの操作方法は下のとおりです。

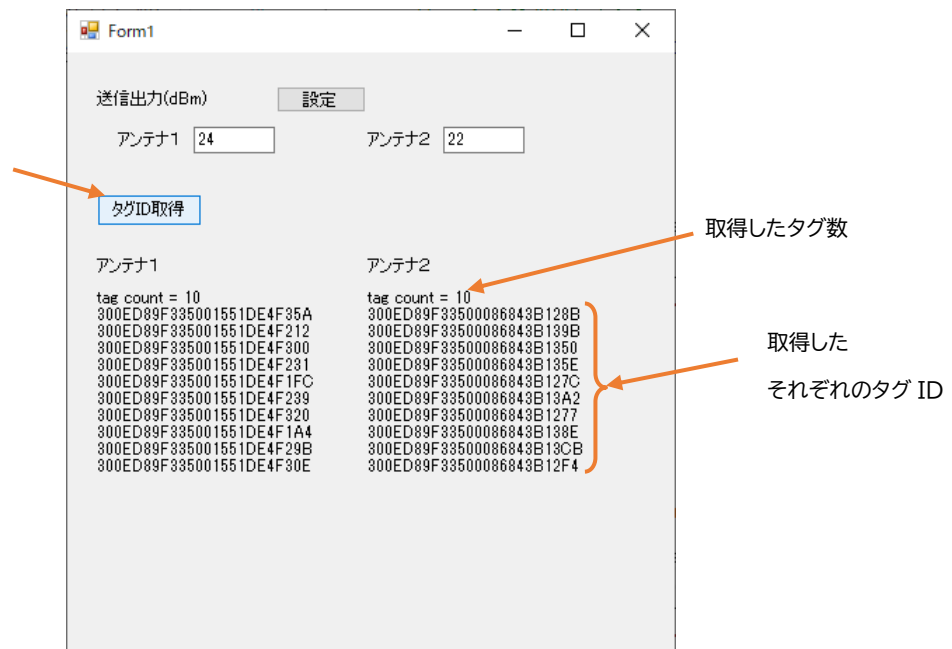
- (1) PC と RFID リーダ・ライタを USB ケーブルで接続します。
また RFID リーダ・ライタのアンテナ1とアンテナ2にアンテナを接続します。
- (2) アプリを起動すると、下の2つを実行した後に下図のダイアログが表示されます。
 - ・RFID リーダ・ライタと接続
 - ・現在のアンテナ1とアンテナ2の各送信出力値を取得して表示



- (3) 送信出力値を変更する場合は、テキストボックスに 5~24 の値を入力してから [設定] ボタンをクリックします。



- (4) アンテナ1とアンテナ2それぞれにタグをかざして[タグ ID 取得] ボタンをクリックすると、アンテナ1でタグを取得して表示し、次にアンテナ2でタグを取得して表示します。



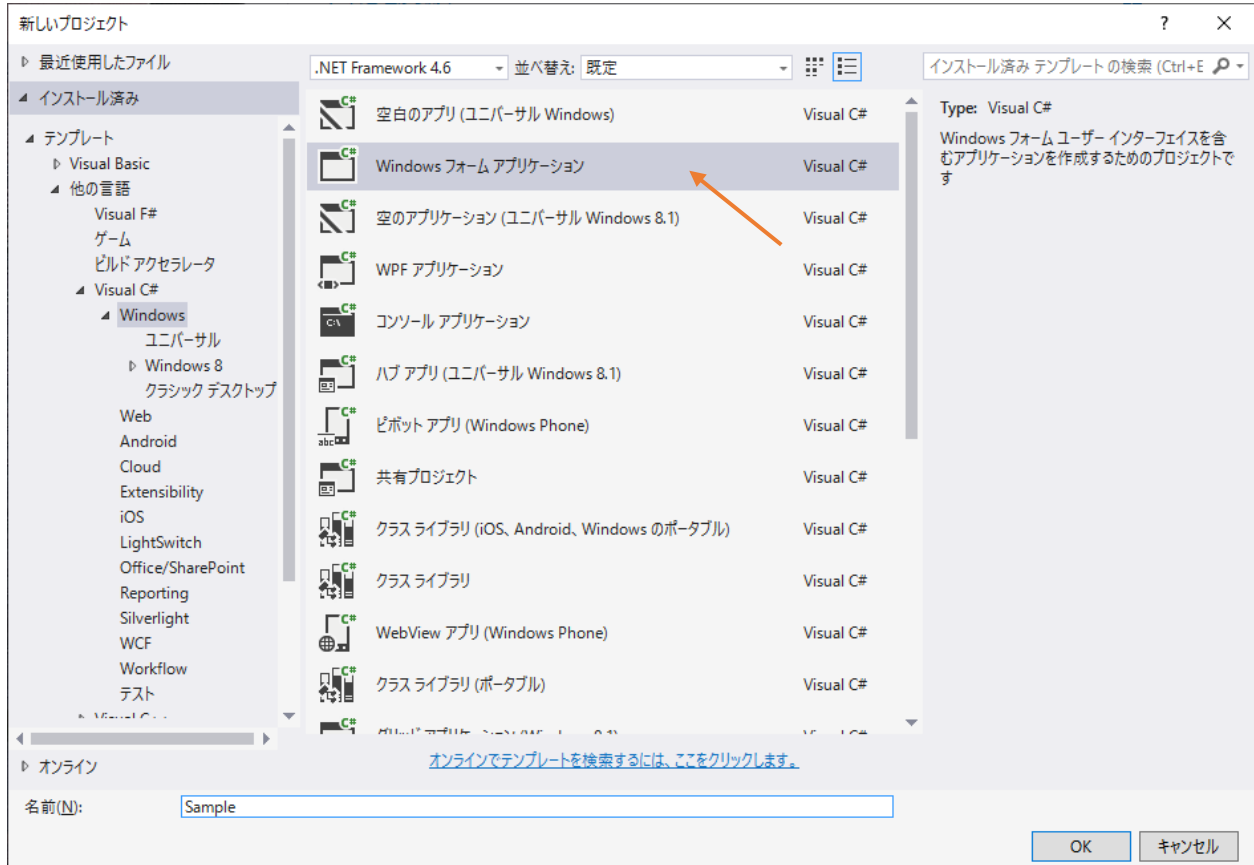
- (5) 画面右上の[×]ボタンでアプリを終了します。
このとき RFID リーダ・ライタと切断します。

それでは、この後実際にこのアプリを作成していきます。

2. アプリの開発

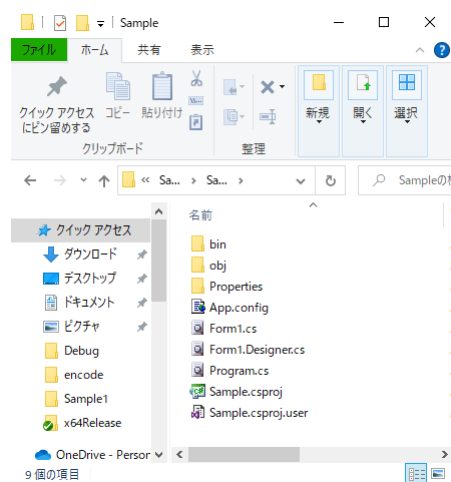
2.1. 新規プロジェクトを作成

Visual Studio を起動して「新しいプロジェクト」を作成します。



[プロジェクト] - [プロパティ] での設定を適宜行います。

一旦[すべて保存]します。保存先をエクスプローラで見ると下のようになっています。



2.2. API をプロジェクトに追加

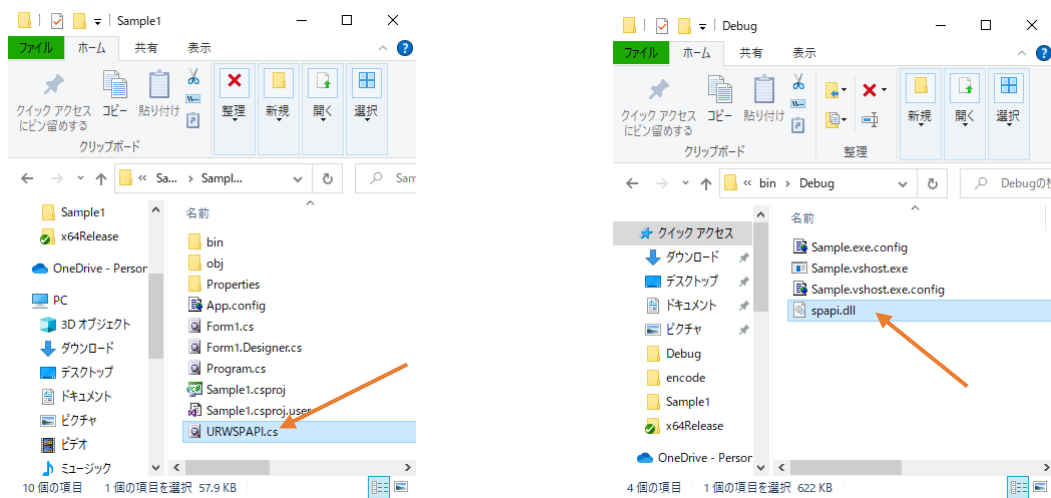
C#での開発で使用する API 関連のファイルは下の2つです。(SDKの中にあります)

- ・URWSPAPI.cs:関数の定義などが記述されているファイルです。
- ・spapi.dll:dll です。

URWSPAPI.cs を左下図のようにアプリケーションのフォルダ内にコピーします。

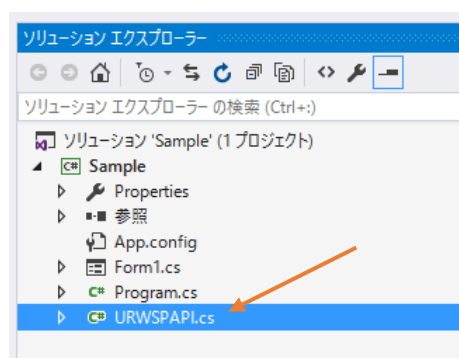
また spapi.dll を右下図のようにアプリケーションの実行ファイル(***.exe)が生成されるフォルダにコピーします。

※spapi.dll は 32bit 用と 64bit 用の2種類あります。生成されるアプリケーションの bit 数により使い分けてください。



URWSPAPI.cs をプロジェクトに追加するために、[プロジェクト] – [既存の項目の追加]でコピーした URWSPAPI.cs を選択します。

下図のように追加されていることを確認してください。



2.3. API を使う準備

Form1.cs を開き API 関数を使うために下図のように

`using spapi;`

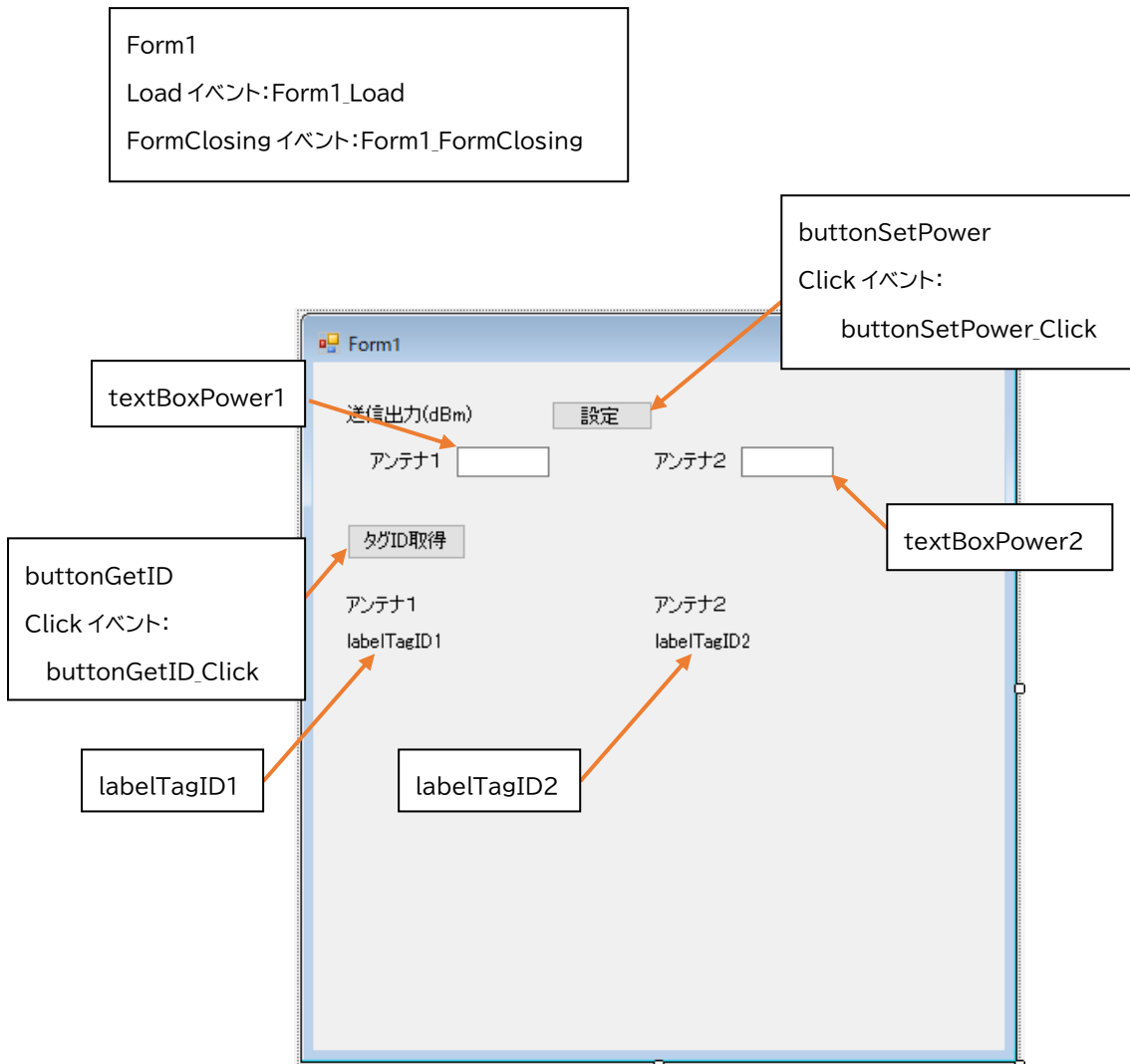
を追記します。

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using spapi; ←

namespace Sample
{
    3 個の参照
    public partial class Form1 : Form
    {
        1 個の参照
        public Form1()
        {
            InitializeComponent();
        }
    }
}
```


2.4. ボタンなどを配置

下図のようにボタンなどを配置します。



2.5. 「接続」と「送信出力値」取得

RFID リーダ・ライタと接続する API 関数は `so_CommSetup()` です。(詳細は API 仕様書を参照)
 このアプリでは Form1 の Load イベント内で接続しています。

下の関数では、CB ファミリの RFID リーダ・ライタと USB で接続しています。

```
this.m_iCommNo = URWSPAPI.so_CommSetup(0, 1, 0);
```

これ以降、他の API 関数を呼ぶときに、この関数の戻り値 `m_iCommNo` を第一引数として渡します。

```
namespace Sample
{
    3 個の参照
    public partial class Form1 : Form
    {
        int m_iCommNo = -1;
        1 個の参照
        public Form1()
        {
            InitializeComponent();
        }
        // Form Load イベント
        1 個の参照
        private void Form1_Load(object sender, EventArgs e)
        {
            short PA = 0, DA = 0, Offset = 0;
            this.m_iCommNo = URWSPAPI.so_CommSetup(0, 1, 0);
            labelTagID1.Text = "-----";
            labelTagID2.Text = "-----";

            for (byte ant = 1; ant < 3; ant++)
            {
                URWSPAPI.so_GetTxPowerAnt(this.m_iCommNo, ant, ref PA, ref DA, ref Offset);
                if (ant == 1)
                    textBoxPower1.Text = (PA / 100).ToString();
                else
                    textBoxPower2.Text = (PA / 100).ToString();
            }
        }
    }
}
```

ここで定義

接続

アンテナ1と2の送信出力値取得

アンテナ1と2の送信出力値表示

`so_CommSetup()` の引数は、RFID リーダ・ライタのファミリー(f/CB)や接続方法(USB/RS-232C)により異なります。

また、LAN 接続するときは別関数の `so_CommSetupIP()` を使います。

送信出力値はアンテナごとに設定/取得できます。送信出力値を取得する API 関数は `so_GetTxPowerAnt()` です。(詳細は API 仕様書を参照)

```
URWSPAPI.so_GetTxPowerAnt(this.m_iCommNo, ant, ref PA, ref DA, ref Offset);
```

第一引数は `so_CommSetup()` からの戻り値です。

第二引数にアンテナ番号を設定します。

第三引数に送信出力値が返ります。100 倍された値を取得するため 100 で割ってから表示します。

2.6. 「切断」

RFID リーダ・ライタと切断する API 関数は so_CommDelete() です。

このアプリでは Form1 の FormClosing イベント内で切断しています。

`URWSPAPI.so_CommDelete(this.m_iCommNo);`

引数は so_CommSetup() からの戻り値です。

```

// Form Closing イベント
1 個の参照
private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    URWSPAPI.so_CommDelete(this.m_iCommNo);
}
    
```

切断

2.7. 送信出力値「設定」

送信出力値を設定する API 関数は so_SetTxPAPowerAnt() です。

`URWSPAPI.so_SetTxPAPowerAnt(this.m_iCommNo, 1, PA);`

第一引数は so_CommSetup() からの戻り値です。

第二引数にアンテナ番号を設定します。

第三引数に送信出力値を指定します。100 倍した値を設定します。

```

// 送信出力「設定」ボタン
1 個の参照
private void buttonSetPower_Click(object sender, EventArgs e)
{
    short PA = 0;
    for (byte ant = 1; ant < 3; ant++)
    {
        if (ant == 1)
            PA = (short)(Int16.Parse(textBoxPower1.Text) * 100);
        else
            PA = (short)(Int16.Parse(textBoxPower2.Text) * 100);
        URWSPAPI.so_SetTxPAPowerAnt(this.m_iCommNo, ant, PA);
    }
}
    
```

アンテナ1と2の
送信出力値

アンテナ1と2の
送信出力値設定

2.8. 「アンテナ切替え」と「タグ ID 取得」

下のシーケンスでタグ ID を取得します。

- (1) アンテナ1に切替え
- (2) tag ID を取得
- (3) アンテナ2に切替え
- (4) tag ID を取得

アンテナ切替えをする API 関数は `so_SetAntennaSwitchNum()` を使います。

`URWSPAPI.so_SetAntennaSwitchNum(this.m_iCommNo, byte antennaNum);`

第一引数は `so_CommSetup()` からの返り値です。

第二引数は切り替えるアンテナ番号です。

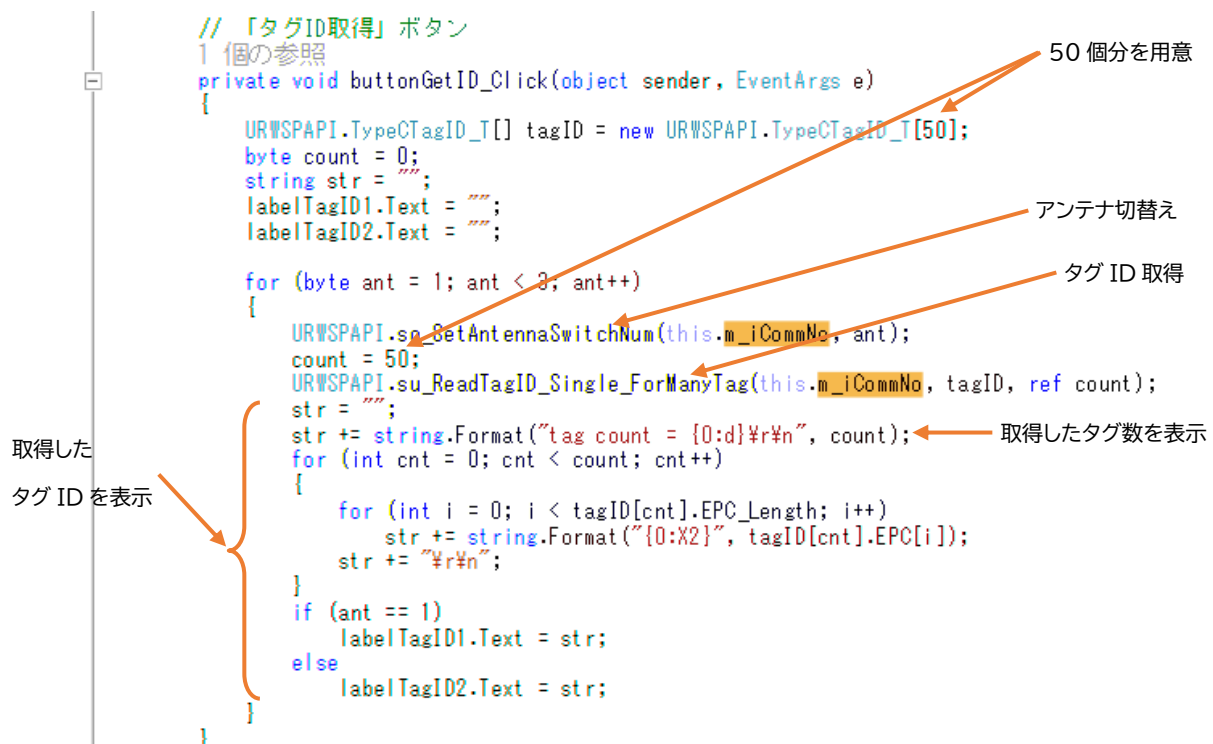
またタグ ID を取得する API 関数は `su_ReadTagID_Single_ForManyTag()` を使います。

`URWSPAPI.su_ReadTagID_Single_ForManyTag(this.m_iCommNo, tagID, ref count);`

第一引数は `so_CommSetup()` からの返り値です。

第二引数は取得するタグ ID を返す構造体です。

第三引数は取得する最大タグ数を渡し、取得後は実際に取得したタグ数が返ります。



```

// 「タグID取得」 ボタン
1 個の参照
private void buttonGetID_Click(object sender, EventArgs e)
{
    URWSPAPI.TypeCTagID_T[] tagID = new URWSPAPI.TypeCTagID_T[50];
    byte count = 0;
    string str = "";
    labelTagID1.Text = "";
    labelTagID2.Text = "";

    for (byte ant = 1; ant < 3; ant++)
    {
        URWSPAPI.so_SetAntennaSwitchNum(this.m_iCommNo, ant);
        count = 50;
        URWSPAPI.su_ReadTagID_Single_ForManyTag(this.m_iCommNo, tagID, ref count);
        str = "";
        str += string.Format("tag count = {0:d}¥r¥n", count);
        for (int cnt = 0; cnt < count; cnt++)
        {
            for (int i = 0; i < tagID[cnt].EPC_Length; i++)
                str += string.Format("{0:X2}", tagID[cnt].EPC[i]);
            str += "¥r¥n";
        }
        if (ant == 1)
            labelTagID1.Text = str;
        else
            labelTagID2.Text = str;
    }
}
    
```

取得したタグ ID を表示

50 個分を用意

アンテナ切替え

タグ ID 取得

取得したタグ数を表示

※ アンテナ切替関数

アンテナを 4ch や 8ch 搭載している RFID リーダ・ライタでは

(1) `so_SetAntennaSwitchNum()`

で切り替えます。

一方、アンテナを 2ch しか搭載していない RFID リーダ・ライタでは

(2) `so_SetAntennaPort()`

で切り替えます。

お使いの RFID リーダ・ライタではどちらの関数を使うかは

`so_GetAntennaSwitchState()`の第二引数の返り値で判断できます。

この値が 0x01(アンテナ切替器を使用する)のときは(1)を、0x00(アンテナ切替器を使用しない)のときは(2)を使います。

※ タグ ID 取得関数

このドキュメントでは `su_ReadTagID_Single_ForManyTag()`を使っています。

もし取得するタグ数が 14 個以下であれば `su_ReadTagID_Single()`もお使いいただけます。

以上