

製品仕様書

製品名 ABS-V31 制御用R/W向けクラスライブラリ

形式 V31Control.dll

仕様書名 リファレンスマニュアル

仕様書番号 M1025-8207A07

総構成	リファレンスマニュアル	M1025-8207A
	ABS-V31 ハードウェア仕様書	ME480-7321A
	ABS-V31 ソフトウェア仕様書	ME480-7330A
	ABS-V31U ハードウェア仕様書	M1054-7695A
	ABS-V31U ソフトウェア仕様書	M1054-7696A
	ABS-V31UWI-I3 ハードウェア仕様書	M1625-8371A
	ABS-V31UWI-I3 ソフトウェア仕様書	M1625-8372A
	ABS-L31U ハードウェア仕様書	M2040-9196A
	ABS-L31U ソフトウェア仕様書	M2040-9197A

当社の承諾なく第三者に開示することはできません。

三和ニューテック株式会社
開発部

2022年5月31日		
承認	審査	作成
		







変更経歴書

版	記号	年月日	項番	変更内容	変更理由	担当
00	—	2013/07/05	—	新規作成	—	近藤
00	—	2013/07/05	以下、リファレンスマニュアル(M1025-7380A06)からの変更点			近藤
			1-1	動作確認済み OS 内容変更	Windows8 対応による	
			3-5-10	パラメータ詳細説明追記	詳細説明記載のため	
			3-5-13	パラメータ詳細説明追記	詳細説明記載のため	
			3-5-15	パラメータ詳細説明追記	詳細説明記載のため	
			3-5-16	パラメータ詳細説明追記	詳細説明記載のため	
			3-5-17	パラメータ詳細説明追記	詳細説明記載のため	
			3-5-18	パラメータ詳細説明追記	詳細説明記載のため	
			3-5-19	パラメータ詳細説明追記	詳細説明記載のため	
			3-5-20	パラメータ詳細説明追記	詳細説明記載のため	
			3-5-21	パラメータ詳細説明追記	詳細説明記載のため	
			3-5-25	取得バージョン情報フォーマット追記	フォーマット記載のため	
			4	Windows8 での USB ドライバインストール手順追記	Windows8 対応による	
			5	Windows8 での USB ドライバアンインストール手順追記	Windows8 対応による	
			6	付録追記	パラメータ詳細説明記載追加に伴う補足付表追記のため	
01	△	2013/07/19	1-2	OS、.NET Framework バージョン対応表に本クラスライブラリのバージョンを追記	記載漏れのため	近藤
				対象 OS に該当する .NET Framework バージョンのクラスライブラリを使用する注意事項を追記	対象 OS による該当クラスライブラリを明確にするため	
02	△	2014/10/31	1-1	対象機種表に、「ABS-V31UWI」項を追記	ABS-V31UWI-I3 の ISO 様式磁気リット/ライト仕様対応に伴う機能追加のため	近藤
			1-2	バージョンを 3.1.0.0 に変更、		
			1-2, 4-3, 5-2	Windows8.1 を追加		
			3-5-13	・リットモード指定 (35h=ISO 様式リット) 追記 ・機種が「ABS-V31UWI」の場合のリットモード指定注意事項追記 ・リットデータチェック指定 (32h=奇数パリティ有り) 追記 ・機種が「ABS-V31UWI」の場合のリットデータチェック指定注意事項追記 ・リットモード指定=35h 時のリットデータチェック指定/トラック指定値範囲を対応表に追記		
			3-5-29	V31Control. WriteForISO メソッド詳細追記		
			6-4 6-5	ISO 様式使用文字一覧表追記		
			全項	以下の語句を統一 ・バックアップ→バックアップ ・クリア→クリア ・キャラクター→キャラクター ・モーター→モーター ・フリンター→フリンター ・パラメータ→パラメータ	語句統一のため	

[illegible]

目 次

～はじめに～	1
1. クラスライブラリの概要	2
1-1. 対象機種	2
1-2. 動作確認済みOSと環境	2
1-3. 開発環境	3
1-4. 使用可能な通信デバイス	3
1-5. 本クラスライブラリの名前空間とアセンブリ情報	3
1-6. 本クラスライブラリ固有の注意点	3
1-6-1. Windows Formで使用する場合	4
1-6-2. Windows Form以外での呼び出し	4
2. サンプルプログラム	5
3. 関数リファレンス	6
3-1. クラスライブラリの構成	6
3-2. リーダーライターレスポンスクラス (V31Response)	7
3-3. シリアル通信リーダーライター制御クラス (V31SerialControl)	9
3-3-1. V31SerialControl.PortNo プロパティ	10
3-3-2. V31SerialControl.PortName プロパティ	10
3-3-3. V31SerialControl.BaudRate プロパティ	11
3-3-4. V31SerialControl.Parity プロパティ	12
3-3-5. V31SerialControl.StopBits プロパティ	13
3-3-6. V31SerialControl.CTS プロパティ	14
3-3-7. V31SerialControl.VbBaudRate プロパティ	15
3-3-8. V31SerialControl.VBParity プロパティ	16
3-3-9. V31SerialControl.VBStopBits プロパティ	17
3-3-10. V31SerialControl.GetPortNames() メソッド	18
3-3-11. V31SerialControl.ChangeCTS イベント	19
3-4. USB通信リーダーライター制御クラス (V31USBControl)	20
3-5. リーダーライター制御クラス (V31Control)	20
3-5-1. V31Control.Timeout プロパティ	22
3-5-2. V31Control.RetryTime プロパティ	23
3-5-3. V31Control.IsOpen プロパティ	24
3-5-4. V31Control.LockEvents プロパティ	24
3-5-5. V31Control.GetNecessaryBytes() メソッド	25
3-5-6. V31Control.V31Control() コンストラクタ	25
3-5-7. V31Control.Dispose() メソッド	26
3-5-8. V31Control.Close() メソッド	27
3-5-9. V31Control.Open() メソッド	28
3-5-10. V31Control.Initialize() メソッド	29
3-5-11. V31Control.StatusRead() メソッド	31
3-5-12. V31Control.GetSensorStatus() メソッド	32
3-5-13. V31Control.Read() メソッド	33
3-5-14. V31Control.Cancel() メソッド	36
3-5-15. V31Control.Write() メソッド	37
3-5-16. V31Control.RegisterGaiji() メソッド	38
3-5-17. V31Control.RegisterImage() メソッド	41
3-5-18. V31Control.PrintSetting() メソッド	43
3-5-19. V31Control.PrintImage() メソッド	45
3-5-20. V31Control.PrintString() メソッド	48
3-5-21. V31Control.PrintBarcode() メソッド	51
3-5-22. V31Control.Eject() メソッド	54
3-5-23. V31Control.Cleaning() メソッド	55
3-5-24. V31Control.SetClock() メソッド	56
3-5-25. V31Control.GetVersion() メソッド	57

3-5-26.	V31Control.GetClock() メソッド	58
3-5-27.	V31Control.GetResponse() メソッド	59
3-5-28.	V31Control.SendCommand() メソッド	60
3-5-29.	V31Control.WriteForISO() メソッド 	61
4.	USBドライバーのインストール方法	63
4-1.	共通事項	63
4-2.	Windows7 の場合	63
4-3.	Windows8/8.1/10/11 の場合   	67
4-3-1.	デジタル証明書のインストール	67
4-3-2.	USBドライバーのインストール	71
5.	USBドライバーのアンインストール方法	75
5-1.	Windows7 の場合	75
5-2.	Windows8/8.1/10 の場合  	78
5-3.	Windows11 の場合 	81
6.	付録	83
6-1.	レスポンス・ステータス一覧表	83
6-2.	コマンド・レスポンス対応表	84
6-3.	印字ANK文字コード表	85
6-4.	ISO様式トラック1 使用文字一覧 	86
6-5.	ISO様式トラック2、3 使用文字一覧 	87
6-6.	印字範囲1「文字印字（行指定モード）」	88
6-7.	印字範囲2「イメージデータ・文字（任意座標モード）・バーコード」	90

～はじめに～

本クラスライブラリは、三和ニューテック株式会社製の磁気カードターミナル「ABS-V31」を使用したシステムのアプリケーション開発を容易にするためのクラスライブラリです。

本クラスライブラリを利用することで、「ABS-V31」とのシリアルやUSBといった通信プロトコルの詳細を意識することなく、簡単に「ABS-V31」が提供する通信機能の全てを操作することができます。

注意) 本クラスライブラリを用いてアプリケーションを作成する際には、本マニュアルだけではなく、必ず各対象機種種の「ソフトウェア仕様書」を参照するようにして下さい。将来において、レスポンス、引数など本マニュアルの記載とソフトウェア仕様書が異なる場合、ソフトウェア仕様書の記載に沿ってアプリケーションを作成して下さい。

●著作権など●

Windows、Windowsロゴは、米国Microsoft Corporationの米国およびその他の国における登録商標です。

Microsoftおよびそのロゴは、米国Microsoft Corporationの米国およびその他の国における商標または登録商標です。

表記中の商標および登録商標は各社に帰属します。

本製品の仕様、および関連資料その他は予告なく変更することがあります。

本製品は、一部のハードまたはソフトウェアでは動作しない場合がありますので、予めご了承下さい。

1. クラスライブラリの概要

本クラスライブラリを用いてアプリケーションを作成する際に考慮すべき事柄について列挙します。

1-1. 対象機種

本クラスライブラリを用いて制御可能となる機種は、以下の通りとなります。

用語	対象機種	備考
「ABS-V31」	ABS-V31	RS-232C 対応
	ABS-V31U	RS-232C/USB 対応
	ABS-L31U [△]	RS-232C/USB 対応 [△]
「ABS-V31UWI」 [△]	ABS-V31UWI-I3 [△]	USB 対応 磁気3トラック ISO 様式 リード/ライト対応 [△]

1-2. 動作確認済み OS と環境

本クラスライブラリは、以下の環境で動作することを確認しております。

また、下記のOS、.NET FrameworkとWindowsデスクトップアプリケーションの組み合わせで動作確認を行っております。

バージョン [△]	対応OS	.NET Framework バージョン
3.1.0.0 ^{△△}	Windows 7 32bit / 64bit 日本語版	3.5
	Windows 8/8.1/10 32bit/64bit 日本語版 ^{△△}	4.5
	Windows 11 64bit 日本語版 [△]	

上記以外の環境で使用する場合には、アプリケーション開発時に十分にテストを行うようお願いいたします。ただし、USBを用いて開発を行う場合、弊社提供USBドライバーが動作する環境に依存します。

USBドライバーのインストール手順については、「4. USBドライバーのインストール方法」を参照してください。

.NET Framework 3.5/4.5 動作環境については、Microsoft社ダウンロードセンターにある、.NET Framework 3.5/4.5 サイトを参照してください。

注) これまでにリリースしていた旧バージョン(Ver. 2.3.0.0 以前)に添付のUSB ドライバー(sntv31usb.sys)は、32bit OS 専用となっており、Windows8、及び、Windows7 64bitに対応していません。

本クラスライブラリに添付のUSBドライバーは、32bit、64bitの両OSに対応しています。

[△] 注) 上記表の通り、対象OSに該当する.NET Frameworkバージョンのクラスライブラリを使用するようにしてください。

本クラスライブラリのアセンブリ(SanwaNewtec.DeviceControls.V31Control.dll)のプロパティ画面にて、どの.NET Frameworkバージョンであるかを確認できます。

1-3. 開発環境

開発環境は、.NET Framework 3.5/4.5の開発要件に加えて、作成するアプリケーションの規模や開発言語によっても異なります。

本リファレンス、添付したサンプルソースは、C#言語で記述いたします。他のVB.NETなどの.NET言語であれば、本クラスライブラリと組み合わせてアプリケーションの作成を行うことができます。開発着手前に十分にご検討下さい。

1-4. 使用可能な通信デバイス

本クラスライブラリでシリアル通信を用いてリーダーライターの制御を行う場合、WindowsAPIを使ってシリアル通信デバイスの操作を行います。

従って、OSが認識する1～9の範囲のシリアルポートであれば、通常問題なく動作すると考えられます。[④](#)

しかし、ごくまれに古いUSBタイプのシリアルポートを使用する場合に、シリアルポートメーカーから最新のドライバーを入手するなど、本クラスライブラリを使ってアプリケーションを開発する側での対応が必要になる可能性があります。

また、本クラスライブラリでUSB通信を用いてリーダーライターの制御を行う場合、ターゲットホストのUSBドライバーが正しく認識されている必要があります。

1-5. 本クラスライブラリの名前空間とアセンブリ情報

本クラスライブラリの名前空間とアセンブリ情報を下記に示します。お使いのクラスライブラリのアセンブリ情報をご確認下さい。

項目	値
名前空間	SanwaNewtec.DeviceControls
アセンブリ名	SanwaNewtec.DeviceControls.V31Control.dll
言語	ニュートラル言語
GUID	8b83a109-699d-4fe0-af6c-f2a836e182da
公開キー	00 24 00 00 04 80 00 00 94 00 00 00 06 02 00 00 00 24 00 00 52 53 41 31 00 04 00 00 01 00 01 00 C9 03 04 9B E8 CD AC 2A D2 DC 95 CC 20 97 08 D6 60 B0 61 C2 51 48 66 87 07 D5 FE 86 ED 86 7A 7B 65 5C 55 6F E1 94 80 2C 50 B0 C8 E0 8A C4 D2 46 AA FE DE 18 4E 54 E7 55 8D 8A DE 60 A9 B1 9D 61 BF A4 B7 EF 93 D2 CE B5 A7 98 28 D6 91 EB B6 67 42 AF B7 58 B7 AC 8F E5 B5 B8 22 DB 17 82 F9 8F C9 A2 BF B4 F3 A6 18 CA 9B 41 85 F0 12 16 D3 A3 A0 18 60 7C A4 DB CD 4E 0B E8 2E 18 A7 29 C2 CA
トークン	50 59 84 83 E4 5D C6 94

1-6. 本クラスライブラリ固有の注意点

本クラスライブラリを使ったアプリケーションの開発は、.NET Frameworkが提供する技術に大きく依存しています。本書では、.NETテクノロジそのものについての説明は行いませんので、アプリケーション作成の際には、同技術に関するMicrosoft社の技術文書や、Microsoft社が提供するMSDN(<http://msdn.microsoft.com/ja-jp/default.aspx>)などをご活用下さい。

本クラスライブラリでは、リーダーライターとの通信を、アプリケーションスレッドとは別のスレッドで行っています。このため、本クラスライブラリから提供されるクラスをインスタンス化する場合には、インスタンスの破棄を確実に行うようにしてください。

これが行われない場合、アプリケーションの動作が不安定になる可能性があります。

以下、プログラミングに関するヒントを示します。

1-6-1. Windows Form で使用する場合

Windows Formで使用する場合、フォームのロード時にインスタンスを作成し、フォームのクローズ時にインスタンスのDispose() メソッドを呼び出すようにして下さい。
このようにすることで、フォームのクローズ時に確実にクラスライブラリ内のスレッドを安全に停止できます。

```
public class MyForm: Form
{
    V31SerialControl _v31= null ; // インスタンス
    // フォームをロードする時
    private void MyForm_Load(object sender, EventArgs e )
    {
        _v31= new V31SerialControl() ; // インスタンス作成
    }
    // フォームを破棄する時
    private void MyForm_FormClosed(object sender, FormClosedEventArgs e)
    {
        _v31.Dispose() ; // インスタンスを破棄
    }
}
```

1-6-2. Windows Form 以外での呼び出し

インスタンスを生成後、使用し終わったらDispose() メソッドを確実に呼び出して下さい。
また、例外を確実に捕捉するコードを記述することで、作成するアプリケーションをより堅牢にすることができます。

```
using(V31SerialControl v31= new V31SerialControl() ){
    try {
        v31.PortName= "COM5" ; // COM5ポートを使用する
        v31.Open() ;           // 通信デバイスを開く
        v31.Initialize() ;     // 初期化コマンド実行
    }
    catch(Exception ex ){
        // エラーが発生したら要因を表示する
        MessageBox.Show( ex.Message ) ;
    }
    finally {
        v31.Close() ; // エラーが発生しても確実に通信デバイスを閉じる
    }
}
```

2. サンプルプログラム

本製品には、C#言語で記述されたサンプルプログラムが添付されています。本クラスライブラリが提供する全てのコマンドの呼び出し方法が記述されていますので、参考にして下さい。

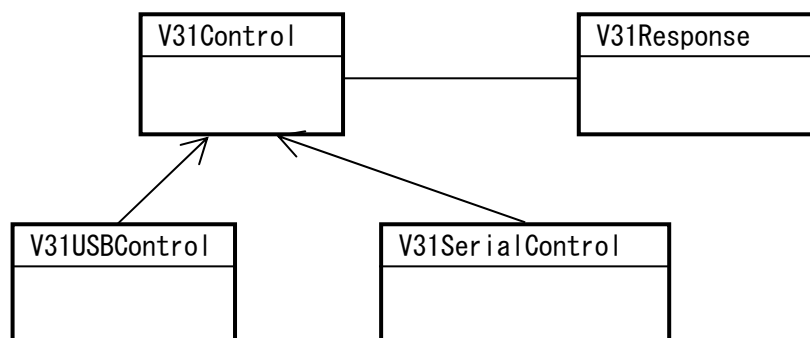
Microsoft Visual Studioでソリューションを開き、サンプルプログラムをビルドしたり、デバッガで処理の追跡を行うことで、クラスライブラリへの理解をより深めることができると思います。

サンプルプログラムは、クラスライブラリの使用方法の一例を示しただけで、アプリケーション作成へのアプローチに対してなんら制限を加えるものではありません。実際のアプリケーション開発では、サンプルコードに囚われることなく、自由な発想でアプリケーションに最適なソリューションを構築してみてください。

3. 関数リファレンス

3-1. クラスライブラリの構成

クラスライブラリは、(1)リーダーライターからのレスポンスの各項目に簡単にアクセスするためのリーダーライターレスポンスクラス (V31Response) と、コンピューターの通信デバイスを設定し、リーダーライターに対して各コマンドを送信し、そのレスポンスを受信する仮想的な関数インターフェースを持つリーダーライター制御クラス (V31Control) と、これから派生した実デバイスとのアクセスを行うシリアル通信リーダーライター制御クラス (V31SerialControl) とUSB通信リーダーライター制御クラス (V31USBControl) という2つのクラスと、サポートのための列挙型定数から構成されます。



- リーダーライター制御クラス (V31Control) の各メソッド用いて、リーダーライターにコマンドを送信した場合、そのレスポンスのバイトストリームはメソッドの戻り値になります。
作成されるアプリケーションプログラムで、レスポンスデータの全てにアクセスしたい場合には、戻り値を取得して、アプリケーション固有の処理を行うように記述して下さい。
- リーダーライター制御クラス (V31Control) の各メソッドは、リーダーライターレスポンスクラス (V31Response) を参照するタイプのものを含めてオーバーロードされて定義されています。
リーダーライターからのレスポンスの一部だけにアクセスしたい場合には、V31Responseインスタンスを渡して、メソッド呼び出しを行い、V31Responseのプロパティにアクセスして、アプリケーション固有の処理を行うように記述して下さい。
- リーダーライター制御クラス (V31Control) のプロパティ、メソッドの操作において問題が発生した場合、クラスライブラリから例外が送出されます。
アプリケーションは例外を捕捉して例外への対応を行ない、堅牢なシステムを作成されるようお願いいたします。

3-2. リーダーライターレスポンスクラス (V31Response)

リーダーライターへのコマンドに対するレスポンスを保持し、そのフィールドに簡単にアクセスするためのクラスです。このクラスのプロパティは全て読み取り専用です。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public class V31Response
```

パブリックプロパティの一覧

プロパティ	説 明
CMD	レスポンスのコマンド (CMD) を示します。
RPS1	レスポンスのカード状態 (RPS1) を示します。
RPS2	レスポンスのエラー状態 (RPS2) を示します。
RPS3	レスポンスの処理状態 (RPS3) を示します。
Data	レスポンスのデータ列 (バイト列) を示します。データのないコマンドに対する応答では、nullとなります。
DataLength	レスポンスのデータ列長を示します。V31プロトコルの制限から、この値は0～249の範囲になります。データのないコマンドに対する応答では、0を返します。

パブリックメソッドの一覧

パブリックメソッドはありません。

《解説》

以下のプログラムは、リーダーライターに対してカード排出要求を発行し、カードが排出されるまで待機する時の記述例です。

V31Responseを用いた場合と用いない場合のプログラムを比較してみてください。

```
using (V31SerialControl v31= new V31SerialControl() ) {  
    V31Response res= null ; // V31Responseのインスタンスをnullで作成  
    try {  
        v31.PortName = "COM5" ; // COM5ポートにV31が接続  
        v31.Open() ;           // 通信ポートのオープン  
        //  
        // カード排出&カードなし待ち  
        v31.Eject( ref res ) ; // カード排出  
        if(res.RPS2!= 0x30 ) throw new Exception("エラー発生" ) ;  
        while( res.RPS1!= 0x30 ) {  
            Thread.Sleep( 100 ) ;           // 再送要求には100ms以上待機  
            v31.GetResponse( ref res ) ; // 再送要求  
            if(res.RPS2!= 0x30 ) throw new Exception("エラー発生" ) ;  
        }  
        // カード排出完了  
    }  
}
```

```

catch(Exception ex ){
    MessageBox.Show( ex.Message ) ; // エラー表示
}
finally {
    v31.Close() ; // 通信ポートを閉じる
}
}

```

同じ処理をV31Responseを用いずに記述した場合、バイトストリーム列に直接アクセスする必要があり、プログラムの可読性が劣ります。

```

using(V31SerialControl v31= new V31SerialControl() ){
    byte[] res= null ; // バイトストリーム列をnullで作成
    try {
        v31.PortName = "COM5" ; // COM5ポートにV31が接続
        v31.Open() ; // 通信ポートのオープン
        //
        // カード排出&カードなし待ち
        res= v31.Eject() ; // カード排出
        if(res[4]!= 0x30 ) throw new Exception("エラー発生" ) ;
        while( res[3]!= 0x30 ){
            Thread.Sleep( 100 ) ; // 再送要求には100ms以上待機
            res= v31.GetResponse() ; // 再送要求
            if(res[4]!= 0x30 ) throw new Exception("エラー発生" ) ;
        }
        // カード排出完了
    }
    catch(Exception ex ){
        MessageBox.Show( ex.Message ) ; // エラー表示
    }
    finally {
        v31.Close() ; // 通信ポートを閉じる
    }
}

```

3-3. シリアル通信リーダーライター制御クラス (V31SerialControl)

コンピュータのシリアル通信デバイスを制御するクラスです。リーダーライターへのコマンド送信、レスポンス受信は、V31Controlで行います。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public class V31SerialControl: V31Control
```

パブリックプロパティの一覧

プロパティ	説 明
PortNo	通信ポート番号を取得または設定します。
PortName	通信ポート名を取得または設定します。
BaudRate	通信速度（ボーレート）を取得または設定します。
Parity	パリティを取得または設定します。
StopBits	ストップビットを取得または設定します。
CTS	CTSの状態を取得します。

パブリックメソッドの一覧

メソッド名	コメント	説 明
GetPortNames()	—	コンピュータで使用可能なシリアルポート名の配列を返します。
V31SerialControl()	—	V31SerialControlインスタンスを生成します。

パブリックイベントの一覧

イベント	説 明
ChangeCTS	CTSの状態が変化したら発生する。

3-3-1. V31SerialControl.PortNo プロパティ

通信ポート番号を取得、または設定します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte PortNo { get ; set ; }
```

《解説》

- ・ポート番号には、未オープンのシリアルポート番号が指定できます。
- ・ポート番号は、1～9の範囲の整数値です。[△](#)
- ・使用できるポート番号は、コントロールパネルから確認できます。
また、V31SerialControl.GetPortNames() メソッドによりポート名を列挙できます。

《例外》

例 外	要 因
System.IO.IOException	既に開いているポート番号を指定した。 コンピューターで使用できないポート番号を指定した。

3-3-2. V31SerialControl.PortName プロパティ

通信ポート名を取得、または設定します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public string PortName { get ; set ; }
```

《解説》

- ・ポート名は、“COM”で始まる1～9の範囲の整数値で表記されます。[△](#)
- ・ポート名には、未オープンのシリアルポート名が指定できます。
- ・使用できるポート名は、コントロールパネルから確認できます。
また、V31SerialControl.GetPortNames() メソッドによりポート名を列挙できます。

《例外》

例 外	要 因
System.ArgumentNullException	ポート名にnullを指定した。
System.FormatException	ポート名が“COM”で始まる整数値でない。
System.IO.IOException	既に開いているポート番号を指定した。 コンピューターで使用できないポート番号を指定した。

3-3-3. V31SerialControl.BaudRate プロパティ

通信速度（ボーレート）を取得、または設定します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public BAUDRATE BaudRate { get ; set ; }
```

《解説》

- ・通信速度は、BAUDRATE列挙型で指定します。
- ・指定できる通信速度は、1200bps、2400bps、4800bps、9600bps、19200bps、38400bpsの6種ですが、「A B S - V 3 1」で使用できるのは、4800bps、9600bps、19200bps、38400bpsの4種のみです。
- ・低速の通信速度は、他の機器のために設定できるようになっています。
- ・通信速度と列挙名の対応を下記の表で示します。

```
public enum BAUDRATE: int
```

列挙名	通信速度
BAUDRATE._1200	1200bps
BAUDRATE._2400	2400bps
BAUDRATE._4800	4800bps
BAUDRATE._9600	9600bps
BAUDRATE._19200	19200bps
BAUDRATE._38400	38400bps

- ・整数値で指定する場合には、9600などのリテラル値を用いて、(BaudRate)でキャストして使用して下さい。

《例外》

例 外	要 因
System.ArgumentOutOfRangeException	通信速度に上記6種以外の値を設定しようとした。
System.IO.IOException	既に関いているV31SerialControlに対して、通信速度の設定を行おうとした。

3-3-4. V31SerialControl.Parity プロパティ

パリティを取得、または設定します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public PARITY Parity { get ; set ; }
```

《解説》

- ・パリティは、PARITY列挙型で指定します。
- ・指定できるパリティは、「パリティなし」、「偶数パリティ」、「奇数パリティ」の3種類です。
- ・パリティと列挙名の対応を下記の表で示します。

```
public enum PARITY: byte
```

列挙名	パリティ
PARITY.NO	パリティなし
PARITY.EVEN	偶数パリティ
PARITY.ODD	奇数パリティ

- ・整数値をPARITY型にキャストして、パリティの設定を行うことも可能ですが、パリティの値が離散値なので、注意して記述してください。

《例外》

例 外	要 因
System.ArgumentOutOfRangeException	パリティに上記3種以外の値を設定しようとした。
System.IO.IOException	既に関いているV31SerialControlに対して、パリティの設定を行おうとした。

3-3-5. V31SerialControl.StopBits プロパティ

ストップビットを取得、または設定します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public STOPBITS StopBits { get ; set ; }
```

《解説》

- ・ストップビットは、STOPBITS列挙型で指定します。
- ・指定できるストップビットは、「1ビット」、「2ビット」の2種類ですが、「ABS-V31」で使用できるのは、1ビットの1種類のみです。
- ・ストップビットと列挙名の対応を下記の表で示します。

```
public enum STOPBITS: byte
```

列挙名	ストップビット
STOPBITS.ONE	1ビット
STOPBITS.TWO	2ビット

- ・整数値をSTOPBITS型にキャストして、ストップビットの設定を行うことも可能ですが、ストップビットの値が離散値なので、注意して記述してください。

《例外》

例 外	要 因
System.ArgumentOutOfRangeException	ストップビットに上記2種以外の値を設定しようとした。
System.IO.IOException	既にかいているV31SerialControlに対して、ストップビットの設定を行おうとした。

3-3-6. V31SerialControl.CTS プロパティ

CTSの状態を取得します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public bool CTS { get ; }
```

《解説》

- ・リーダーライターは受信準備が整うと、RTS（ホスト側ではCTS）をONにして受信準備が整ったことをホストに通知します。本プロパティを参照することで、リーダーライターの受信準備が完了したこと知ることができます。
- ・CTSのOFF時間を監視することで、リーダーライターとホストが接続されていない、あるいは、リーダーライターの電源がOFFしているなどの判定にも利用することができます。
- ・V31SerialControlが閉じている状態で本プロパティを参照すると例外が送出されます。

《例外》

例 外	要 因
System.IO.IOException	V31SerialControlがクローズ状態である。 通信デバイスの操作に失敗した。

3-3-7. V31SerialControl.VbBaudRate プロパティ

通信速度（ボーレート）を取得、または設定します。VisualBasicによる開発用プロパティです。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public VBEBAUDRATE VbBaudRate { get ; set ; }
```

《解説》

- ・通信速度は、VBEBAUDRATE列挙型で指定します。
- ・指定できる通信速度は、1200bps、2400bps、4800bps、9600bps、19200bps、38400bpsの6種ですが、「ABS-V31」で可以使用するのは、4800bps、9600bps、19200bps、38400bpsの4種のみです。
- ・低速の通信速度は、他の機器のために設定できるようになっています。
- ・通信速度と列挙名の対応を下記の表で示します。

```
public enum VBEBAUDRATE: int
```

列挙名	通信速度
VBEBAUDRATE._1200	1200bps
VBEBAUDRATE._2400	2400bps
VBEBAUDRATE._4800	4800bps
VBEBAUDRATE._9600	9600bps
VBEBAUDRATE._19200	19200bps
VBEBAUDRATE._38400	38400bps

- ・整数値で指定する場合には、9600などのリテラル値を用いて、(VbBaudRate)でキャストして使用して下さい。

《例外》

例 外	要 因
System.ArgumentOutOfRangeException	通信速度に上記6種以外の値を設定しようとした。
System.IO.IOException	既に関いているV31SerialControlに対して、通信速度の設定を行おうとした。

3-3-8. V31SerialControl.VBParity プロパティ

パリティを取得、または設定します。VisualBasicによる開発用プロパティです。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public VBEPARITY VbParity { get ; set ; }
```

《解説》

- ・パリティは、VBEPARITY列挙型で指定します。
- ・指定できるパリティは、「パリティなし」、「偶数パリティ」、「奇数パリティ」の3種類です。
- ・パリティと列挙名の対応を下記の表で示します。

```
public enum VBEPARITY: byte
```

列挙名	パリティ
VBEPARITY.NO	パリティなし
VBEPARITY.EVEN	偶数パリティ
VBEPARITY.ODD	奇数パリティ

- ・整数値をVBEPARITY型にキャストして、パリティの設定を行うことも可能ですが、パリティの値が離散値なので、注意して記述してください。

《例外》

例 外	要 因
System.ArgumentOutOfRangeException	パリティに上記3種以外の値を設定しようとした。
System.IO.IOException	既に関いているV31SerialControlに対して、パリティの設定を行おうとした。

3-3-9. V31SerialControl.VBStopBits プロパティ

ストップビットを取得、または設定します。VisualBasicによる開発用プロパティです。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public VBESTOPBITS VbStopBits { get ; set ; }
```

《解説》

- ・ストップビットは、VBESTOPBITS列挙型で指定します。
- ・指定できるストップビットは、「1ビット」、「2ビット」の2種類ですが、「ABS-V31」で使用できるのは、1ビットの1種類のみです。
- ・ストップビットと列挙名の対応を下記の表で示します。

```
public enum VBESTOPBITS: byte
```

列挙名	ストップビット
VBESTOPBITS.ONE	1ビット
VBESTOPBITS.TWO	2ビット

- ・整数値をVBESTOPBITS型にキャストして、ストップビットの設定を行うことも可能ですが、ストップビットの値が離散値なので、注意して記述してください。

《例外》

例 外	要 因
System.ArgumentOutOfRangeException	ストップビットに上記2種以外の値を設定しようとした。
System.IO.IOException	既にかいているV31SerialControlに対して、ストップビットの設定を行おうとした。

3-3-10. V31SerialControl.GetPortNames() メソッド

コンピューターで使用可能なシリアルポート名の配列を返します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public static string[] GetPortNames()
```

《解説》

- ・コンピューターで使用可能なシリアルポート名の配列を返します。
- ・現在、オープン中のシリアルポート名は含まれません。
- ・シリアルポート名は、“COM”で始まる 1 ～ 9 の範囲の整数値で表記されます。Ⓐ

《引数》

なし。

《戻り値》

シリアルポート名を示す文字列の配列。

《例外》

なし。

《コード例》

- ・本メソッドを用いて、System.Windows.Forms.ComboBoxにシリアルポート名の一覧を設定するコード例を示します。

```
using System.Windows.Forms ;
using SanwaNewtec.DeviceControls ;

class MyForm: Form
{
    System.Windows.Forms.ComboBox cbPortName ; // ポート名を保持するコンボボックス
    public MyForm()
    {
        cbPortName= new ComboBox() ;
    }
    private void MyForm_Load(object sender, EventArgs e )
    {
        // コンピューターで利用できるシリアルポート名を列挙し、コンボボックスに
        // 項目リストとして設定する。
        cbPortName.Items.AddRange( V31SerialControl.GetPortNames() ) ;
    }
}
```


3-3-11. V31SerialControl.ChangeCTS イベント

CTSの状態が変化したら発生する。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public delegate void ChangeCTSEventHandler(object sender, CTSEventArgs e)
public event ChangeCTSEventHandler ChangeCTS
```

《解説》

- ・リーダーライターのRTS（ホスト側CTS）オン、オフに合わせて発生するイベントです。
- ・V31SerialControlがオープンした状態で発生します。
- ・CTSのON/OFFは、CTSEventArgsのCTSプロパティで参照できます。
- ・イベントハンドラは、V31SerialControlの通信スレッドから呼び出されるため、イベントハンドラ内でWindows Formのコントロールの操作を行う場合、Invoke呼び出しを行い、スレッド境界を越えるよう配慮してください。
- ・イベントハンドラを登録し、Invoke呼び出しを行う際、V31Control.LockEventsプロパティを真にしないで下さい。アプリケーションのメッセージループ呼び出しが抑制されるため、本イベントが発生した場合、V31SerialControlの通信スレッドが停止し、その結果、アプリケーションが停止します。

《プログラム例》

下記のプログラムは、CTSオン、オフイベント発生時にSystem.Windows.Forms.Labelのテキスト文字列を変更し、ユーザーに視覚的に通知します。

```
class MyForm: Form
{
    V31SerialControl _v31= null ;
    System.Windows.Forms.Label lbCTS ; // CTS-ON, OFF 表示ラベル
    private void MyForm_Load(object sender, EventArgs e )
    {
        _v31= new V31SerialControl() ;
        // イベントハンドラの登録
        _v31.ChangeCTS+= new V31SerialControl.ChageCTSEventHandler (OnChangeCTSHandler ) ;
    }
    private void OnChangeCTSHandler(object sender, CTSEventArgs e )
    {
        // Invokeで匿名メソッド呼び出し
        Invoke( new MethodInvoker(
            delegate {
                lbCTS.Text= e.CTS? "CTS-ON": "CTS-OFF" ; // ラベルの表示の変化
            }
        ) ) ;
    }
}
```

3-4. USB 通信リーダーライター制御クラス (V31USBControl)

コンピュータのUSBホストコントローラーを制御するクラスです。

本クラスを使用するためには、ターゲットホストに弊社提供USBドライバーを予めインストールしておく必要があります。リーダーライターへのコマンド送信、レスポンス受信はV31Controlで行います。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public class V31USBControl: V31Control
```

USBには、通信速度などの設定パラメーターがないため固有のプロパティ、メソッドは存在しません。

3-5. リーダーライター制御クラス (V31Control)

コンピュータの通信デバイスを制御し、リーダーライターへコマンドを送信し、そのレスポンスを取得するためのクラスです。実際の通信デバイスへの詳細な設定は、通信デバイス毎に設けられたV31SerialControlとV31USBControlで行います。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public class V31Control: IDisposable
```

パブリックプロパティの一覧

プロパティ	説 明
Timeout	コマンド実行時のタイムアウトを取得または設定します。
RetryTime	リーダーライターが無応答時にコマンドを再送するまでの待機時間を取得または設定します。
IsOpen	通信デバイスが開いていれば真を、閉じていれば偽を返します。
LockEvents	コマンド実行時におけるメッセージループの無効状態を取得、または設定します。

パブリックメソッドの一覧

メソッド名	コマンドコード	説明
GetNecessaryBytes()	—	イメージの幅と高さ（ドット単位）を与えると、そのイメージを保持するために必要なメモリサイズをバイト数で返します。
V31Control()	—	V31Controlのインスタンスを生成します。
Dispose()	—	V31Controlのインスタンスを破棄します。
Close()	—	通信デバイスを解放し、V31Controlをクローズします。
Open()	—	通信デバイスを取得し、V31Controlを開きます。
Initialize()	10h	イニシャルコマンドを実行します。
StatusRead()	20h	ステータスリードコマンドを実行します。
GetSensorStatus()	21h	リーダーライター搬送路のセンサー情報を取得します。
Read()	33h	リードコマンドを実行します。
Cancel()	40h	キャンセルコマンドを実行します。
Write()	53h	ライトコマンドを実行します。
RegisterGaiji()	76h	外字登録コマンドを実行します。
RegisterImage()	77h	イメージデータ登録コマンドを実行します。
PrintSetting()	78h	印字設定コマンドを実行します。
PrintImage()	7Bh	イメージデータ印字コマンドを実行します。
PrintString()	7Ch	文字印字コマンドを実行します。
PrintBarcode()	7Eh	バーコード印字コマンドを実行します。
Eject()	80h	カード排出コマンドを実行します。
Cleaning()	A0h	クリーニングコマンドを実行します。
SetClock()	E1h	時計設定コマンドを実行します。
GetVersion()	F0h	バージョン取得コマンドを実行します。
GetClock()	F1h	現在時刻取得コマンドを実行します。
GetResponse()	—	レスポンス再送要求（ENQ送信）を実行します。
SendCommand()	—	任意のコマンド列を送信します。

3-5-1. V31Control.Timeout プロパティ

コマンド実行時のタイムアウトを取得、または設定します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public int Timeout { get ; set ; }
```

《解説》

- ・V31Controlは、リーダーライターへのコマンド送信の際、リーダーライターからレスポンスが返るまで、また、返ったレスポンスが正しいフォーマットであるか確認を行い、問題があれば、コマンドやENQの再送信（USBであれば、インタラプトトランザクションの実行）を行います。
- ・本Timeoutプロパティは、このV31Controlのタイムアウト時間をミリ秒単位で設定し、コマンド送信、すなわち、V31Controlに対するメソッド呼び出しからこの時間が経過すると、リーダーライターとの通信中であっても処理を中断して、アプリケーションに復帰します。
- ・タイムアウトが発生してアプリケーションに復帰する場合には、例外としてTimeoutExceptionが送出されます。
- ・Timeout値として、50～60,000[msec]まで設定可能です。

《例外》

例 外	要 因
System.ArgumentOutOfRangeException	タイムアウトに不正な値を設定しようとした。
System.InvalidOperationException	リーダーライターと通信中のV31Controlに対して、タイムアウトの設定を行おうとした。

3-5-2. V31Control.RetryTime プロパティ

リーダーライターが無応答時にコマンドを再送するまでの待機時間を取得、または設定します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public int RetryTime { get ; set ; }
```

《解説》

- ・V31Controlは、リーダーライターへのコマンド送信後、リーダーライターが無応答の場合、再度コマンドを送信します。この時の無応答と判定してからコマンドを再送するまでの待機時間をミリ秒単位で設定します。
- ・具体的にはコマンド送信からACK受信までの待機時間と、ENQ送信からレスポンス受信までの時間のことです。（USBの場合、NAK受信後からの再送時間になります）
- ・RetryTime値として、10～1,000[msec]まで設定可能です。
- ・RetryTime値より短い時間で、Timeout値が設定された場合、Timeout値の方が優先されます。このような矛盾した設定を行っても例外は送出されませんのでご注意ください。

《例外》

例 外	要 因
System.ArgumentOutOfRangeException	RetryTimeに不正な値を設定しようとした。
System.InvalidOperationException	リーダーライターと通信中のV31Controlに対して、タイムアウトの設定を行おうとした。

3-5-3. V31Control.IsOpen プロパティ

V31Controlが開いていれば真を、閉じていれば偽を返します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public bool IsOpen { get ; }
```

《解説》

- ・ V31Controlインスタンスに対し、Open() メソッドを呼び出すと、V31Controlは通信デバイスをオープンし、リーダーライターとの送受信準備を行います。本プロパティを参照することで、RWControlが開いているかどうかを知ることができます。
- ・ 本プロパティが真を返すとき、通信デバイスの設定を行うことはできません。

《例外》

なし

3-5-4. V31Control.LockEvents プロパティ

V31Controlのコマンド実行時にWindowsメッセージループの無効状態を取得、あるいは設定します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public bool LockEvents { get ; set ; }
```

《解説》

- ・ V31Controlを使ってリーダーライターにコマンド送信を行った場合、レスポンスが返るかタイムアウトが発生するまで呼出元の処理は停止されます。この時、呼出元のWindowsメッセージループを無効化するかどうかを本プロパティで設定できます。
- ・ インスタンス生成時のデフォルト値は、偽、すなわち、Windowsメッセージループが有効になっています。
- ・ このプロパティが真で、Windows Formを使用するとき、V31Controlがリーダーライターと通信中の場合、Windows Formのコントロールの制御が行えないことがあります。

《例外》

例 外	要 因
System.InvalidOperationException	リーダーライターと通信中のRWControlに対して、Windowsメッセージループの有効／無効化を行おうとした。

3-5-5. V31Control.GetNecessaryBytes() メソッド

イメージの幅と高さ（ドット単位）を与えると、そのイメージを保持するために必要なメモリサイズをバイト数で返します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public static uint GetNecessaryBytes(ushort width, ushort height)
```

《解説》

- ・ イメージは、1 ドット = 1 ビットで表現されますが、イメージ幅は1 バイト単位で管理されるため、幅 1 ドット × 高さ 10 ドットのイメージであっても、1 バイト × 10 ケ = 10 バイトのメモリが必要になります。
- ・ 本メソッドは、この必要なメモリサイズをバイト単位でします。

《引数》

width ドット単位でのイメージ幅
height ドット単位でのイメージ高さ

《戻り値》

イメージを保持するために必要なバイト数。

《例外》

なし

3-5-6. V31Control.V31Control() コンストラクタ

V31Controlのインスタンスを生成します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public V31Control()
```

《解説》

V31Controlクラスのデフォルトコンストラクタです。

《引数》

なし。

《例外》

なし。

3-5-7. V31Control.Dispose() メソッド

V31Controlのインスタンスを破棄します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public void Dispose()
```

《解説》

- ・ V31Controlインスタンスによって使用されているアンマネージリソースを解放します。
V31Controlインスタンスを使い終わったら必ず呼び出す必要があります。
- ・ 確実な方法は、前述しましたが、Windows Formであれば、フォームのロードでインスタンスを生成し、フォームのクローズ時にDispose()呼び出しを行うことです。
- ・ Visual Basic .NET、Visual C#であれば、Using (C#ではusing) 文を使うことで、Dispose()呼び出しを確実に行うことができます。具体的には、同言語のリファレンスを参照して下さい。

《引数》

なし。

《戻り値》

なし。

《例外》

なし。

3-5-8. V31Control.Close() メソッド

通信デバイスを解放し、V31Controlをクローズします。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public void Close()
```

《解説》

- ・リーダーライターとの通信を行う内部スレッドを停止し、通信デバイスを閉じます。
- ・通信デバイスを閉じた後は、通信設定（ポート名、ボーレートなど）を変更することができます。
- ・通信デバイスに問題がなければ、再び開くことも可能です。
- ・既に閉じているV31Controlインスタンスに対して再度Close()メソッドを呼び出しても何ら問題はありません。

《引数》

なし。

《戻り値》

なし。

《例外》

なし。

3-5-9. V31Control.Open() メソッド

通信デバイスを取得し、V31Controlを開きます。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public void Open()
```

《解説》

- ・通信デバイスを開いて、リーダーライターとの通信を行うスレッドを内部的に起動します。
- ・通信デバイスを開いた後は、通信設定（ポート名、ボーレート）などの変更は行えません。これらの変更を行った場合、例外が送出されます。
- ・既に関しているV31Controlに対して、再度Open()メソッドを呼び出すと例外が送出されます。

《引数》

なし。

《戻り値》

なし。

《例外》

例 外	要 因
System.InvalidOperationException	既にオープン済みのRWControlに対して、再度オープンメソッドを呼び出した。
System.IO.IOException	通信デバイスの操作でエラーが発生した。

3-5-10. V31Control.Initialize() メソッド

イニシャルコマンド（CMD=10h）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] Initialize(byte iniMode, byte rspMode) ;  
public byte[] Initialize(ref V31Response res, byte iniMode, byte rspMode) ;
```

《解説》

- ・エラーのクリアー、モーターの動作確認、およびカード挿入待ちの解除などリーダーライターの初期化処理を行います。
- ・イニシャル処理が正常終了していない場合には、以後のリーダーライターの動作を保証できませんので、レスポンスを取得し、異常のないことを確認してから処理を行うよう、アプリケーションを作成して下さい。

《引数》

iniMode 動作モード指定

30h = イニシャル動作のみ

31h = イニシャル動作+カード排出（カードが搬送路にある場合、挿入口側に排出）

32h = リセット指定（ハードウェアリセット）

※リセット指定は、エラー発生時の復旧をより確実にを行うための指定です。

「RPS3=30h：コマンド実行終了」のレスポンス送信後、リーダーライターを強制リセットし電源投入直後と同じ状態にします（自動イニシャル動作より始めます）。

※リセット指定でイニシャル実行をした後は、2秒程、正常な応答が出来ません。

実行後、3秒間ほどはコマンド送信等を行わないようにして下さい。

rspMode レスポンスモード指定

30h = 互換1モード（不正処理、警告レスポンス無し）

31h = 互換2モード（不正処理レスポンス有り、警告レスポンス無し）

32h = V31モード（不正処理、警告レスポンス有り）

※電源投入時（自動イニシャル動作後、リセット指定後も含む）の初期値は

「互換1モード」です。

※V31モードでは、不正処理エラーが発生した場合、イニシャルコマンドを実行しないと他のコマンドを受け付けません。

※不正処理、警告レスポンスについては、「6-1. レスポンス・ステータス一覧表」を参照して下さい。

《戻り値》

受信パケットのバイトストリーム列

《例外》

例 外	要 因
System. ArgumentOutOfRangeException	引数の指定が誤っている。
System. InvalidateOperationExcption	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

3-5-11. V31Control.StatusRead() メソッド

ステータスリードコマンド（CMD＝20h）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] StatusRead() ;  
public byte[] StatusRead(ref V31Response res ) ;
```

《解説》

- ・リーダーライター内のカードの有無、およびカードの位置情報を含むレスポンスを取得します。
- ・カード処理が終了して、カードを排出した場合には、GetResponse() メソッドを用いるか、本メソッドを用いて、カードの挿入口からカードが取り去られることを監視して下さい。
- ・カードが残留していると次のカードの取り込みが実行不可となります。

《引数》

なし。

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System. InvalidateOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

3-5-12. V31Control.GetSensorStatus() メソッド

センサー情報取得コマンド（CMD＝21h）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] GetSensorStatus(ref bool sensor1, ref bool sensor2,  
                             ref bool sensor3, ref bool sensor4,  
                             ref bool sensor5 );  
  
public byte[] GetSensorStatus(ref V31Response res, ref bool sensor1,  
                             ref bool sensor2, ref bool sensor3,  
                             ref bool sensor4, ref bool sensor5 );
```

《解説》

- ・リーダーライター搬送路内のセンサー情報を取得します。
- ・各センサー情報は、遮光された状態がON、透過状態がOFFで返されます。
- ・搬送路センサーは、カード挿入口から奥に向かって順に、センサー１、２、３、４、５となります。

《引数》

sensor1	センサー１ (ON/OFF) 状態
sensor2	センサー２ (ON/OFF) 状態
sensor3	センサー３ (ON/OFF) 状態
sensor4	センサー４ (ON/OFF) 状態
sensor5	センサー５ (ON/OFF) 状態

※OFF状態 = false、ON状態 = true

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System.InvalidOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System.IO.IOException	通信デバイスの操作でエラーが発生した。
System.TimeoutException	タイムアウトが発生した。

3-5-13. V31Control.Read() メソッド

リードコマンド (CMD=33h) を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》


```
public byte[] Read(byte mode, byte check, byte track) ;  
public byte[] Read(ref V31Response res, byte mode, byte check, byte track) ;
```

《解説》


- ・カードの取り込み、カードの磁気情報の読み込みを行います。

《引数》


mode リードモード指定


- 30h = 標準 (69バイト) リード
- 31h = C R A - 2 2 0 0 互換リード
- 32h = カード取り込み
- 33h = 可変長 (1~69バイト) リード
- 34h = 逆可変長 (1~69バイト) リード
- 35h = ISO 様式リード 

※32h: カード取り込みを指定した場合、リードモードパラメーター以外の指定パラメータはチェックしません。また、磁気データに異常があってもリードエラーにはなりません。
何も書かれていないカードの取り込み時に使用します。

※制御機種が「ABS-V31UWI」の場合、32h = カード取り込み、35h = ISO 様式リード以外のリードモードは指定しないでください。 

check リードデータチェック指定




- 30h = パリティ有り (7ビット磁気リード)
磁気データの (偶数) パリティチェックを行います。パリティ有りを指定してライトした磁気データを読む場合に指定します。
- 31h = パリティ無し (8ビット磁気リード)
磁気データのパリティのチェックを行いません。パリティ無しを指定してライトした磁気データを読む場合に指定します。
- 32h = 奇数パリティ有り (ISO様式リード) 
磁気データの (奇数) パリティチェックを行います。ISO様式を指定してライトした磁気データを読む場合に指定します。


※制御機種が「ABS-V31UWI」の場合、32h = 奇数パリティ有り (ISO様式リード) 以外のリードデータチェックは指定しないでください。 

track トラック指定

- 30h = トラック 1 のリード指定
- 31h = トラック 2 のリード指定
- 32h = トラック 3 のリード指定
- 33h = トラック 1、2 のリード指定
- 34h = トラック 1、3 のリード指定
- 35h = トラック 2、3 のリード指定
- 36h = トラック 1、2、3 全てのリード指定


※リードモード指定の値により、リードチェック指定とトラック指定で利用できる値が変わります。

リードモード指定	リードデータチェック指定	トラック指定
3 0 h	3 0 h または 3 1 h	3 0 h ~ 3 6 h
3 1 h	3 0 h または 3 1 h	3 0 h
3 2 h	パラメーターチェックしない	パラメーターチェックしない
3 3 h	3 0 h または 3 1 h	3 0 h または 3 1 h
3 4 h	3 0 h	3 1 h
3 5 h 	3 2 h 	3 0 h ~ 3 6 h 

- ※リードモード指定：3 3 h (可変長) のトラック指定：3 1 h (トラック 2)、及び、
リードモード指定：3 4 h (逆可変長) は、「ABS-V 3 1」の Ver. 04. 00 ソフトウェアより対応しています。
- リードモード指定：3 5 h (ISO 様式リード) は、「ABS-V 3 1 UWI」の Ver. 03. 00 ソフトウェアより対応します。 

《戻り値》

- ・受信パケットのバイタストリーム列

※リードモード指定：3 5 h (ISO 様式リード) 時の磁気データ形式 

磁気データ					
トラック 1 磁気データ 0~76 バイト	分離 符号	トラック 2 磁気データ 0~37 バイト	分離 符号	トラック 3 磁気データ 0~104 バイト	分離 符号
①	FF	②	FF	③	FF

* : 状態変化
H E X

- ・ISO 様式リードモード (リードモード = 3 5 h) を指定した場合のレスポンスです。
- ・3 トラックの磁気データを、トラック 1 → 3 の順に返します。
- ・読み込めなかったトラックのデータについては、0 バイトで返します。
- ・磁気データには、ASCII コードでセットされます。
また、各トラックの対応文字、データについては、「6-4. ISO 様式 トラック 1 使用文字一覧」及び、「6-5. ISO 様式 トラック 2、3 使用文字一覧」を参照ください。
- ・指定トラックが 3 トラック未満の場合、指定トラック分のみのリードデータを返します。

例) トラック 2 のみリード指定時でリードした場合の磁気データ形式

磁気データ			
分離符号	トラック 2 磁気データ 1~37 バイト	分離符号	分離符号
FF	②	FF	FF

* : 状態変化
H E X

《例外》

例 外	要 因
System. ArgumentOutOfRangeException	引数の指定が誤っている。
System. InvalidateOperationExc eption	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行で きない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

3-5-14. V31Control.Cancel() メソッド

キャンセルコマンド（CMD=40h）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] Cancel() ;  
public byte[] Cancel(ref V31Response res ) ;
```

《解説》

- ・リードコマンド、クリーニングコマンド発行時の「カード挿入待ち」状態を解除します。

《引数》

なし。

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System.InvalidOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System.IO.IOException	通信デバイスの操作でエラーが発生した。
System.TimeoutException	タイムアウトが発生した。

3-5-15. V31Control.Write() メソッド

ライトコマンド (CMD=53h) を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] Write(byte mode, byte check, byte track, byte[] writeData) ;  
public byte[] Write(ref V31Response res, byte mode, byte check, byte track,  
                    byte[] writeData) ;
```

《解説》

- ・カードに磁気情報を書き込みます。

《引数》

mode ライトモード指定

- 30h = 標準 (69バイト) ライト
- 31h = C R A - 2 2 0 0 互換ライト
- 33h = 可変長 (1~69バイト) ライト
- 34h = 逆可変長 (1~69バイト) ライト

check パリティ指定

- 30h = パリティ有り (7ビット磁気ライト)
指定された磁気データにパリティを付加してカードに書き込みます。
データの有効ビット数は7ビットです。
- 31h = パリティ無し (8ビット磁気ライト)
磁気データに偶数パリティを付加しません。データの有効ビットは8ビットです。

track トラック指定

- 30h = トラック 1 のライト指定
- 31h = トラック 2 のライト指定
- 32h = トラック 3 のライト指定
- 33h = トラック 1、2 のライト指定
- 34h = トラック 1、3 のライト指定
- 35h = トラック 2、3 のライト指定
- 36h = トラック 1、2、3 全てのライト指定

※ライトモード指定の値により、パリティ指定とトラック指定で利用できる値が変わります。

ライトモード指定	パリティ指定	トラック指定
3 0 h	3 0 h または 3 1 h	3 0 h ~ 3 6 h
3 1 h	3 0 h または 3 1 h	3 0 h
3 3 h	3 0 h または 3 1 h	3 0 h または 3 1 h
3 4 h	3 0 h	3 1 h

※ライトモード指定：3 3 h (可変長) のトラック指定：3 1 h (トラック 2)、及び、
ライトモード指定：3 4 h (逆可変長) は、「ABS-V 3 1」の Ver. 04. 00 ソフトウェア
より対応しています。

《戻り値》

- ・受信パケットのバイトストリーム列

《例外》

例 外	要 因
System. ArgumentOutOfRangeException	引数の指定が誤っている。
System. ArgumentNullException	ライトデータが null である。
System. InvalidOperationException	V31Control が未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

3-5-16. V31Control.RegisterGaiji() メソッド

外字登録コマンド (CMD = 7 6 h) を実行します。

名前空間 SanwaNewtec. DeviceControls

アセンブリ SanwaNewtec. DeviceControls. V31Control. dll

《構文》

```
public byte[] RegisterGaiji(byte number, byte size, byte[] gaijiData) ;
public byte[] RegisterGaiji(ref V31Response res, byte number, byte size,
                             byte[] gaijiData) ;
```

《解説》

- ・リーダーライターに外字を登録します。
- ・登録できる外字は、24×24ドット、16×16ドット、12×24ドット、8×16ドットの4種類で合計 2 5 5 個まで登録できます。
与える外字データの大きさは、V31Control. GetNecessaryBytes() メソッドで算出した大きさで定義する必要があります。

《引数》

number 外字番号
01 ~ FFh

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System. ArgumentOutOfRangeException	引数の指定が誤っている。
System. ArgumentNullException	外字がnullである。
System. InvalidOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

3-5-17. V31Control.RegisterImage() メソッド

イメージデータ登録コマンド (CMD=77h) を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] RegisterImage(ushort address, ushort width, ushort height,  
                             byte[] imageData) ;  
public byte[] RegisterImage(ref V31Response res, ushort address, ushort width,  
                             ushort height, byte[] imageData) ;
```

《解説》

- ・リーダーライター内の指定したメモリアドレス上にイメージを登録します。
- ・登録アドレス、イメージ幅、イメージ高さの2バイトデータは、ビッグエンディアンで指定して下さい。

例：0 1 2 3 h 上位バイト 下位バイト

0 1	2 3
-----	-----

- ・イメージデータの大きさは、V31Control.GetNecessaryByte() で取得した値に揃える必要があります。
- ・イメージデータの転送は、V31Controlインスタンス内で、送信データを複数のパケットに分割して送信します。従って多量のデータを送る際には、Timeoutプロパティで設定するタイムアウト値を長めに再設定して下さい。

《引数》

address 登録アドレス

0000h ~ FFFEh

イメージデータを、リーダーライター内のメモリのどのアドレスから登録するか指定します。

登録アドレスは偶数となるように指定して下さい。

1つの登録データとして、最大320×576ドットのイメージデータが登録できます。

width イメージ幅

0001 ~ 0140h

印字するイメージデータのX軸幅（1～320ドット）を指定します。

height イメージ高さ

0001 ~ 0240h

印字するイメージデータのY軸幅（1～576ドット）を指定します。

imageData イメージデータ

イメージデータは、外字フォントと同様に、左上から、ライン順にデータを送信して下さい（データ格納はバイト単位）。

7 ... 0	7 ... 0	...
データ 1	データ 2	...
イメージデータ		

※複数のイメージを登録する場合は、登録する領域が重複しないよう管理して下さい。

例えば、320×576ドットのイメージデータをアドレス0000hから登録した場合は、メモリを23, 040バイト（320÷8×576）使用します。

この場合、「0000～59FFh」の領域が使用されていますので、次に登録するイメージの登録アドレスは、「5A00h」以降として下さい。

※リーダーライターのメモリ範囲（アドレス：0000～FFFFh）を越えないように指定して下さい。

※このコマンドで登録されたデータは、電源OFFまたはイニシャル後も保持されます。

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System. ArgumentOutOfRangeException	引数の指定が誤っている。
System. ArgumentNullException	イメージデータがnullである。
System. InvalidOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

3-5-18. V31Control.PrintSetting() メソッド

印字設定コマンド（CMD＝78h）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

[半角フォント指定あり]

```
public byte[] PrintSetting(byte card, byte mode, byte align, byte font) ;  
public byte[] PrintSetting(ref V31Response res, byte card, byte mode, byte align,  
                           byte font) ;
```

[半角フォント指定なし]

```
public byte[] PrintSetting(byte card, byte mode, byte align) ;  
public byte[] PrintSetting(ref V31Response res, byte card, byte mode,  
                           byte align) ;
```

《解説》

- ・各種印字コマンドの印字条件を設定します。
- ・半角フォント指定を行わないメソッド呼び出しが行われた場合、リーダーライターの標準フォントが使われます。

《引数》

card カード種類

データ	選択カード			
30h	白濁リライト	リコー	FB-661M	発色：白
32h	ロイコリライト	三菱製紙	TRCGAACS/CH	発色：青
33h	90℃サーマル	水三島紙工	SNT-231-N1-T2	発色：黒
34h	110℃サーマル	水三島紙工	SNT-220-T3	発色：黒
37h	ロイコリライト	三菱製紙	TRCG99SS/SH	発色：青
38h	ロイコリライト	リコー	CR-631FB	発色：黒
39h	紙カード	サトー		発色：黒
※1 ㊦				
3Ah	REDカード	小林記録紙	KSNB-5360/5376	発色：白
※1 ㊦				
3Bh	ロイコリライト	三菱製紙	TRCGB3BS	発色：黒

※サーマル、紙カードを指定した場合、印字方法の指定に依らず重ね書き印字となります。

※白濁リライトカードのクーダス03（35h）、クーダス04（36h）については、他機種からのカード切り替え運用のみ対応可能です。通常運用はできません。もし通常運用された場合は、カード寿命が著しく低下して、最悪100回前後の処理回数で白濁化が始まり印字品質が低下します。

※1：対象機種がABS-L31Uの場合、「RPS3=32h コマンド実行不可」となります。㊦

mode 印字方法

30h = 消去＋印字

消去バーで以前に書かれていた文字を消去してから、新しい印字を行います。
リライトカード使用時のみ有効です。（デフォルト）

31h = 重ね書き印字

印字のみを行います。

align 印字座標

30h = 標準座標 (デフォルト)

33h = 左回転座標

34h = 右回転座標

※回転座標指定時に消去を行った場合は、自動的に全面消去となります。

font 印字する半角フォント

30h = 標準フォント (デフォルト)

31h = 軸太フォント

※このコマンドで設定された印字条件は、電源OFFまたはイニシャル後も保持されます。

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System. ArgumentOutOfRangeException	引数の指定が誤っている。
System. InvalidateOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

3-5-19. V31Control.PrintImage() メソッド

イメージデータ印字コマンド（CMD=7Bh）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

[登録アドレス指定]

```
public byte[] PrintImage(byte mode, byte clear, ushort width, ushort height,  
                          ushort x, ushort y, ushort address) ;
```

```
public byte[] PrintImage(ref V31Response res, byte mode, byte clear, ushort width,  
                          ushort height, ushort x, ushort y, ushort address) ;
```

[送信イメージデータ指定]

```
public byte[] PrintImage(byte mode, byte clear, ushort width, ushort height,  
                          ushort x, ushort y, byte[] imageData) ;
```

```
public byte[] PrintImage(ref V31Response res, byte mode, byte clear, ushort width,  
                          ushort height, ushort x, ushort y, byte[] imageData) ;
```

《解説》

- ・ 指定された座標にイメージデータを印字します。
- ・ イメージデータの印字には、予めリーダーライターに登録されているイメージデータ呼び出して印字する方法と、印字するイメージデータをコマンド実行の度に送信して印字する2つのパターンがあります。必要に応じてアプリケーションで最適と思われる方法と選択してください。
- ・ 印字範囲は、「6-7. 印字範囲2」を参照して下さい。
- ・ 回転座標指定時にも、イメージデータは回転しません（標準座標で印字）。
- ・ 印字バッファークリアーをせずに、前回の展開範囲と重なる範囲に展開した場合、重なった範囲の印字データが重なって展開されます。
- ・ X軸イメージ幅、Y軸イメージ幅、X軸印字位置、Y軸印字位置の2バイトデータは、ビッグエンディアンで指定して下さい。

例：0 1 2 3 h 上位バイト 下位バイト

0 1	2 3
-----	-----

《引数》

mode 実行モード指定

30h = 展開+印字

イメージデータを印字バッファに展開後、印字を行う。

31h = 展開

イメージデータを印字バッファに展開する（印字は行わない）。

32h = 展開+印字+排出

イメージデータを印字バッファに展開後、印字を行い、カード排出を行う。

33h = 登録データの展開+印字

登録されたイメージデータを印字バッファに展開後、印字を行う。

34h = 登録データの展開

登録されたイメージデータを印字バッファに展開する（印字は行わない）。

35h = 登録データの展開+印字+排出

登録されたイメージデータを印字バッファに展開後、印字を行い、カード排出を行う。

注）印字を行う際には、印字バッファに展開されているデータ（前回バッファークリアー後に展開されたデータ）全てを印字します。

clear 印字バッファークリアー指定

30h = 印字バッファークリアー（実行モード指定の印字データ展開前にクリアー）。

31h = 印字バッファークリアーしない。

width X軸イメージ幅

0001 ~ 0140h

印字するイメージデータのX軸幅（1～320ドット）を指定します。

height Y軸イメージ幅

0001 ~ 0240h

印字するイメージデータのY軸幅（1～576ドット）を指定します。

x X軸印字位置

0000 ~ 013Fh

イメージデータのX軸印字開始位置（0～319ドット）を指定します。

y Y軸印字位置

0000 ~ 023Fh

イメージデータのY軸印字開始位置（0～575ドット）を指定します。

address 登録アドレス

0000h ~ FFFEh

イメージ登録コマンドで指定した登録アドレスを指定します。

imageData イメージデータ

イメージデータは、外字フォントと同様に、左上から、ライン順にデータを送信して下さい（データ格納はバイト単位）。

7 ... 0	7 ... 0	...
データ 1	データ 2	...
イメージデータ		

※印字可能範囲を越えないように指定して下さい。

X軸イメージ幅 + X軸印字位置 ≤ 320

Y軸イメージ幅 + Y軸印字位置 ≤ 576

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System. ArgumentOutOfRangeException	引数の指定が誤っている。
System. ArgumentNullException	送信イメージデータ指定呼び出しで、イメージデータがnullである。
System. InvalidateOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

3-5-20. V31Control.PrintString() メソッド

文字印字コマンド（CMD＝7Ch）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] PrintString(byte mode, byte clear, byte line, byte[] strData) ;  
public byte[] PrintString(ref V31Response res, byte mode, byte clear, byte line,  
                           byte[] strData) ;
```

《解説》

- ・ 行指定モードと任意座標指定モードを選択して印字を行います。
- ・ 任意座標指定モードでは、指定された任意の印字開始位置から印字します。
- ・ 印字位置指定がない場合には、座標(000, 000)から展開します。
- ・ 印字範囲は、行指定モードの場合は「6-6. 印字範囲1」、任意座標モードの場合は「6-7. 印字範囲2」を参照して下さい。
- ・ 印字バッファークリアーをせずに、前回の展開範囲と重なる範囲に展開した場合、その範囲の印字データは重なったまま展開されます。

《引数》

mode 実行モード指定

30h = 展開＋印字

文字データを印字バッファーに展開後、印字を行う。

31h = 展開

文字データを印字バッファーに展開する（印字は行わない）。

32h = 展開＋印字＋排出

文字データを印字バッファーに展開後、印字を行い、カード排出を行う。

注）印字を行う際には、印字バッファーに展開されているデータ（前回バッファークリアー後に展開されたデータ）全てを印字します。

clear 印字バッファークリアー指定

30h = 印字バッファーをクリアー（実行モード指定の印字データ展開前にクリアー）。

31h = 印字バッファーをクリアーしない。

line 展開開始行

印字バッファー展開開始行指定（行単位の消去は出来ません）

00h = 任意座標モード

※展開する座標を1ドット単位で任意に指定します。

01h ～ = 行指定モード

※行指定に従って展開する座標（行）を指定します。

01 ～ 14h = 標準座標 （1～20行）

01 ～ 0dh = 左回転座標 （1～13行）

01 ～ 0dh = 右回転座標 （1～13行）

strData 文字データ

ANK半角文字（「6-3. 印字ANK文字コード表」参照）

全角文字（シフトJIS漢字コード JIS X 0208準拠）

※第一・第二水準以外の漢字コードを指定した場合の印字は保証できません。

〈制御コード〉

0Dh : 改行コード

11h : 縦倍角コード

14h : 文字サイズ解除コード（縦1倍・横1倍に設定します）

1Bh+73h+yy+xx : 文字サイズ設定コード

yy=31h（縦1倍）～34h（縦4倍）

xx=31h（横1倍）～34h（横4倍）

1Bh+48h : 16ドットフォントに切替

1Bh+4Eh : 24ドットフォントに切替（デフォルト）

1Bh+42h : 太字フォントに切替

1Bh+53h : 標準フォントに切替

1Bh+70h+‘yyy’+‘xxx’ : 印字開始位置指定

yyy（縦方向）、xxx（横方向）

‘yyy’=‘000’～‘575’ドット

回転座標時‘000’～‘319’ドット

‘xxx’=‘000’～‘319’ドット

回転座標時‘000’～‘575’ドット

1Bh+67h+外字No. : 外字No.で指定した外字登録フォントを印字

※印字データが印字範囲を超えた場合、超えた分は無効となります。

※縦倍角、文字サイズ設定、16ドット、太字の指定は、改行、または印字開始位置指定を行うと解除されます。必要な場合は再度指定して下さい。

※任意座標モードで改行コードを使用すると、前回の印字開始位置指定の位置から改行された場所に移動します（行間5ドット）。

※登録していない外字を印字データとして指定した場合の印字は、保証できません。

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System.ArgumentOutOfRangeException	引数の指定が誤っている。
System.ArgumentNullException	印字データがnullである。
System.InvalidOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System.IO.IOException	通信デバイスの操作でエラーが発生した。
System.TimeoutException	タイムアウトが発生した。

○任意座標モードでの印字指定例

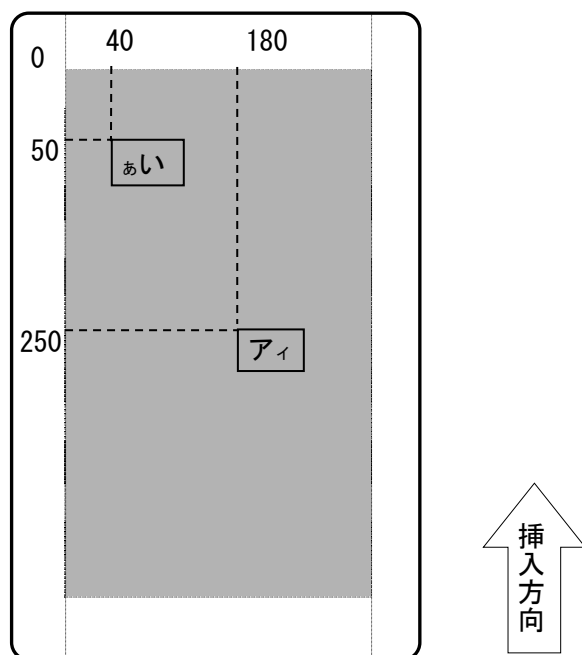
Y座標 50、X座標 40に16ドットフォントの「あ」と24ドットフォントの「い」、
Y座標250、X座標180に24ドットフォントの「ア」と16ドットフォントの「イ」
を印字する場合

データ指定例

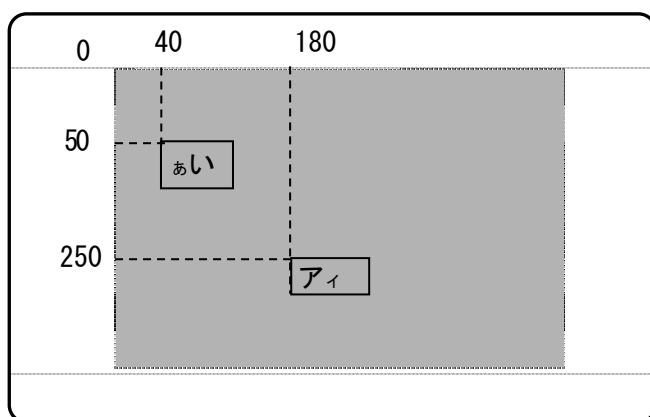
「1Bh+70h+‘050’+‘040’+1Bh+48h+‘あ’+1Bh+4Eh+
‘い’+1Bh+70h+‘250’+‘180’+‘ア’+1Bh+48h+‘イ’」

※指定した座標を原点として、下基準（最大文字の底辺に文字を合わせる）で印字展開
します。

・標準座標時



・回転座標時



3-5-21. V31Control.PrintBarcode() メソッド

バーコード印字コマンド (CMD = 7 E h) を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] PrintBarcode(byte mode, byte clear, byte kind, byte tip, byte width,
                           byte height, byte x, byte y, byte[] barcodeData) ;
public byte[] PrintBarcode(ref V31Response res, byte mode, byte clear, byte kind,
                           byte tip, byte width, byte height, byte x, byte y,
                           byte[] barcodeData) ;
```

《解説》

- ・バーコードの印字を行います。
- ・印字範囲外への印字指定や印字範囲を超える場合は印字されません。
- ・回転座標指定時にも、バーコード印字は回転しません（標準座標で印字）。
- ・前回の展開範囲と重なる範囲に展開した場合、重なった範囲のデータを消去して展開を行います。

※ロイコ及び白濁リライトカードへのバーコード印字は保証出来ません。また、下地を可能な限り白くする、余白を十分に取る等の工夫を行って下さい。

《引数》

mode 実行モード指定

30h = 展開 + 印字

バーコードデータを印字バッファに展開後、印字を行う。

31h = 展開

バーコードデータを印字バッファに展開する（印字は行わない）。

32h = 展開 + 印字 + 排出

バーコードデータを印字バッファに展開後、印字を行い、カード排出を行う。

注）印字を行う際には、印字バッファに展開されているデータ（前回バッファークリア後に展開されたデータ）全てを印字します。

clear 印字バッファークリアー指定

30h = 印字バッファークリアーする（実行モード指定の印字データ展開前にクリアー）。

31h = 印字バッファークリアーしない。

kind バーコード種類指定

30h = J A N 標準（データサイズ：12バイト）

31h = J A N 短縮（データサイズ：7バイト）

※J A N 標準、J A N 短縮は、リーダーライターがチェックデジットを付加します。

32h = N W - 7（データサイズ：可変）

※印字範囲を超えるバーコードデータは無視されます。

33h = C O D E 3 9（データサイズ：可変）

※印字範囲を超えるバーコードデータは無視されます。

tip バーコード下の解説文字の有無

30h = 有り

31h = 無し

注) NW-7において、「a, b, c, d, e, n, t」のように小文字のキャラクターを指定した場合は、それぞれの大文字を印字します。また、「\$」を指定した場合は、「¥」を印字します。

width バーコード幅指定

30h = 1モジュール寸法が2ドット (0. 250mm)

31h = 1モジュール寸法が3ドット (0. 375mm)

width 値		30h	31h
バーコード種類			
JAN標準: 1モジュール寸法		0. 250mm	0. 375mm
JAN短縮: 1モジュール寸法		0. 250mm	0. 375mm
NW-7 CODE 39	ナローバー寸法	0. 250mm	0. 375mm
	ワイドバー寸法	0. 625mm	0. 750mm

height バーコード高さ指定

01 ~ 30h

印字するバーコードの高さを指定 (1mm~48mm: 1mm/dev)

注) 但しバーコードの上下には、上記指定の高さ以外に、上1. 5mm、下0. 5mmのマージンを自動的に取ります (解説文字有りの時は、下0. 5mmのマージンからさらに1. 5mmの解説文字を付加します)。

x 印字X座標

印字するバーコードの座標を指定 (1mm/dev)

バーコード種類		指定できる印字X座標の範囲
JAN標準	1モジュール=2dot	00h~0Dh
	1モジュール=3dot	00h (固定)
JAN短縮	1モジュール=2dot	00h~13h
	1モジュール=3dot	00h~09h
NW-7	1モジュール=2dot	00h~1Eh
	1モジュール=3dot	00h~1Ah
CODE 39	1モジュール=2dot	00h~1Dh
	1モジュール=3dot	00h~18h

y 印字Y座標

00 ~ 40h

印字するバーコードのY座標を指定 (1mm/dev)

barcodeData バーコードデータ

バーコードデータをASCIIコードで指定してください。

NW-7、CODE39はスタート/ストップキャラクターも指定する必要があります。

(下記の最大有効桁数には、スタート/ストップキャラクターを含む)

バーコード種類		最大有効桁数 (X座標=00h)
JAN標準	1モジュール=2dot	12桁 (固定) チェックデジット付加
	1モジュール=3dot	12桁 (固定) チェックデジット付加
JAN短縮	1モジュール=2dot	7桁 (固定) チェックデジット付加
	1モジュール=3dot	7桁 (固定) チェックデジット付加
NW-7	1モジュール=2dot	14桁
	1モジュール=3dot	9桁
CODE39	1モジュール=2dot	11桁
	1モジュール=3dot	7桁

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System. ArgumentOutOfRangeException	引数の指定が誤っている。
System. ArgumentNullException	バーコードデータがnullである。
System. InvalidOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

3-5-22. V31Control.Eject() メソッド

カード排出コマンド（CMD＝80h）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] Eject() ;  
public byte[] Eject(ref V31Response ) ;
```

《解説》

- ・リーダーライター内のカードの排出を行います。
- ・装置内にカードがあればカードを挿入口側に排出し、なければモーターを排出方向に約 0.2 秒間回します（挿入口にカードがある場合も排出動作を実行し、「RPS1＝34h：挿入口カード残留」レスポンスを返します）。
- ・カード排出後は「RPS1＝34h：挿入口カード残留」レスポンスを返します。ステータスリードコマンドまたは GetResponse メソッドの再送を行い「RPS1＝30h：装置内カードなし」を確認して排出コマンドの終了として下さい。「RPS1＝34h：挿入口カード残留」の状態では、次のカード取り込みが「RPS3＝32h：コマンド実行不可」となります。

《引数》

なし。

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System.InvalidOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System.IO.IOException	通信デバイスの操作でエラーが発生した。
System.TimeoutException	タイムアウトが発生した。

3-5-23. V31Control.Cleaning() メソッド

クリーニングコマンド（CMD=A0h）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] Cleaning() ;  
public byte[] Cleaning(ref V31Response ) ;
```

《解説》

- ・リーダーライターにクリーニング動作を要求します。
クリーニングコマンドを発行すると、リーダーライターは「カード挿入待ち」状態になります。クリーニングカードを挿入すると、磁気ヘッド、サーマルヘッド、消去ヘッドのクリーニングを行います。
- ・クリーニング終了後はカードを排出します。排出動作はカード排出コマンドの動作に準じます。
- ・装置内にカードがある場合は、そのカードでクリーニング動作を行います。
- ・「カード挿入待ち」の解除は、キャンセルコマンドで行うことができます。

《引数》

なし。

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System. InvalidateOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

3-5-24. V31Control.SetClock() メソッド

時計設定コマンド（CMD=E 1 h）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] SetClock(System.DateTime dt) ;  
public byte[] SetClock(ref V31Response, System.DateTime dt) ;
```

《解説》

- ・リーダーライターの内蔵時計の時刻を設定します。

《引数》

dt 設定したい日時

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System.InvalidOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System.IO.IOException	通信デバイスの操作でエラーが発生した。
System.TimeoutException	タイムアウトが発生した。

3-5-25. V31Control.GetVersion() メソッド

バージョン取得コマンド（CMD=F0h）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] GetVersion() ;
public byte[] GetVersion(ref V31Response res ) ;
```

《解説》

- ・リーダーライターのソフトウェアバージョンを取得します。

《引数》

なし。

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System.InvalidOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System.IO.IOException	通信デバイスの操作でエラーが発生した。
System.TimeoutException	タイムアウトが発生した。

※取得バージョン情報のフォーマットは、以下の通りとなります。

バイト	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
名称	ヘッダー	アプリケーション	アプリ	アプリ種別	ソフト管理No.				区切り文字	バージョン				区切り文字	ヘッダー		ファーム	ファーム種別
HEX	41	50	3A	56	*	*	*	*	2D	*	*	*	*	2C	4F	53	3A	56
ASCII	A	P	:	V					-					,	0	S	:	V

19	20	21	22	23	24	25	26	27	28	29	30	31	32
ソフト管理No.				区切り文字	バージョン				区切り文字	予備			
*	*	*	*	2D	*	*	*	*	2C	20	20	20	20
				-					,				

3-5-26. V31Control.GetClock() メソッド

現在時刻取得コマンド（CMD=F 1 h）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] GetClock(ref System.DateTime dt) ;  
public byte[] GetClock(ref V31Response res, ref System.DateTime dt) ;
```

《解説》

- ・リーダーライターの内蔵時計の設定値を読み出します。

《引数》

dt リーダーライターの内蔵時計の設定値

《戻り値》

受信パケットのバイタストリーム列

《例外》

例 外	要 因
System.InvalidOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System.IO.IOException	通信デバイスの操作でエラーが発生した。
System.TimeoutException	タイムアウトが発生した。
System.FormatException	リーダーライターからのレスポンスの書式が誤っている。

3-5-27. V31Control.GetResponse() メソッド

レスポンス再送要求（ENQ送信）を実行します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] GetResponse() ;  
public byte[] GetResponse(ref V31Response res ) ;
```

《解説》

- ・リーダーライターにENQを送信し、そのレスポンスを取得します。
- ・コマンド送信からENQを連続送信する場合、送信間隔として100ms以上確保してください。

《引数》

なし。

《戻り値》

受信パケットのバイトストリーム列。

《例外》

例 外	要 因
System. InvalidateOperationException	V31Controlが未オープン。
	リーダーライターと通信中のため、本メソッドが実行できない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

3-5-28. V31Control.SendCommand() メソッド

任意のコマンド列を送信します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

[コマンドのみ]

```
public byte[] SendCommand(byte cmd ) ;
```

```
public byte[] GetResponse(ref V31Response res, byte cmd ) ;
```

[付加データあり]

```
public byte[] SendCommand(byte cmd, byte[] data )
```

```
public byte[] GetResponse(ref V31Response res, byte cmd, byte[] data ) ;
```

《解説》

- ・リーダーライターに任意のコマンドを送信します。また、付加データを指定することも可能です。
- ・付加データを指定する場合、データ長は、1～249バイトの範囲で指定してください。この範囲外で指定すると、例外を送出します。

《引数》

cmd コマンド

data 付加データ

《戻り値》

受信パケットのバイトストリーム列

《例外》

例 外	要 因
System.ArgumentOutOfRangeException	引数の指定が誤っている。
System.ArgumentNullException	付加データありの場合に、付加データがnullである。
System.InvalidOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System.IO.IOException	通信デバイスの操作でエラーが発生した。
System.TimeoutException	タイムアウトが発生した。

本メソッドは、V31Controlがサポートしていない任意のコマンドを送信することができ、リーダーライター製品の機能強化にV31Controlのバージョンアップが遅れた場合、あるいは、過去のリーダーライター製品に対して、V31Controlがサポートしていないコマンドを送信するために実装されています。本メソッドを利用してアプリケーションを開発する場合には、対象となるリーダーライターの製品仕様書を熟読の上、慎重に取り扱われますようお願いいたします。

3-5-29. V31Control.WriteForISO() メソッド

ライトコマンド (CMD=53h) を実行します。

※ISO様式ライト専用メソッドとなります。

※本メソッドは、「ABS-V31UWI」のVer.03.00ソフトウェアより対応します。

名前空間 SanwaNewtec.DeviceControls

アセンブリ SanwaNewtec.DeviceControls.V31Control.dll

《構文》

```
public byte[] WriteForISO( byte track,
                           ushort writeDataLen1, byte[] writeData1,
                           ushort writeDataLen2, byte[] writeData2,
                           ushort writeDataLen3, byte[] writeData3 );
public byte[] WriteForISO( ref V31Response res, byte track,
                           ushort writeDataLen1, byte[] writeData1,
                           ushort writeDataLen2, byte[] writeData2,
                           ushort writeDataLen3, byte[] writeData3 );
```

《解説》

- ・カードに磁気情報を書き込みます。

《引数》

track トラック指定

- 30h = トラック 1 のライト指定
- 31h = トラック 2 のライト指定
- 32h = トラック 3 のライト指定
- 33h = トラック 1、2 のライト指定
- 34h = トラック 1、3 のライト指定
- 35h = トラック 2、3 のライト指定
- 36h = トラック 1、2、3 全てのライト指定

writeDataLen1～writeDataLen3指定

- writeDataLen1 : トラック 1 磁気データ長 (0～76バイト)
- writeDataLen2 : トラック 2 磁気データ長 (0～37バイト)
- writeDataLen3 : トラック 3 磁気データ長 (0～104バイト)

※指定トラック以外のトラック磁気データ長は、0をセットしてください。

writeData1～writeData3指定

writeData1 : トラック 1 磁気データ

writeData2 : トラック 2 磁気データ

writeData3 : トラック 3 磁気データ

※トラック 1→3 の順にデータを指定します。

※磁気データはそれぞれのトラックによりデータ長が異なります。規定を超えるデータ数を送信した場合は、書き込みデータを保証できません。

※磁気データはASCIIコードで指定してください。トラックにより使用可能な文字が限定されますのでご注意ください。各トラックの対応文字、データについては、「6-4. ISO 様式 トラック 1 使用文字一覧」及び、「6-5. ISO 様式 トラック 2、3 使用文字一覧」を参照ください。

※トラック指定 (track パラメータ) に応じて、該当するトラックの磁気データを指定してください。

※トラック 1～3 に使用可能な文字の範囲チェックは行いません。

《戻り値》

- ・受信パケットのバイトストリーム列

《例外》

例 外	要 因
System. ArgumentOutOfRangeException	引数の指定が誤っている。
System. ArgumentNullException	ライトデータがnullである。
System. InvalidateOperationException	V31Controlが未オープン。 リーダーライターと通信中のため、本メソッドが実行できない。
System. IO. IOException	通信デバイスの操作でエラーが発生した。
System. TimeoutException	タイムアウトが発生した。

例) トラック 2 のライト指定、トラック 2 磁気データが 7 バイトの場合

```
byte[] ret = null;
V31Control rwControl = new V31USBControl();
V31Response res = new V31Response();
byte[] wData1 = new byte[ 76 ];
byte[] wData2 = new byte[ 7 ]{ 0x31, 0x32, 0x33, 0x34, 0x35, 0x37 };
byte[] wData3 = new byte[ 104 ];

ret = rwControl.WriteForISO( ref res,
                             (byte)0x31,
                             (ushort)0,
                             wData1,
                             (ushort)7,
                             wData2,
                             (ushort)0,
                             wData3 );
```

4. USB ドライバーのインストール方法

USB通信を用いてリーダーライターの制御を行う場合、アプリケーションを動作させるコンピュータに弊社提供USBドライバーをインストールする必要があります。

USBドライバーは、添付のサンプルソースを納めたパッケージ内に格納されています。以下の手順に従い、インストールを行って下さい。

また、添付のUSBドライバーは、動作確認済みのOS上での動作のみを保証しますので、他のOSが動作するコンピュータに添付のUSBドライバーをインストールすることは避けて下さい。

旧バージョンのUSBドライバー(sntv31usb.sys)を使用していた場合は、sntv31usb.sysをアンインストールしてから、本クラスライブラリに添付のUSBドライバーをインストールしてください。

※各 Windows OS の操作については、各パソコンメーカーやマイクロソフト社にお問い合わせください。△

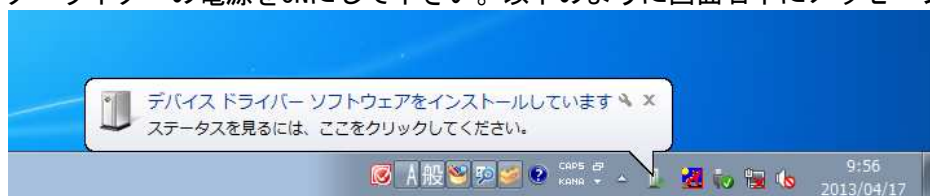
4-1. 共通事項

リーダーライターの電源が切れていることを確認して、コンピュータのUSBポートに接続します。お使いのコンピュータ上で、他のアプリケーションが動作していないことを確認して下さい。動作中のアプリケーションがある場合、動作中のアプリケーションを終了して下さい。

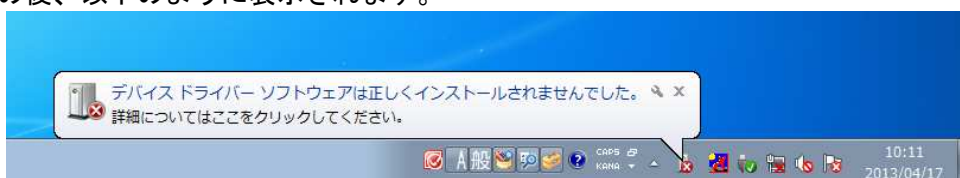
4-2. Windows7 の場合

1) リーダーライターの電源を ON する。

リーダーライターの電源をONにして下さい。以下のように画面右下にメッセージが表示されます。



その後、以下のように表示されます。



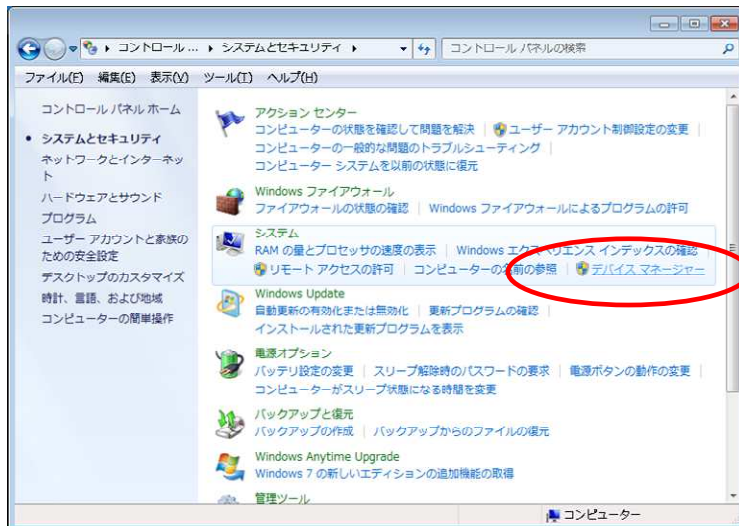
ここではインストールされません。

2) デバイスマネージャーを開いて USB ドライバーをインストールする。

①コントロールパネルを開いて、「システムとセキュリティ」をクリックします。

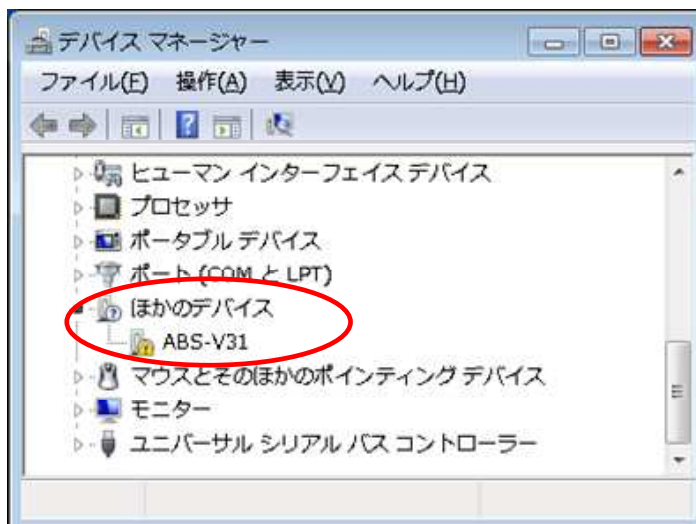


② 「デバイスマネージャー」をクリックします。

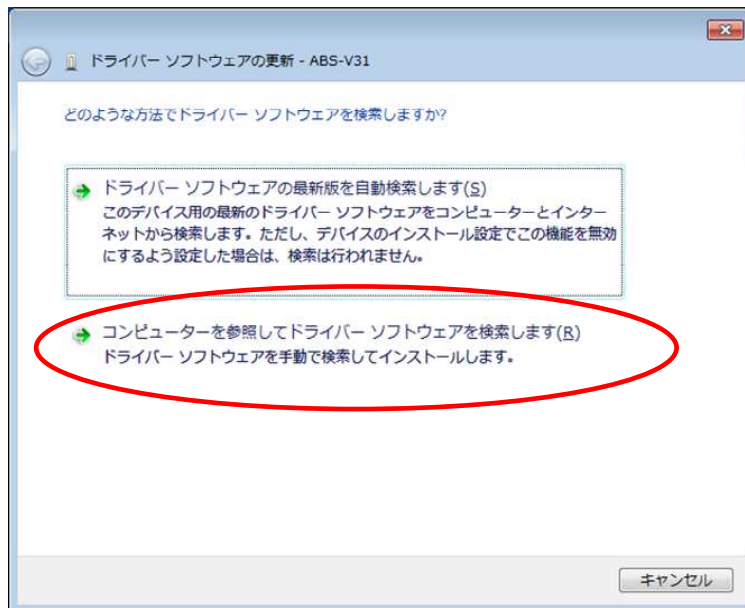


③ 「ほかのデバイス」に「ABS-V31」と表示されます。(ABS-L31U 接続時も「ABS-V31」と表示されます。△)

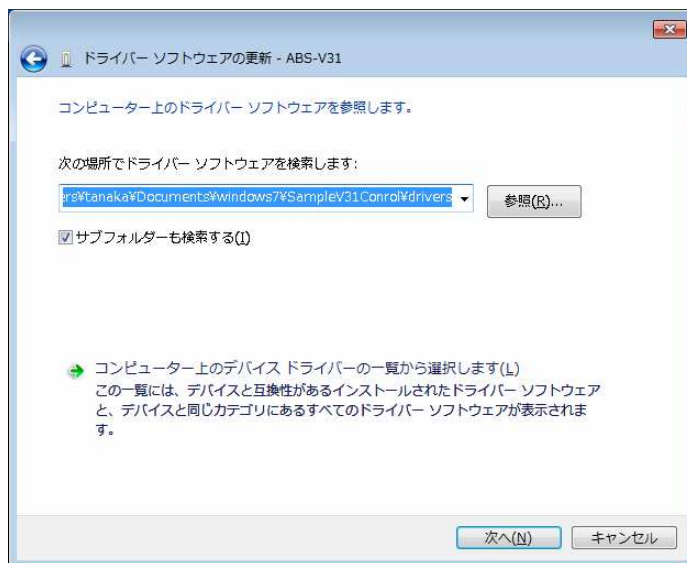
「ABS-V31」を右クリックして、「ドライバーソフトウェアの更新」をクリックします。



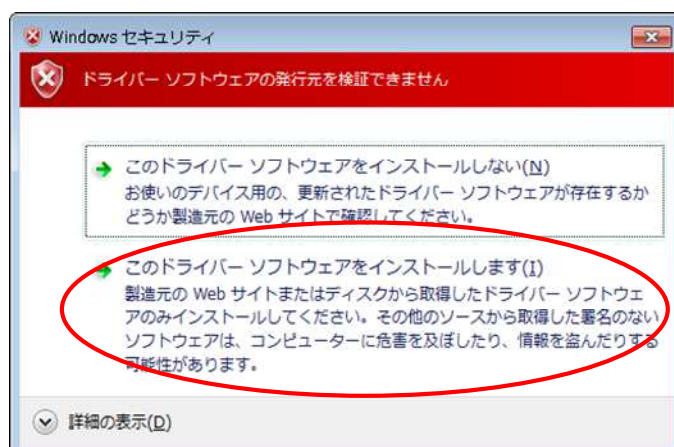
- ④「コンピューターを参照してドライバーソフトウェアを検索します」をクリックします。



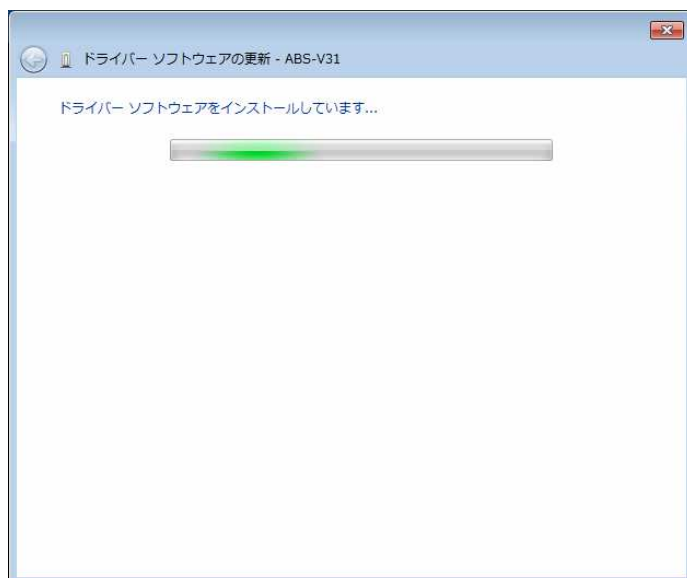
- ⑤USB ドライバーの inf ファイルを指定します。
inf ファイルは、展開したフォルダーの「SampleV31Control¥driver」にあります。
フォルダーを選択したら、「次へ」 ボタンをクリックします。



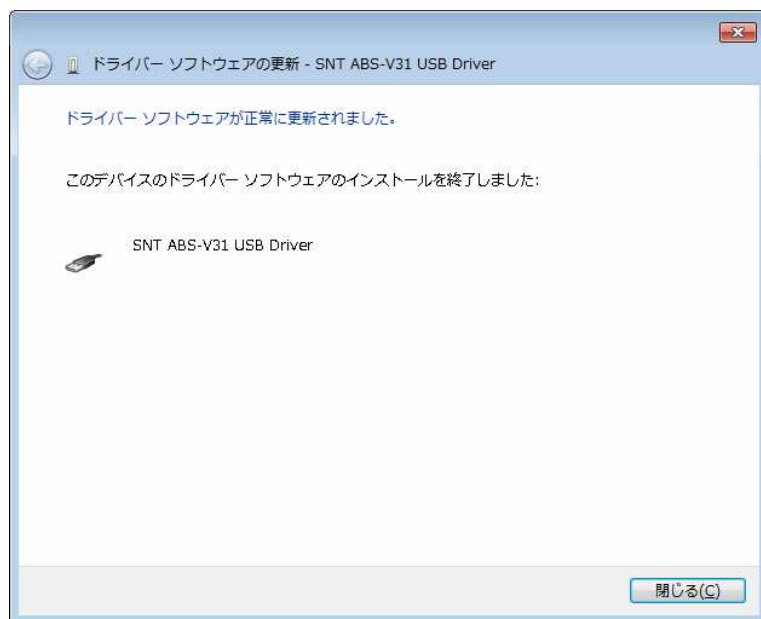
- ⑥「このドライバーソフトウェアをインストールします」をクリックします。



⑦インストールを開始します。



⑧インストールが完了すると下記の画面となります。
「閉じる」ボタンをクリックします。



以上で、USB ドライバーのインストールは完了となります。

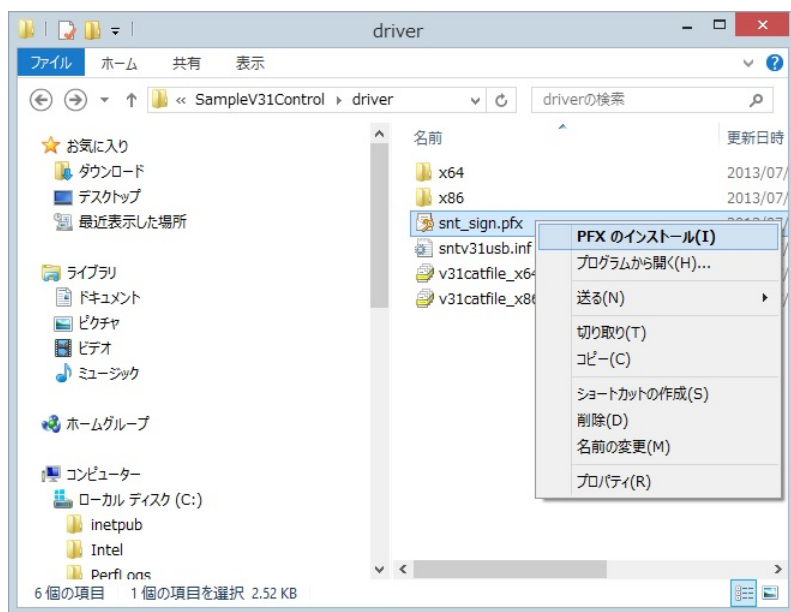
4-3. Windows8/8.1/10/11 の場合

デジタル証明と USB ドライバーのインストールが必要です。

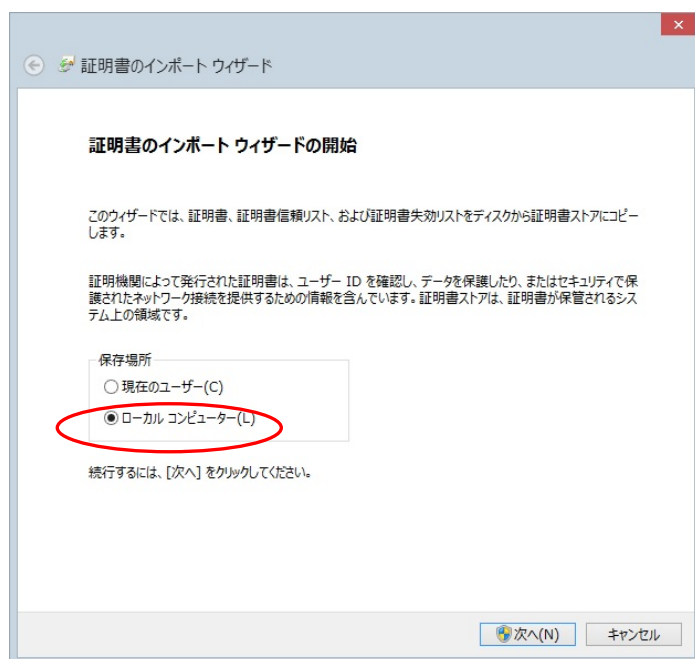
4-3-1. デジタル証明書のインストール

USB ドライバーインストール用デジタル証明書をインストールする。

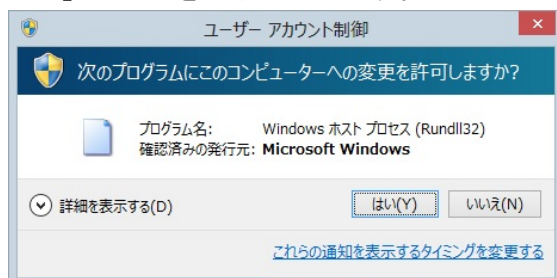
①snt_sign.pfx ファイルを右クリックして、「PFX のインストール」をクリックします。



②「ローカルコンピューター」をチェックして、「次へ」ボタンをクリックします。

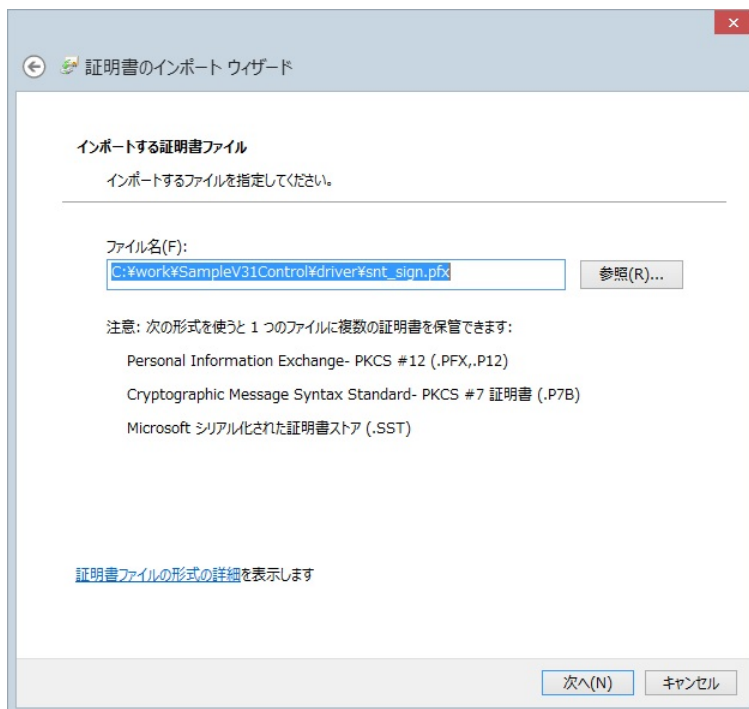


③「はい」ボタンをクリックします。

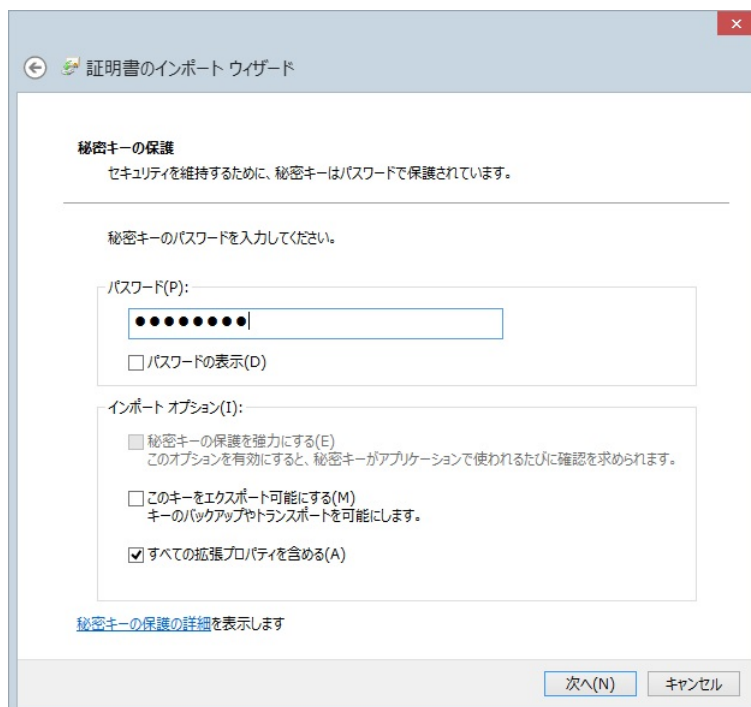


④snt_sign.pfx ファイルを指定します。

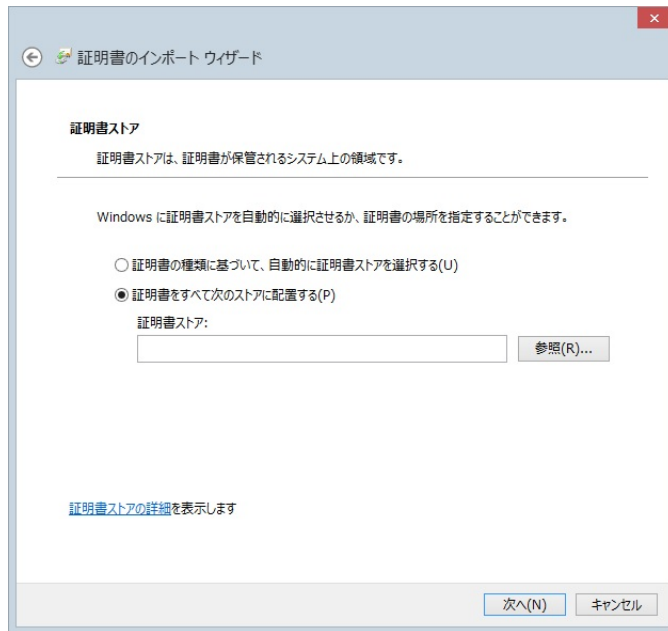
snt_sign.pfx ファイルは、展開したフォルダーの「SampleV31Control¥driver」にあります。
フォルダーを選択したら、「次へ」ボタンをクリックします。



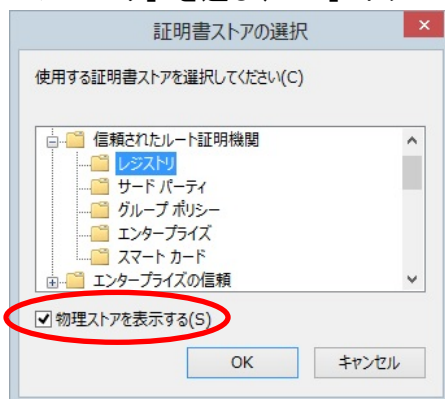
⑤パスワードに、「Z5rjxt64」と入力して、「次へ」ボタンをクリックします。



- ⑥ 「証明書をすべて次のストアに配置する」をチェックして、「参照」ボタンをクリックします。



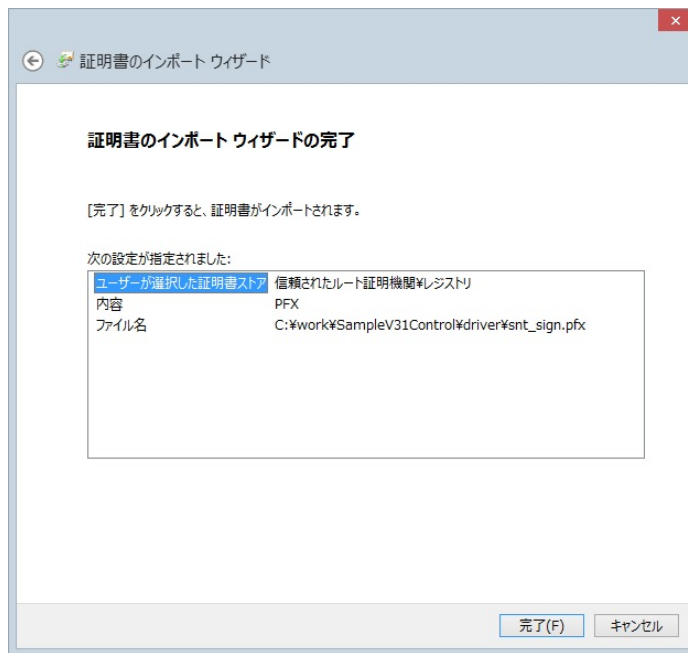
- ⑦ 「物理ストアを表示する」をチェックして、「信頼されたルート証明機関」の下にある「レジストリ」を選び、「OK」ボタンをクリックします。



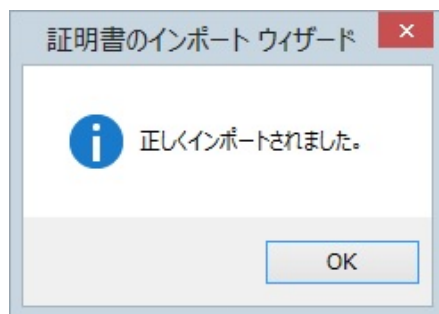
- ⑧ 「次へ」ボタンをクリックします。



⑨ 「完了」 ボタンをクリックします。



⑩ 「OK」 ボタンをクリックします。



以上で、USB ドライバーインストール用デジタル証明書のインストールは完了となります。

4-3-2. USB ドライバーのインストール

デバイスマネージャーを開いて USB ドライバーをインストールする。

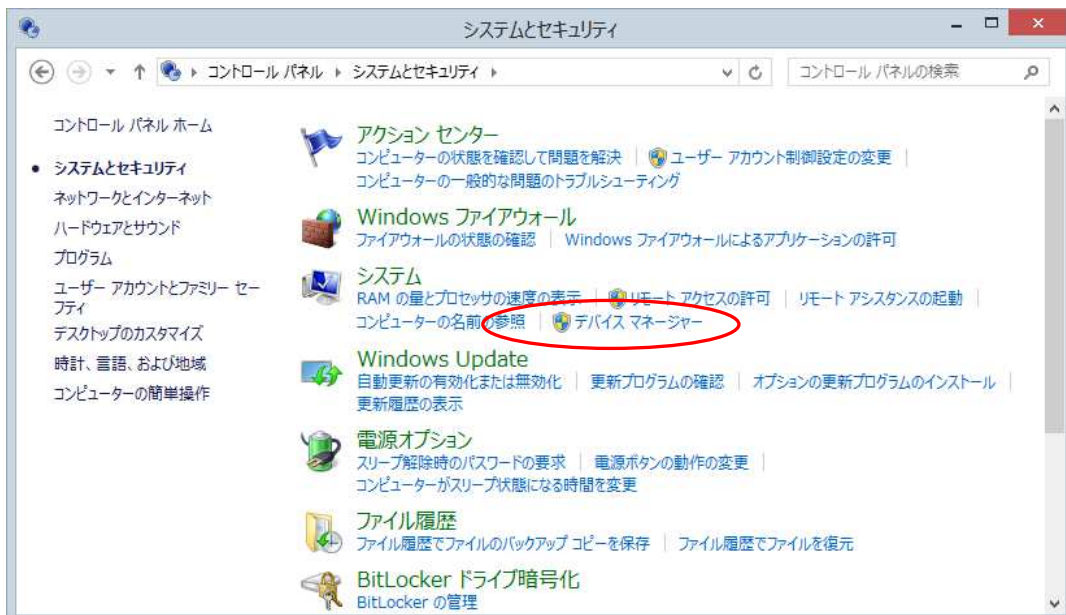
① リーダーライターの電源を ON する。

リーダーライターのUSBポートとコンピューターのUSBポート間にUSBケーブルが接続されていることを確認した後、リーダーライターの電源をONにして下さい。

② コントロールパネルを開いて、「システムとセキュリティ」をクリックします。



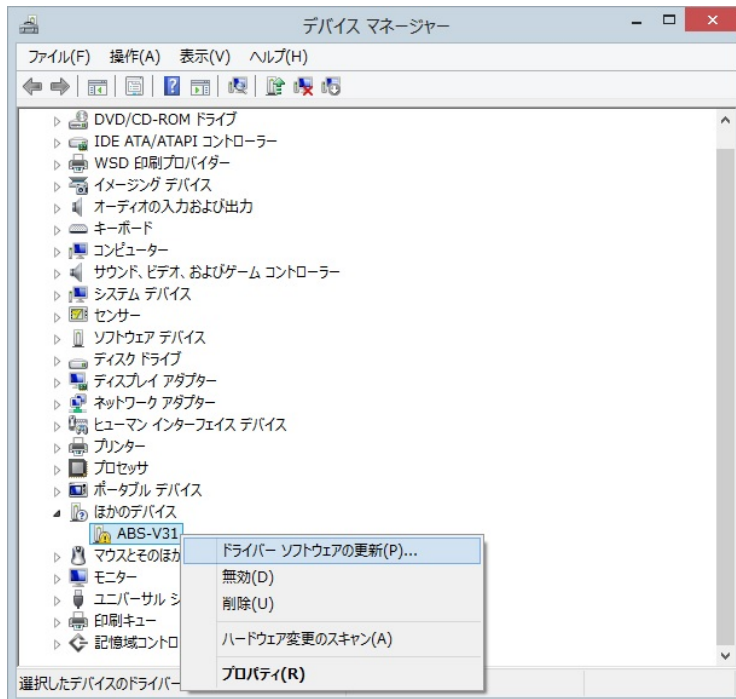
③ 「デバイスマネージャー」をクリックします。



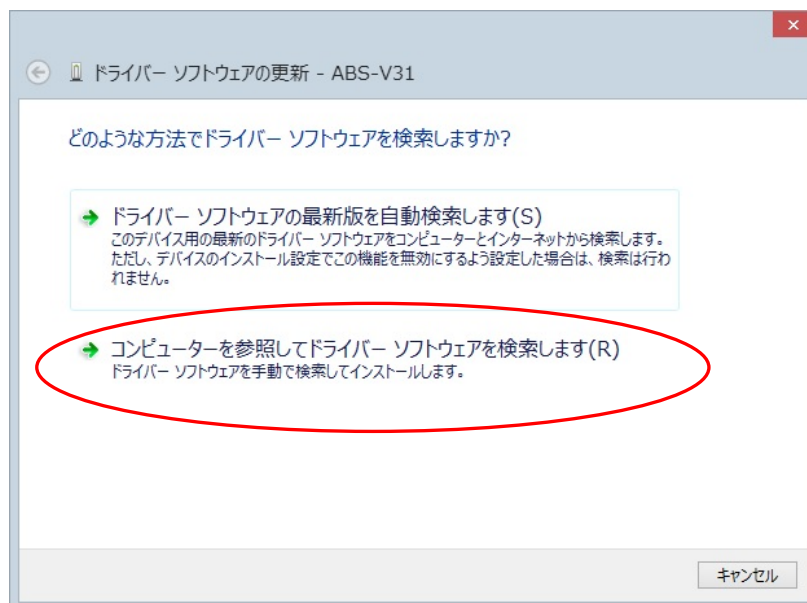
※デバイスマネージャーが表示されていない場合は、「システム」をクリックし、画面左側にある、「デバイスマネージャー」をクリックします。

- ④ 「ほかのデバイス」に「ABS-V31」と表示されます。(ABS-L31U 接続時も「ABS-V31」と表示されます。△)

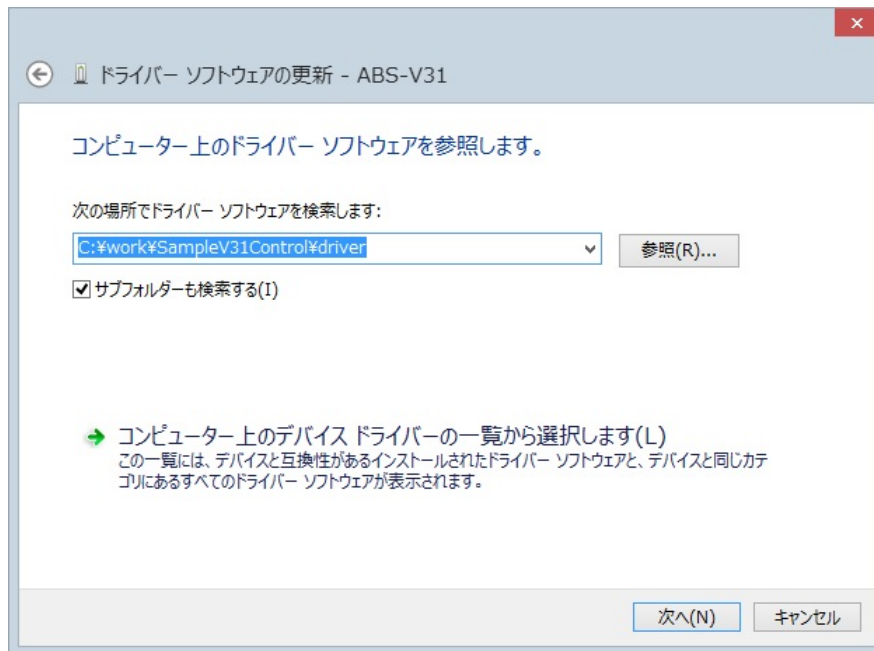
「ABS-V31」を右クリックして、「ドライバーソフトウェアの更新」をクリックします。



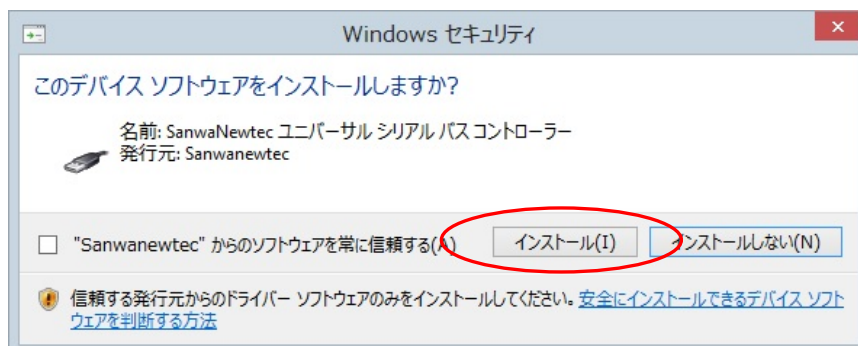
- ⑤ 「コンピューターを参照してドライバーソフトウェアを検索します」をクリックします。



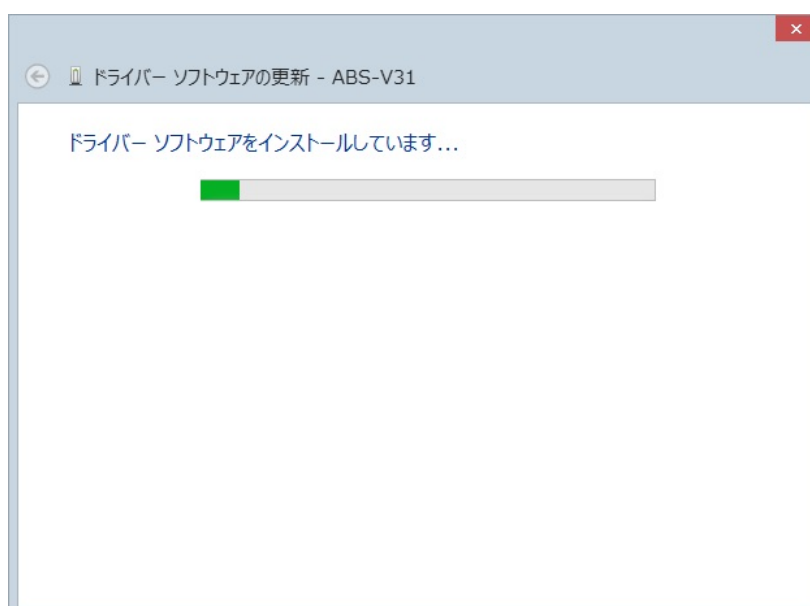
- ⑥USB ドライバーの inf ファイルを指定します。
inf ファイルは、展開したフォルダーの「SampleV31Control¥driver」にあります。
フォルダーを選択したら、「次へ」ボタンをクリックします。



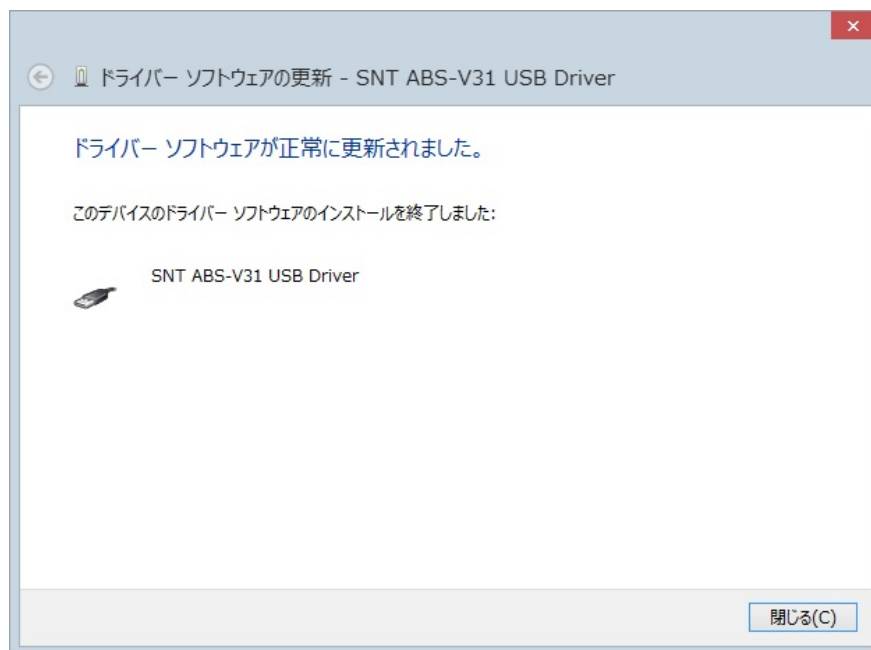
- ⑦以下画面が表示された場合は、「インストール」ボタンをクリックします。⚠



- ⑧インストールを開始します。



- ⑨インストールが完了すると下記のダイアログが表示されます。
「閉じる」ボタンをクリックします。



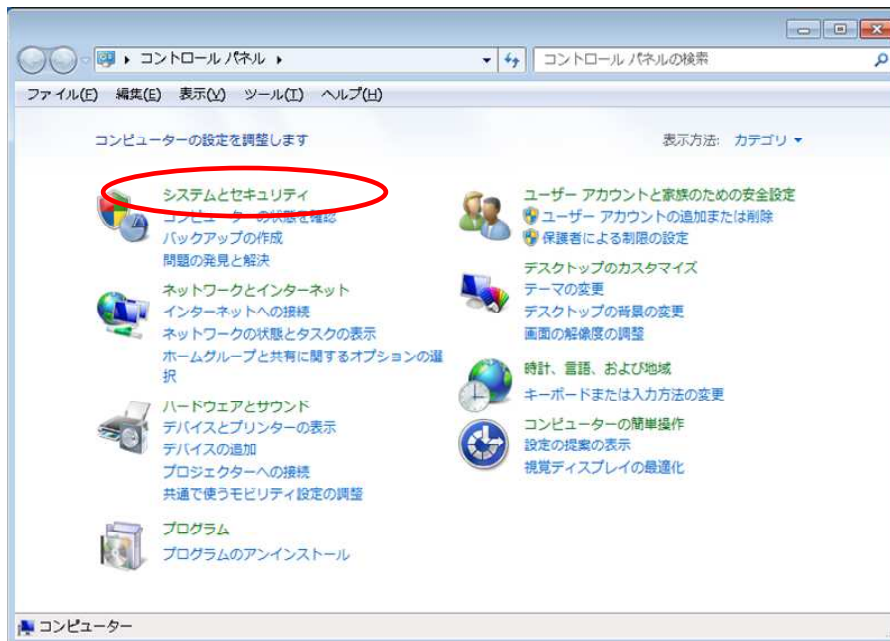
以上で、USB ドライバーのインストールは完了となります。

5. USB ドライバーのアンインストール方法

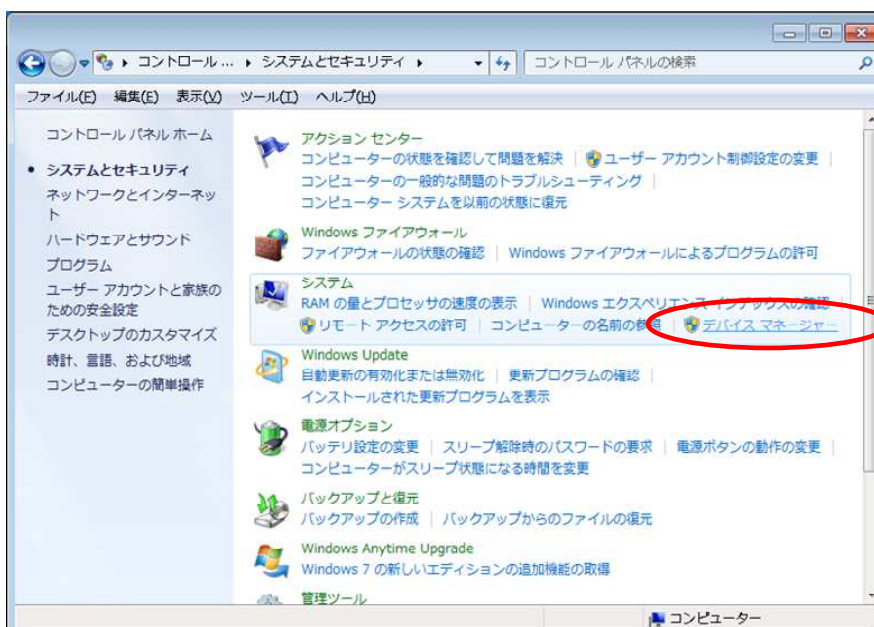
5-1. Windows7 の場合

1) デバイスマネージャーを開いて USB ドライバーをアンインストールする。

①コントロールパネルを開いて、「システムとセキュリティ」を選択します。



②「デバイスマネージャー」を選択します。



- ③ 「ユニバーサルシリアルバスコントローラー」の「SNT ABS-V31 USB Driver」を右クリックして「削除」をクリックします。



- ④ 「このデバイスのドライバーソフトウェアを削除する」にチェックをして、「OK」 ボタンをクリックします。



- ⑤ アンインストールを開始します。



- ⑥アンインストールが完了すると、「ユニバーサルシリアルバスコントローラー」のリストから「SNT ABS-V31 USB Driver」が削除されます。



以上で、USB ドライバーのアンインストールは完了となります。

5-2. Windows8/8.1/10 の場合^{②③}

1) デバイスマネージャーを開いて USB ドライバーをアンインストールする。

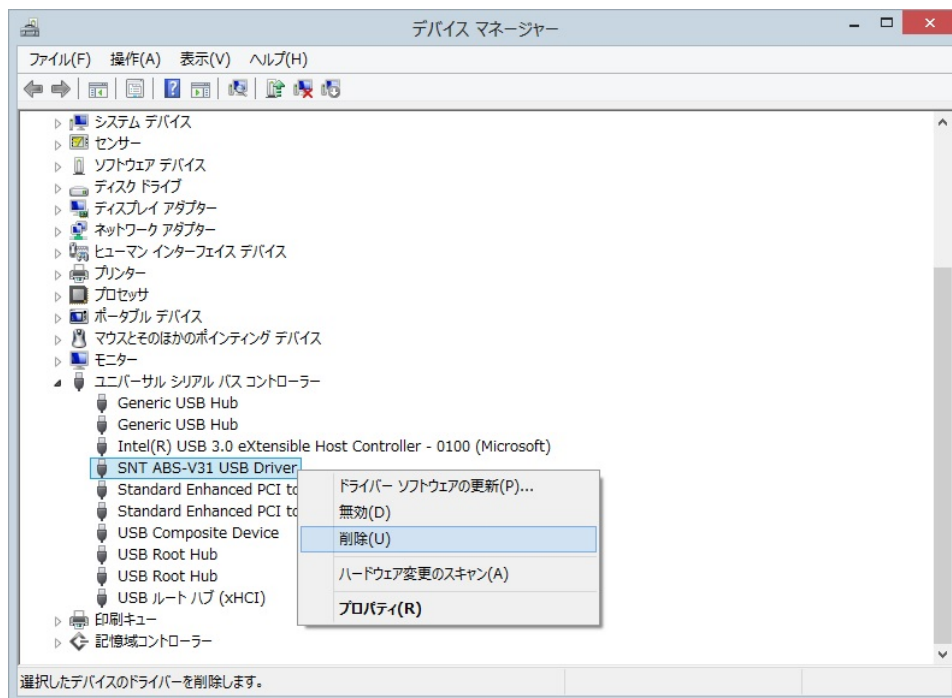
①コントロールパネルを開いて、「システムとセキュリティ」を選択します。



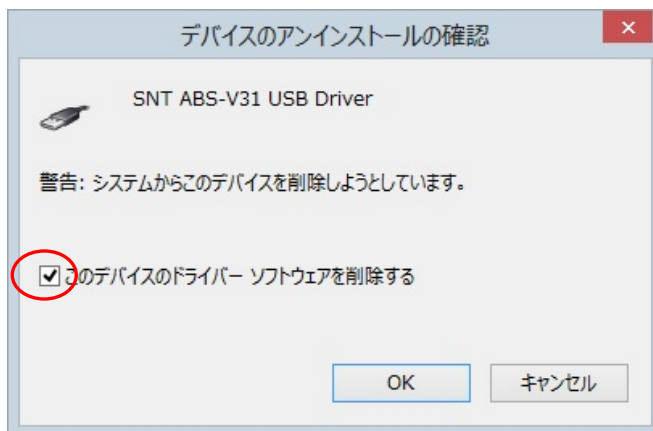
②「デバイスマネージャー」をクリックします。



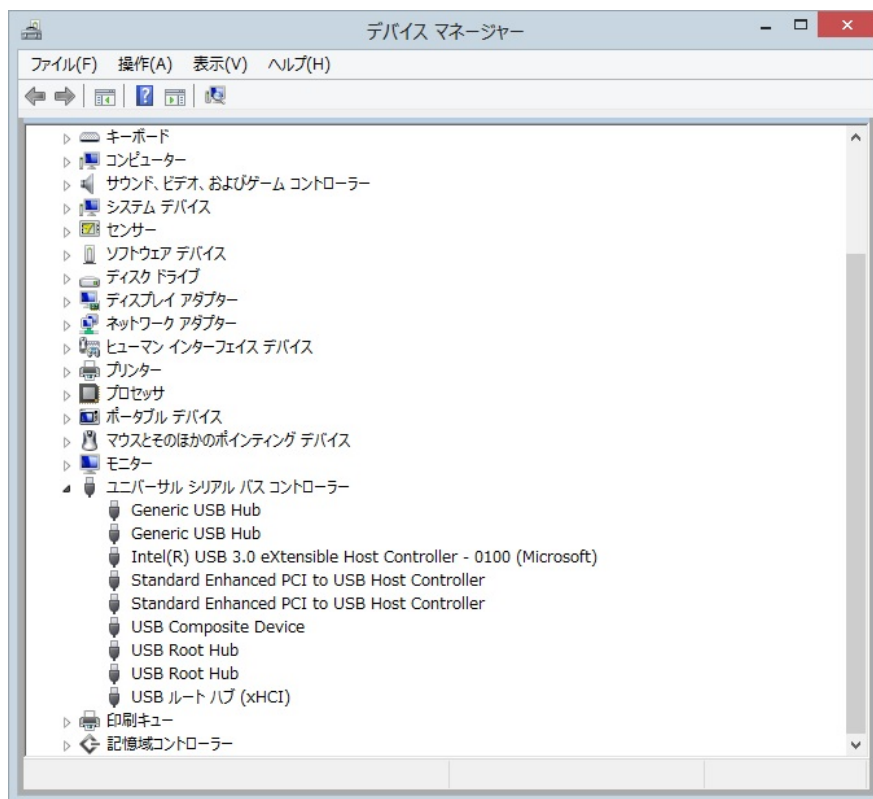
- ③ 「ユニバーサルシリアルバスコントローラー」の「SNT ABS-V31 USB Driver」を右クリックして「削除」をクリックします。



- ④ 「このデバイスのドライバーソフトウェアを削除する」にチェックをして、「OK」 ボタンをクリックします。



- ⑤ アンインストールが完了すると、「ユニバーサルシリアルバスコントローラー」のリストから「SNT ABS-V31 USB Driver」が削除されます。



以上で、USB ドライバーのアンインストールは完了となります。

5-3. Windows11 の場合△

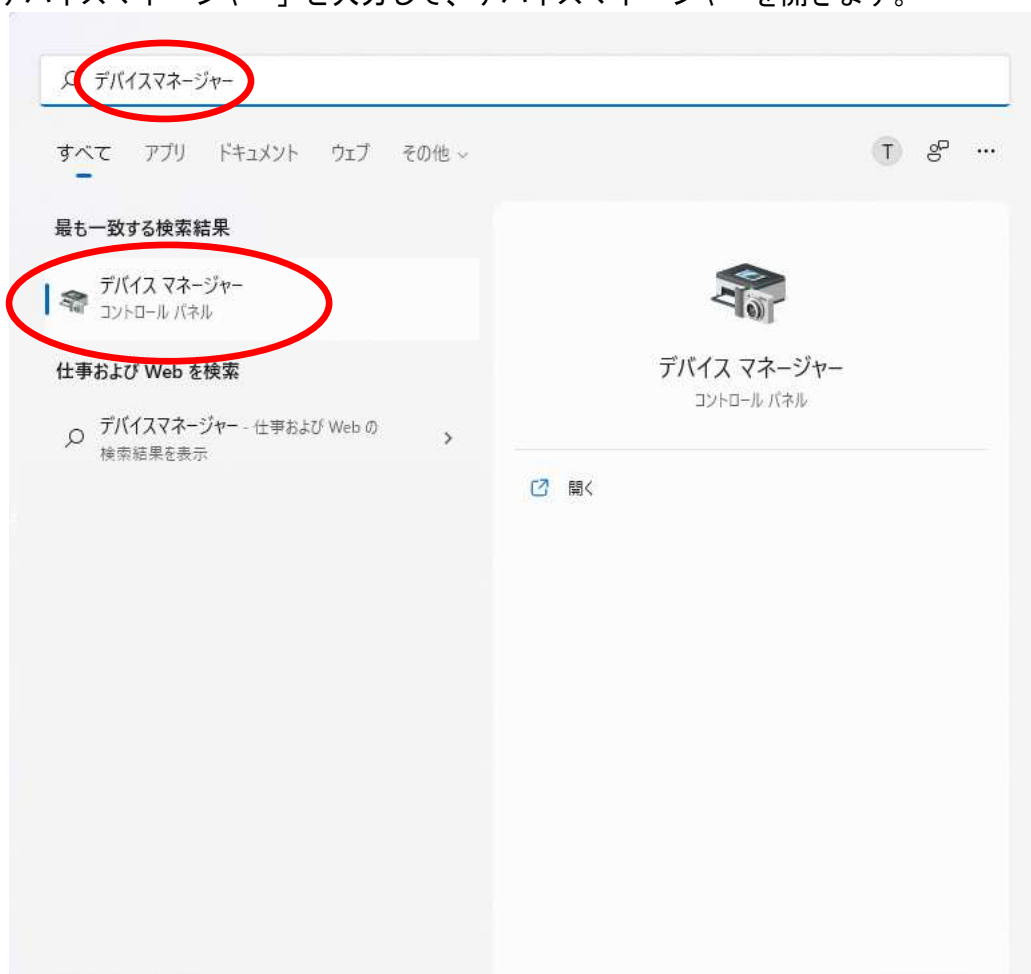
1) デバイスマネージャーを開く

Windows タスクバーで「検索」のアイコンをクリックします。



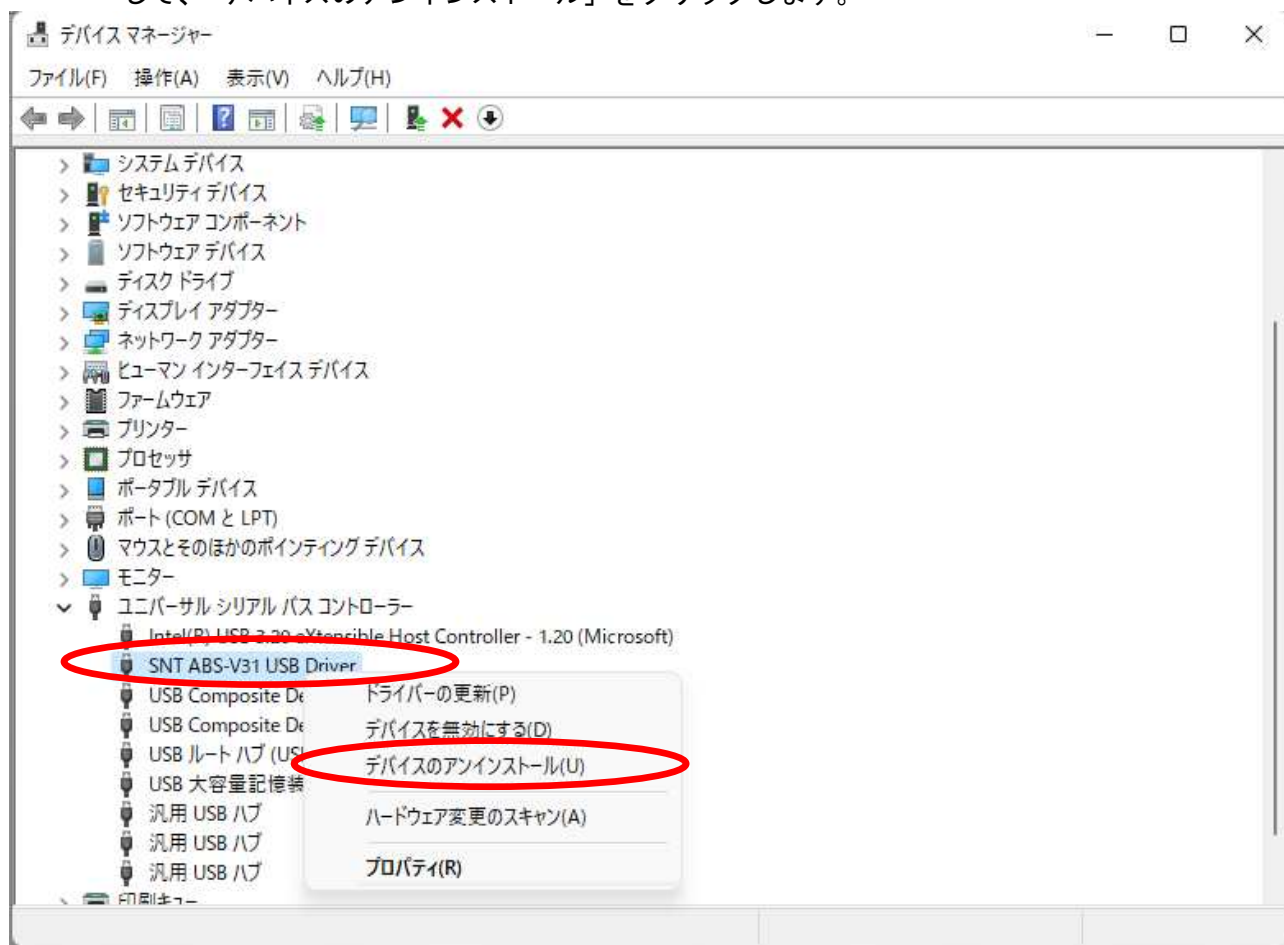
検索のアイコン

「デバイスマネージャー」と入力して、デバイスマネージャーを開きます。



2) 「SNT ABS-V31 USB Driver」を選択する

「ユニバーサルシリアルバスコントローラー」の「SNT ABS-V31 USB Driver」を右クリックして、「デバイスのアンインストール」をクリックします。



3) アンインストールする

「このデバイスのドライバーを削除しようとしています。」をクリックして、「アンインストール」をクリックします。



アンインストールが完了すると、「ユニバーサルシリアルバスコントローラー」のリストから「SNT ABS-V31 USB Driver」が削除されます。

以上で、USB ドライバーのアンインストールは完了となります。

6. 付録

6-1. レスポンス・ステータス一覧表

	コード	意味	備考
R P S 1 カード状態	3 0 h	装置内カード無し	装置内カード無し
	3 1 h	装置内カード有り	装置内カード有り
	3 2 h	カード状態異常	3 0、3 1、3 3、3 4 hに該当しないセンサー状態
	3 3 h	装置内カード有り 2	装置内後部にカード有り（センサー5のみ遮光された状態）
	3 4 h	挿入口カード残留 *1)	カード排出コマンドで排出されたカードが挿入口に残留
R P S 2 エラー状態	3 0 h	エラー無し	エラーの発生無し
	3 1 h	リードエラー	指定された全トラックの磁気データ読み込みエラー（トラック指定によらず共通）
	3 2 h	ライトエラー	磁気データ書き込みエラー（ペリファイチェックエラー）
	3 3 h	カード詰まりエラー	カード詰まり
	3 5 h	プリンターエラー	ヘッド昇降の異常 サーマル／消去ヘッドの加熱エラー、予熱エラー
	3 8 h	不正処理エラー *2)	装置内のカードが抜き取られた際に発生するエラー
	4 0 h	電池電圧低下警告 *3)	バックアップ用電池の電圧が低い時に発生する警告
	4 2 h	磁気データ無し警告	カードに磁気データが無い場合に発生する警告
	5 1 h	トラック 1 リードエラー	トラック 1 の磁気データ読み込みエラー（トラック指定＝3 3、3 4、3 6 h の場合のみ発生）
	5 2 h	トラック 2 リードエラー	トラック 2 の磁気データ読み込みエラー（トラック指定＝3 3、3 5、3 6 h の場合のみ発生）
	5 3 h	トラック 3 リードエラー	トラック 3 の磁気データ読み込みエラー（トラック指定＝3 4、3 5、3 6 h の場合のみ発生）
	5 4 h	トラック 1、2 リードエラー	トラック 1 とトラック 2 の磁気データ読み込みエラー（トラック指定＝3 6 h の場合のみ発生）
	5 5 h	トラック 1、3 リードエラー	トラック 1 とトラック 3 の磁気データ読み込みエラー（トラック指定＝3 6 h の場合のみ発生）
	5 6 h	トラック 2、3 リードエラー	トラック 2 とトラック 3 の磁気データ読み込みエラー（トラック指定＝3 6 h の場合のみ発生）
R P S 3 処理状態	3 0 h	コマンド実行終了	コマンドの実行が終了
	3 2 h	コマンド実行不可	実行不可能なコマンドを受信、またはコマンド伝文に誤り（パラメーター異常）がある
	3 3 h	コマンド実行中	コマンド実行中
	3 4 h	カード挿入待ち	リード、クリーニングコマンド受付後の、カード挿入待ち状態

* 1) 新たにカードを挿入口に差し込んだ場合には、リーダーライターは「3 0 h : 装置内カード無し」を返します。「3 4 h : 挿入口カード残留」は、排出コマンド実行後にカードが挿入口に残留している場合のレスポンスです。

* 2) 処理中に上パネルを開けられた場合、またはカードが抜き取られた場合等、搬送路センサー状態に異常が発生した場合のレスポンスです。「イニシャルコマンド」で「互換 2 または V 3 1 モード」を指定した場合に発生します。指定しなかった場合、または「互換 1」を指定した場合には、「3 3 h : カード詰まり」を返します。「V 3 1 モード」では、不正処理エラーが発生した場合、「イニシャルコマンド」を実行するまで、他のコマンドを受け付けません。

* 3) R P S 2 の警告レスポンス（4 0、4 2 h）は、イニシャルコマンドのレスポンスモード指定で、「3 2 h : V 3 1 モード」を指定した場合のみ、発生します。

6-2. コマンド・レスポンス対応表

ステータス コマンド		レスポンス	R P S 1				R P S 2								R P S 3			
		コード	30	31 33	32	34	30	31	32	33	35	38	40	42	30	32	33	34
		名称	装置内カード無し	装置内カード有り(2)	カード状態異常	挿入口カード残留	エラー無し	リードエラー	ライトエラー	カード詰まりエラー	プリンターエラー	不正処理エラー	電池電圧低下警告	磁気データ無し警告	コマンド実行終了	コマンド実行不可	コマンド実行中	カード挿入待ち
コード	名称																	
10h	イニシャル		○	○	○	○	○			○	○	○	○		○	○	○	
20h	ステータスリード		○	○	○	○	○			○		○			○			
21h	センサー情報取得		○	○	○	○	○			○		○			○			
33h	リード		○	○	○	○	○	○		○	○	○		○	○	○	○	○
40h	キャンセル		○	○	○	○	○			○		○			○	○		
53h	ライト		○	○	○	○	○		○	○	○	○			○	○	○	
76h	外字登録		○	○	○	○	○			○		○			○	○		
77h	イメージデータ登録		○	○	○	○	○			○		○			○	○		
78h	印字設定		○	○	○	○	○			○		○			○	○		
7Bh	イメージデータ印字		○	○	○	○	○			○	○	○			○	○	○	
7Ch	文字印字		○	○	○	○	○			○	○	○			○	○	○	
7Eh	バーコード印字		○	○	○	○	○			○	○	○			○	○	○	
80h	カード排出		○	○	○	○	○			○	○	○			○	○	○	
A0h	クリーニング		○	○	○	○	○			○	○	○			○	○	○	○
E1h	時計設定		○	○	○	○	○			○		○			○	○		
F0h	バージョン取得		○	○	○	○	○			○		○			○			
F1h	現在時刻取得		○	○	○	○	○			○		○			○			

注1) RPS2がエラー(31～61h)となった場合、RPS3は「30h: コマンド実行終了」となります。またRPS3が「30h: コマンド実行終了」以外の場合、RPS2は「エラー無し」となります。

注2) RPS2のトラック別リードエラー(51～56h)の対応は、31hのリードエラーと同じになります。

6-3. 印字ANK文字コード表

上位 下位	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0			SP	0	@	P	`	p				-	タ	ミ		
1		DC 1	!	1	A	Q	a	q			。	ア	チ	ム		
2			“	2	B	R	b	r			「	イ	ツ	メ		
3			#	3	C	S	c	s			」	ウ	テ	モ		
4		DC 4	\$	4	D	T	d	t			、	エ	ト	ヤ		
5			%	5	E	U	e	u			・	オ	ナ	ユ		
6			&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
7			‘	7	G	W	g	w			ア	キ	ヌ	ラ		
8			(8	H	X	h	x			イ	ク	ネ	リ		
9)	9	I	Y	i	y			ウ	ケ	ノ	ル		
A			*	:	J	Z	j	z			エ	コ	ハ	レ		
B		ES C	+	:	K	[k	{			オ	サ	ヒ	ロ		
C			,	<	L	¥	l				ヤ	シ	フ	ワ		
D	CR		-	=	M]	m	}			ユ	ス	ヘ	ン		
E			.	>	N	^	n	—			ヨ	セ	ホ	°		
F			/	?	O	_	o	■			ツ	ソ	マ	°		

※ CR, DC1, DC4, ESC は制御コードを示します（印字されるイメージではありません）

6-4. ISO 様式 トラック 1 使用文字一覧

文字	2 進符号							文字	2 進符号						
	p	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰		p	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰
スペース	1	0	0	0	0	0	0	@	0	1	0	0	0	0	0
!	0	0	0	0	0	0	1	A	1	1	0	0	0	0	1
"	0	0	0	0	0	1	0	B	1	1	0	0	0	1	0
#	1	0	0	0	0	1	1	C	0	1	0	0	0	1	1
\$	0	0	0	0	1	0	0	D	1	1	0	0	1	0	0
%	1	0	0	0	1	0	1	E	0	1	0	0	1	0	1
&	1	0	0	0	1	1	0	F	0	1	0	0	1	1	0
'	0	0	0	0	1	1	1	G	1	1	0	0	1	1	1
(0	0	0	1	0	0	0	H	1	1	0	1	0	0	0
)	1	0	0	1	0	0	1	I	0	1	0	1	0	0	1
*	1	0	0	1	0	1	0	J	0	1	0	1	0	1	0
+	0	0	0	1	0	1	1	K	1	1	0	1	0	1	1
,	1	0	0	1	1	0	0	L	0	1	0	1	1	0	0
-	0	0	0	1	1	0	1	M	1	1	0	1	1	0	1
.	0	0	0	1	1	1	0	N	1	1	0	1	1	1	0
/	1	0	0	1	1	1	1	O	0	1	0	1	1	1	1
0	0	0	1	0	0	0	0	P	1	1	1	0	0	0	0
1	1	0	1	0	0	0	1	Q	0	1	1	0	0	0	1
2	1	0	1	0	0	1	0	R	0	1	1	0	0	1	0
3	0	0	1	0	0	1	1	S	1	1	1	0	0	1	1
4	1	0	1	0	1	0	0	T	0	1	1	0	1	0	0
5	0	0	1	0	1	0	1	U	1	1	1	0	1	0	1
6	0	0	1	0	1	1	0	V	1	1	1	0	1	1	0
7	1	0	1	0	1	1	1	W	0	1	1	0	1	1	1
8	1	0	1	1	0	0	0	X	0	1	1	1	0	0	0
9	0	0	1	1	0	0	1	Y	1	1	1	1	0	0	1
:	0	0	1	1	0	1	0	Z	1	1	1	1	0	1	0
;	1	0	1	1	0	1	1	[0	1	1	1	0	1	1
<	0	0	1	1	1	0	0	¥	1	1	1	1	1	0	0
=	1	0	1	1	1	0	1]	0	1	1	1	1	0	1
>	1	0	1	1	1	1	0	^	0	1	1	1	1	1	0
?	0	0	1	1	1	1	1	_	1	1	1	1	1	1	1

注) 背景がグレーの文字については、以下の制約がありますので、ご注意ください。

データとして使用した場合、コマンド実行不可となります。

※「!」、「"」、「&」、「'」、「*」、「+」、「,」、「:」、「;」、「<」、「=」、「>」、「@」、「_」の 14 文字は、未対応コードとなるため、データとしては使用しないでください。

※「%」、「?»の 2 文字は、以下の意味に限定しますので、データとしては使用しないでください。

- ・「%」: 始め符号
- ・「?»: 終わり符号

6-5. ISO 様式 トラック 2、3 使用文字一覧

文字	2 進符号					文字	2 進符号				
	P	2 ³	2 ²	2 ¹	2 ⁰		P	2 ³	2 ²	2 ¹	2 ⁰
0	1	0	0	0	0	8	1	1	0	0	0
1	0	0	0	0	1	9	1	1	0	0	1
2	0	0	0	1	0	:	1	1	0	1	0
3	1	0	0	1	1	;	0	1	0	1	1
4	0	0	1	0	0	<	1	1	1	0	0
5	1	0	1	0	1	=	0	1	1	0	1
6	1	0	1	1	0	>	0	1	1	1	0
7	0	0	1	1	1	?	1	1	1	1	1

注) 背景がグレーの文字については、以下の制約がありますので、ご注意ください。

データとして使用した場合、コマンド実行不可となります。

※ 「:」、「<」、「>」の3文字は、未対応コードとなるため、データとしては使用しないでください。

※ 「;」、「?」の2文字は、以下の意味に限定します。

- ・「;」: 始め符号
- ・「?」: 終わり符号

6-6. 印字範囲 1「文字印字（行指定モード）」

注）印字範囲は、目安の値です（通常±0.5mm カードの状態等により±1mm程の誤差も出ます）

図 1-1. 標準 [全角 13 文字×20 行]、行高さ：24 ドット，行間隔：5 ドット

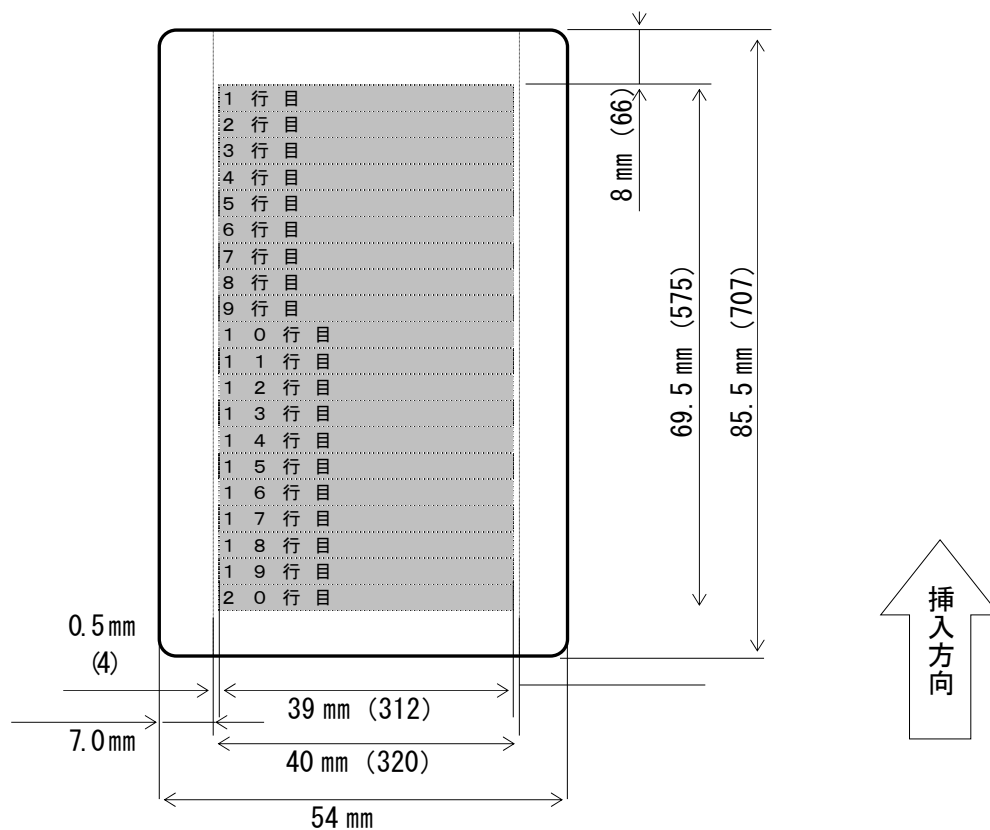
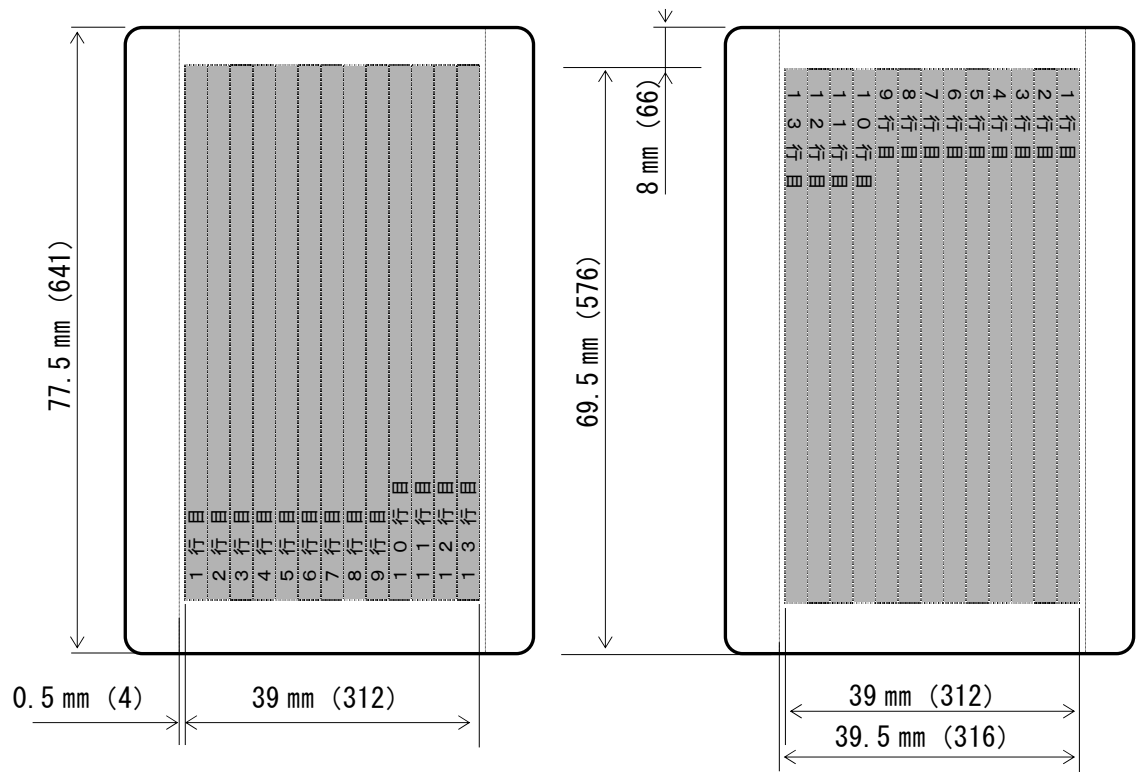


図 1-2. 左回転 [全角 24 文字×13 行]

図 1-3. 右回転 [全角 24 文字×13 行]

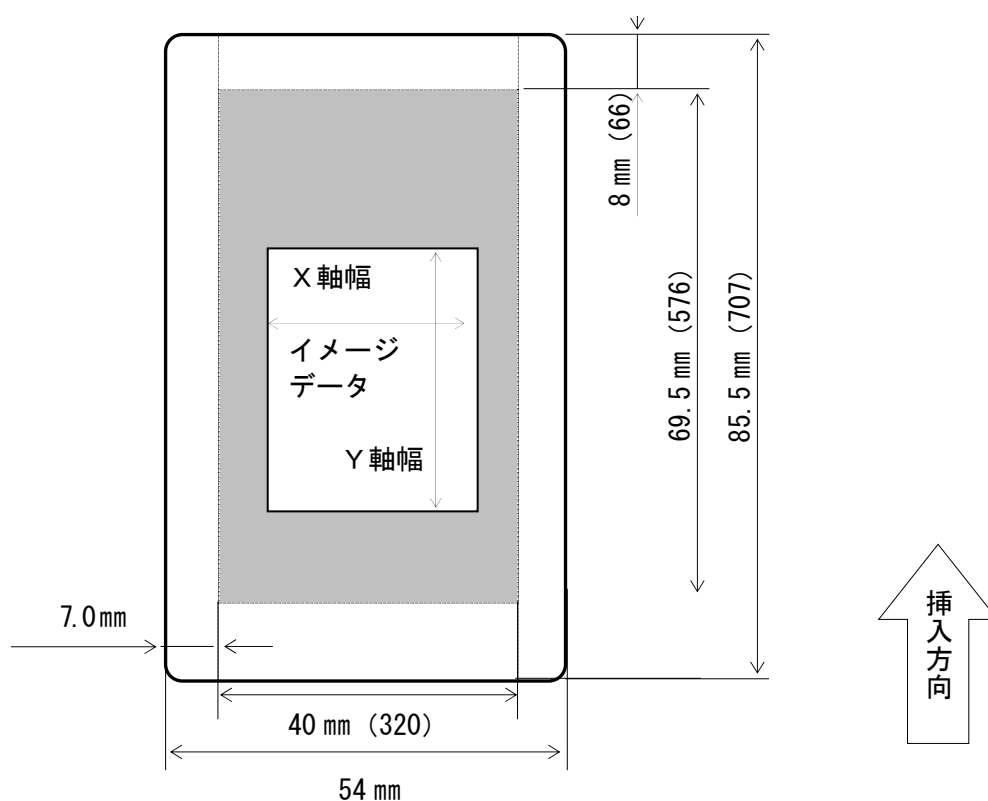
行高さ：24 ドット，行間隔：0 ドット



※寸法表記中の（ ）はドット数を示します。

6-7. 印字範囲 2「イメージデータ・文字（任意座標モード）・バーコード」

注) 印字範囲は、目安の値です（通常：±0.5 mm カードの状態等により±1 mm程度の誤差も出ます）



※寸法表記中の（ ）はドット数を示します。

※1ドットは長手方向0.121 mm、短手方向0.125 mmです。