CASIO

プリンタライブラリ
マニュアル

このマニュアルは、プリンタライブラリの
仕様について記載します。

Ver. 1.06

ご注意

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

変更履歴

バージョン	変更日付	ページ	内容
1.00	2008.11		新規作成
1.01	2009.03	2	開発環境とプログラミング言語の対応表を訂正
		27	PRNSetPrinterProperty関数の宣言の誤記を訂正
		30	PRNGetPrinterProperty関数の宣言の誤記を訂正
1.03	2010.05	38	内蔵漢字フォントの詳細を追加
1.04	2011.11	－	対象機種に IT-9000 を追加
		35	PRNSetBarcodeType関数を追加
		37	PRNGetBarcodeType関数を追加
	2012.01	42、54	ESC コマンドを追加
	2012.02	35	PRNSetBarcodeType関数に戻り値を追加
		37	PRNGetBarcodeType関数に戻り値を追加
	2012.03	7	PRNInitializePrinter関数に戻り値を追加
		11	PRNTextOut関数に説明を追加
		25	PRNSetPaperWidth関数に戻り値を追加
		26	PRNGetPaperWidth関数に戻り値を追加
		27	PRNSetPrinterProperty関数に戻り値を追加
		30	PRNGetPrinterProperty関数に戻り値を追加
		54	ESC コマンド（マーカ検出モード）を修正
		44	「裏面バーコードの読み取りについて」を追加
		45～50	サンプルプログラムを追加
1.05	2012.12	56,57	ESC コマンド（紙送り）を修正
1.06	2014.05	7～30	PRN_SUSPEND_OCCURRED の説明を追加

目次

1. 概要

プリンタライブラリは、携帯情報端末に内蔵しているプリンタモジュールを利用した、印刷機能を提供します。
プリンタライブラリを使用すれば、業務アプリケーションから直接プリンタモジュールを制御できます。

プリンタクラスライブラリは、プリンタライブラリを .NET Compact Framework アプリケーションから直接利用できるようにする、ラッパーライブラリです。

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

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

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

2. 動作環境

プリンタライブラリの動作環境を以下に示します。

対象機種

DT-9800 / IT-9000

対象 OS

- Microsoft Windows CE 5.0

開発環境とプログラミング言語

表 2-1

開発環境	Visual C++	Visual Basic, Visual C#
Microsoft embedded Visual C++ Version 4.0 + SP4	○	-
Microsoft Visual Studio.NET 2003 + SP1	×	○
Microsoft Visual Studio 2005 + SP1	○	○
Microsoft Visual Studio 2008	○	○

（○:利用可、×:利用不可、-:機能なし）

提供ファイル

表 2-2

ファイル	Visual C++	Visual Basic, Visual C#
PrinterLib.h	○	-
PrinterLib.lib	○	-
PrinterLib.dll	○	○
PrinterLibNet.dll（クラスライブラリ）	-	○

（○:必要、-:不要）

使用方法

Visual C++ の場合

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

Visual Basic または Visual C# の場合

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

名前空間とクラス

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

表 2-3

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

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

3. 関数

プリンタライブラリの提供する関数は、以下のとおりです。

表 3-1 関数一覧

関数名	概要	DT-9800	IT-9000
PRNOpen	プリンタの電源を投入し、プリンタを使用可能に設定	○	○
PRNClose	プリンタを開放し、プリンタ電源 OFF	○	○
PRNInitializePrinter	プリンタの設定の初期化	○	○
PRNPrintScreen	フルスクリーン画面の印刷	○	○
PRNPrintWindow	クライアントウィンドウの印刷	○	○
PRNTextOut	テキストデータの印刷	○	○
PRNImageOut	ビットイメージの印刷	○	○
PRNBarcodeOut	指定されたバーコードの印刷	○	○
PRNBMPOut	指定された Bitmap の印刷	○	○
PRNCheckMarker	マーカ位置までの紙送り	○	○
PRNGetStatus	現在のプリンタのエラー状態の取得	○	○
PRNGetLastError	最後に印刷した時のプリンタのエラー状態の取得	○	○
PRNSetPaperWidth	使用する用紙の用紙幅の指定	○	○
PRNGetPaperWidth	用紙幅の設定状態の取得	○	○
PRNSetPrinterProperty	プリンタの各種設定	○	○
PRNGetPrinterProperty	プリンタの各種設定状態の取得	○	○
PRNDecodeBarcode	裏面バーコードのデコード	-	○
PRNResetDecoder	裏面バーコードの読取データリセット	-	○
PRNSetBarcodeType	裏面バーコードの読取方式の設定	-	○
PRNGetBarcodeType	裏面バーコードの読取方式の取得	-	○

○ 関数サポート

― 関数未サポート＝ 関数を呼ぶと未サポートエラーが返ります。

3.1 PRNOpen

プリンタの電源を投入し、プリンタを使用可能にします。

```
[C++]
DWORD PRNOpen()
```

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

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

パラメータ

なし

戻り値

PRN_NORMAL	: 正常終了
PRN_ALREADY_OPEN	: アプリケーションの二重オープンか、正常にプリンタをオープンできません
PRN_DRIVER_NOTEXIST	: 他のアプリケーションで既に使用されているか、プリンタドライバが非常駐状態のためプリンタが使用できません Device Emulator では発生しません
PRN_FEEDKEY_ERROR	: フィードキーでフィード中のため、プリンタが使用できません。 Device Emulator では発生しません
FUNCTION_UNSUPPORT	: 未サポートエラー

補足

フィードキーによるフィード中は、アプリケーションからのプリンタ制御はできません。
また、アプリケーションがプリンタ制御している間は、フィードキーによるフィードはできません。

対応情報

機種	: DT-9800 / IT-9000
ヘッダ	: PrinterLib.h
ライブラリ	: PrinterLib.lib

3.2 PRNClose

プリンタを開放し、プリンタの電源を切ります。

```
[C++]
DWORD PRNClose()
```

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

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

パラメータ

なし

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません
FUNCTION_UNSUPPORT	: 未サポートエラー

補足

本関数を実行する前は、必ず ESC コマンドの「未印字吐き出し」を実行してください。

対応情報

機種	: DT-9800 / IT-9000
ヘッダ	: PrinterLib.h
ライブラリ	: PrinterLib.lib

3.3 PRNInitializePrinter

レジストリに保存されないプリンタの設定を初期化します。

```
[C++]
DWORD PRNInitializePrinter()
```

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

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

パラメータ

なし

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません
PRN_VDETP_OCCURRED	: VDETP が発生しました Device Emulator では発生しません
PRN_SUSPEND_OCCURRED	: 印刷中にサスペンドが発生し、印刷を中止しました。 Device Emulator では発生しません
FUNCTION_UNSUPPORT	: 未サポートエラー

対応情報

機種	: DT-9800 / IT-9000
ヘッダ	: PrinterLib.h
ライブラリ	: PrinterLib.lib

3.4 PRNPrintScreen

フルスクリーン画面を印刷します。

```
[C++]
DWORD PRNPrintScreen()
```

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

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

解説

本関数は、フルスクリーン画面を印刷します。
DeviceEmulatorでは、フルスクリーン画面をPrinterImage.binに出力します。詳細は、PrinterImage.binを参照してください。

パラメータ

なし

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_HARDWARE_ERROR	: プリンタのハードウェアが異常です。 Device Emulator では発生しません
PRN_PLATEN_OPEN	: プラテンがオープンしています。 Device Emulator では発生しません
PRN_PAPER_END	: 用紙がありません。 Device Emulator では発生しません
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません
PRN_SUSPEND_OCCURRED	: 印刷中にサスペンドが発生し、印刷を中止しました。 Device Emulator では発生しません
PRN_HEADTEMP_ERROR	: ヘッド温度エラーが発生しました。 Device Emulator では発生しません
PRN_AUTOLOADING	: オートローディング中です。 Device Emulator では発生しません
PRN_COVER_CLOSE	: スプラッシュカバーが閉じています。 DeviceEmulator では発生しません。
FUNCTION_UNSUPPORT	: 未サポートエラー

対応情報

機種	: DT-9800 / IT-9000
ヘッダ	: PrinterLib.h
ライブラリ	: PrinterLib.lib

3.5 PRNPrintWindow

指定されたウィンドウを印刷します。

```
[C++]
DWORD PRNPrintWindow(
  HWND hWindow
)
```

```
[Visual Basic]
Public Shared Function PRNPrintWindow( _
  ByVal hWindow As IntPtr _
) As Int32
```

```
[C#]
public static Int32 PRNPrintWindow(
  IntPtr hWindow
);
```

解説

本関数は、指定したウィンドウを印刷します。
DeviceEmulatorでは、指定したウィンドウをPrinterImage.binに出力します。詳細は、PrinterImage.binを参照してください。

パラメータ

hWindow
印刷するウィンドウのハンドルを指定します。

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_HARDWARE_ERROR	: プリンタのハードウェアが異常です。 Device Emulator では発生しません
PRN_PLATEN_OPEN	: プラテンがオープンしています。 Device Emulator では発生しません
PRN_PAPER_END	: 用紙がありません。 Device Emulator では発生しません
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません
PRN_SUSPEND_OCCURRED	: 印刷中にサスペンドが発生し、印刷を中止しました。 Device Emulator では発生しません
PRN_HEADTEMP_ERROR	: ヘッド温度エラーが発生しました。 Device Emulator では発生しません
PRN_AUTOLOADING	: オートローディング中です。 Device Emulator では発生しません
PRN_COVER_CLOSE	: スプラッシュカバーが閉じています。 DeviceEmulator では発生しません。

FUNCTION_UNSUPPORT　　：未サポートエラー

補足

指定されたウィンドウのうち、画面に表示されている部分のみ印刷することが可能です。

対応情報

機種　　：DT-9800 / IT-9000
ヘッダ　　：PrinterLib.h
ライブラリ　：PrinterLib.lib

3.6 PRNTextOut

テキストデータを印刷します。
また、ESC コマンドを送信する場合も本関数を使用します。

```
[C++]
DWORD PRNTextOut(
  DWORD dwLength,
  TCHAR *szTextData
)
```

```
[Visual Basic]
Public Shared Function PRNTextOut( _
  ByVal dwLength As Int32, _
  ByVal szTextData As String _
) As Int32
```

```
[C#]
public static Int32 PRNTextOut(
  Int32 dwLength,
  string szTextData
);
```

解説

本関数は、テキストデータを印刷します。
また、ESC コマンドを送信する場合も本関数を使用します。
DeviceEmulatorでは、テキストデータをPrinterImage.binに出力します。詳細は、PrinterImage.binを参照してください。

パラメータ

dwLength
テキストデータのデータ長を指定します。(0～65535)
DT-9800 では 0 を指定すると、szTextData の先頭から 0x0000 までを印刷(送信)します。
IT-9000 では 0 を指定すると、szTextData の先頭から 0x0000 の直前までを印刷(送信)します。

szTextData
テキストデータ(ESC コマンドも含む)が格納されているアドレスを指定します。

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_PARAMETER_ERROR	: パラメータエラー
PRN_HARDWARE_ERROR	: プリンタのハードウェアが異常です。 Device Emulator では発生しません
PRN_PLATEN_OPEN	: プラテンがオープンしています。 Device Emulator では発生しません
PRN_PAPER_END	: 用紙がありません。 Device Emulator では発生しません

PRN_VDETP_OCCURRED : VDETP が発生しました。
Device Emulator では発生しません
PRN_SUSPEND_OCCURRED : 印刷中にサスペンドが発生し、印刷を中止しました。
Device Emulator では発生しません
PRN_HEADTEMP_ERROR : ヘッド温度エラーが発生しました。
Device Emulator では発生しません
PRN_AUTOLOADING : オートローディング中です。
Device Emulator では発生しません
PRN_COVER_CLOSE : スプラッシュカバーが閉じています。
DeviceEmulator では発生しません。
FUNCTION_UNSUPPORT : 未サポートエラー

補足

PRNTextOut で入力する文字コードに関しては、「プログラミング上の注意点」を参照して下さい。

対応情報

機種 : DT-9800 / IT-9000
ヘッダ : PrinterLib.h
ライブラリ : PrinterLib.lib

3.7 PRNImageOut

ビットイメージデータを印刷します。

```
[C++]
DWORD PRNImageOut(
  DWORD dwWidth,
  DWORD dwHeight,
  DWORD dwFeedLength,
  BYTE *pbyImageData
)
```

```
[Visual Basic]
Public Shared Function PRNImageOut( _
  ByVal dwWidth As Int32, _
  ByVal dwHeight As Int32, _
  ByVal dwFeedLength As Int32, _
  ByVal pbyImageData As Byte() _
) As Int32
```

```
[C#]
public static Int32 PRNImageOut(
  Int32 dwWidth,
  Int32 dwHeight,
  Int32 dwFeedLength,
  Byte[] pbyImageData
);
```

解説

本関数は、ビットイメージデータを印刷します。
DeviceEmulatorでは、ビットイメージデータをPrinterImage.binに出力します。詳細は、PrinterImage.binを参照してください。

パラメータ

dwWidth
ビットイメージデータの横方向バイト数を指定します。
(58mm 用紙:1～48　80mm 用紙:1～72)

dwHeight
ビットイメージデータの縦方向ドットライン数を指定します。(1～65535)

dwFeedLength
ラインフィードする長さを指定します。(0～96)

pbyImageData
イメージデータが格納されているアドレスを指定します。

戻り値

PRN_NORMAL	: 正常終了
PRN_PARAMETER_ERROR	: パラメータエラー
PRN_NOTOPEN	: プリンタがオープンされていません。 Device Emulator では発生しません
PRN_HARDWARE_ERROR	: プリンタのハードウェアが異常です。 Device Emulator では発生しません
PRN_PLATEN_OPEN	: プラテンがオープンしています。 Device Emulator では発生しません
PRN_PAPER_END	: 用紙がありません。 Device Emulator では発生しません
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません
PRN_SUSPEND_OCCURRED	: 印刷中にサスペンドが発生し、印刷を中止しました Device Emulator では発生しません
PRN_HEADTEMP_ERROR	: ヘッド温度エラーが発生しました。 Device Emulator では発生しません
PRN_AUTOLOADING	: オートローディング中です。 Device Emulator では発生しません
PRN_COVER_CLOSE	: スプラッシュカバーが閉じています。 DeviceEmulator では発生しません。
FUNCTION_UNSUPPORT	: 未サポートエラー

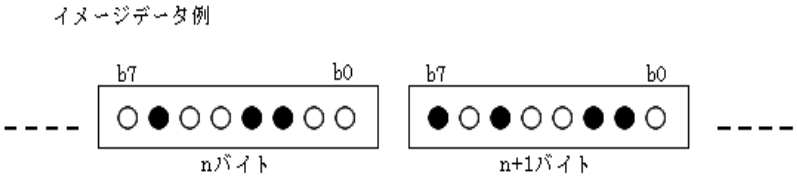


図 3.1 イメージデータ例

対応情報

機種 : DT-9800 / IT-9000
ヘッダ : PrinterLib.h
ライブラリ : PrinterLib.lib

3.8 PRNBarcodeOut

指定されたバーコードを印刷します。

```
[C++]
DWORD PRNBarcodeOut(
  DWORD dwCode,
  DWORD dwHeight,
  DWORD dwCheckDigit,
  DWORD dwFont,
  DWORD dwLeftMargin,
  DWORD dwDirection,
  DWORD dwLength,
  TCHAR *szBarcodeData
)
```

```
[Visual Basic]
Public Shared Function PRNBarcodeOut( _
  ByVal dwCode As Int32, _
  ByVal dwHeight As Int32, _
  ByVal dwCheckDigit As Int32, _
  ByVal dwFont As Int32, _
  ByVal dwLeftMargin As Int32, _
  ByVal dwDirection As Int32, _
  ByVal dwLength As Int32, _
  ByVal szBarcodeData As String _
) As Int32
```

```
[C#]
public static Int32 PRNBarcodeOut(
  Int32 dwCode,
  Int32 dwHeight,
  Int32 dwCheckDigit,
  Int32 dwFont,
  Int32 dwLeftMargin,
  Int32 dwDirection,
  Int32 dwLength,
  string szBarcodeData
);
```

解説

本関数は、指定したバーコードを印刷します。
DeviceEmulatorでは、指定したバーコードをPrinterImage.binに出力します。詳細は、PrinterImage.binを参照してください。

パラメータ

dwCode
印刷するバーコードの種類を指定します。
0 : JAN

1 : NW7
2 : CODE39
3 : ITF
4 : UPC-E
5 : CODE128

dwHeight
印刷するバーコードの高さを指定します。(1～63mm)

dwCheckDigit
チェックデジットの有無を指定します。
0 : チェックデジット無し
1 : チェックデジット有り

dwFont
バーコードの下に印刷するデータの有無や文字の種類を指定します。
0 : 付加文字無し
1 : 標準文字(ANK 8 X 16 ドット)
2 : 縮小文字(ANK 6 X 7 ドット)
3 : OCR-B Ⅰ (16 X 30 ドット)

dwLeftMargin
バーコードの左側に空けるスペースをミリ単位で指定します。
(0～69mm。センタリングする場合は 0xFFFF を指定)

dwDirection
バーコードを印刷する方向を指定します。
0 : 縦方向
1 : 横方向

dwLength
バーコードのデータ長を指定します。

szBarcodeData
バーコードデータ(Unicode)が格納されているアドレスを指定します。

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_PARAMETER_ERROR	: パラメータエラー
PRN_HARDWARE_ERROR	: プリンタのハードウェアが異常です。 Device Emulator では発生しません
PRN_PLATEN_OPEN	: プラテンがオープンしています。 Device Emulator では発生しません
PRN_PAPER_END	: 用紙がありません。 Device Emulator では発生しません
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません

PRN_SUSPEND_OCCURRED : 印刷中にサスペンドが発生し、印刷を中止しました。
Device Emulator では発生しません
PRN_HEADTEMP_ERROR : ヘッド温度エラーが発生しました。
Device Emulator では発生しません
PRN_AUTOLOADING : オートローディング中です。
Device Emulator では発生しません
PRN_COVER_CLOSE : スプラッシュカバーが閉じています。
DeviceEmulator では発生しません。
FUNCTION_UNSUPPORT : 未サポートエラー

補足

バーコード印刷は、必ず印字速度をグラフィックモードに設定して下さい。

対応情報

機種 : DT-9800 / IT-9000
ヘッダ : PrinterLib.h
ライブラリ : PrinterLib.lib

3.9 PRNBMPOut

指定された Bitmap を印刷します。

```
[C++]
DWORD PRNBMPOut(
  TCHAR *szFilename
)
```

```
[Visual Basic]
Public Shared Function PRNBMPOut( _
  ByVal szFilename As String _
) As Int32
```

```
[C#]
public static Int32 PRNBMPOut(
  string szFilename
);
```

解説

本関数は、指定した bmp ファイルを印刷します。
DeviceEmulatorでは、指定したbmpファイルをPrinterImage.binに出力します。詳細は、PrinterImage.binを参照してください。

パラメータ

szFilename
印刷するビットマップファイルのフルパス名が格納されているアドレスを指定します。

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_FILE_NOTEXIST	: 指定されたファイルがありません。
PRN_FILEOPEN_ERROR	: 指定されたファイルが開けません。
PRN_FILEFORMAT_ERROR	: 指定されたファイルの形式が不正です。 Device Emulator では発生しません
PRN_HARDWARE_ERROR	: プリンタのハードウェアが異常です。 Device Emulator では発生しません
PRN_PLATEN_OPEN	: プラテンがオープンしています。 Device Emulator では発生しません
PRN_PAPER_END	: 用紙がありません。 Device Emulator では発生しません
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません
PRN_SUSPEND_OCCURRED	: 印刷中にサスペンドが発生し、印刷を中止しました。 Device Emulator では発生しません
PRN_HEADTEMP_ERROR	: ヘッド温度エラーが発生しました。 Device Emulator では発生しません

PRN_AUTOLOADING : オートローディング中です。
Device Emulator では発生しません
PRN_COVER_CLOSE : スプラッシュカバーが閉じています。
DeviceEmulator では発生しません。
FUNCTION_UNSUPPORT : 未サポートエラー

補足

白黒 2 色の Bitmap のみ印刷可能です。

対応情報

機種 : DT-9800 / IT-9000
ヘッダ : PrinterLib.h
ライブラリ : PrinterLib.lib

3.10 PRNCheckMarker

マーカ位置まで紙送りします。

```
[C++]
DWORD PRNCheckMarker()
```

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

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

解説

本関数は、マーカ位置まで紙送りします。
Device Emulatorでは、PRNOpen関数の実行確認のみを行います。

パラメータ

なし

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_NOTFOUND	: マーカが検出できません Device Emulator では発生しません
PRN_HARDWARE_ERROR	: プリンタのハードウェアが異常です。 Device Emulator では発生しません
PRN_PLATEN_OPEN	: プラテンがオープンしています。 Device Emulator では発生しません
PRN_PAPER_END	: 用紙がありません。 Device Emulator では発生しません
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません
PRN_SUSPEND_OCCURRED	: 印刷中にサスペンドが発生し、印刷を中止しました。 Device Emulator では発生しません
PRN_HEADTEMP_ERROR	: ヘッド温度エラーが発生しました。 Device Emulator では発生しません
PRN_AUTOLOADING	: オートローディング中です。 Device Emulator では発生しません
PRN_COVER_CLOSE	: スプラッシュカバーが閉じています。 DeviceEmulator では発生しません。
FUNCTION_UNSUPPORT	: 未サポートエラー

補足

本関数は、印刷終了後（印刷データが残っていない状態）に、実行して下さい。

対応情報

機種　　：DT-9800 / IT-9000
ヘッダ　　：PrinterLib.h
ライブラリ　：PrinterLib.lib

3.11 PRNGetStatus

現在のプリンタのエラー状態を取得します。
また、本関数を実行すると PRNGetLastError で取得するエラーへ反映されます。

```
[C++]
DWORD PRNGetStatus()
```

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

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

解説

本関数は、現在のプリンタのエラー状態を取得します。
また、本関数を実行すると PRNGetLastError関数で取得するエラーに反映します。
Device Emulatorでは、PRNOpen関数の実行確認のみを行います。

パラメータ

なし

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_HARDWARE_ERROR	: プリンタのハードウェアが異常です。 Device Emulator では発生しません
PRN_PLATEN_OPEN	: プラテンがオープンしています。 Device Emulator では発生しません
PRN_PAPER_END	: 用紙がありません。 Device Emulator では発生しません
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません
PRN_HEADTEMP_ERROR	: ヘッド温度エラーが発生しました。 Device Emulator では発生しません
PRN_AUTOLOADING	: オートローディング中です。 Device Emulator では発生しません
PRN_COVER_CLOSE	: スプラッシュカバーが閉じています。 DeviceEmulator では発生しません。
FUNCTION_UNSUPPORT	: 未サポートエラー

対応情報

機種	: DT-9800 / IT-9000
ヘッダ	: PrinterLib.h
ライブラリ	: PrinterLib.lib

3.12 PRNGetLastError

最後に実行した関数の実行状態を取得します。

```
[C++]
DWORD PRNGetLastError()
```

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

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

パラメータ

なし

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_NOTFOUND	: マーカが検出できません Device Emulator では発生しません
PRN_NOTCHANG	: 用紙幅を変更できません Device Emulator では発生しません
PRN_PARAMETER_ERROR	: パラメータエラー
PRN_FILE_NOTEXIST	: 指定されたファイルがありません。
PRN_FILEOPEN_ERROR	: 指定されたファイルが開けません。
PRN_FILEFORMAT_ERROR	: 指定されたファイルの形式が不正です。 Device Emulator では発生しません
PRN_HARDWARE_ERROR	: プリンタのハードウェアが異常です。 Device Emulator では発生しません
PRN_PLATEN_OPEN	: プラテンがオープンしています。 Device Emulator では発生しません
PRN_PAPER_END	: 用紙がありません。 Device Emulator では発生しません
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません
PRN_SUSPEND_OCCURRED	: 印刷中にサスペンドが発生し、印刷を中止しました。 Device Emulator では発生しません
PRN_HEADTEMP_ERROR	: ヘッド温度エラーが発生しました。 Device Emulator では発生しません
PRN_AUTOLOADING	: オートローディング中です。 Device Emulator では発生しません
PRN_COVER_CLOSE	: スプラッシュカバーが閉じています。 DeviceEmulator では発生しません。
FUNCTION_UNSUPPORT	: 未サポートエラー

対応情報

機種　　：DT-9800 / IT-9000
ヘッダ　　：PrinterLib.h
ライブラリ　：PrinterLib.lib

3.13 PRNSetPaperWidth

使用する用紙の用紙幅を指定します。

```
[C++]
DWORD PRNSetPaperWidth(
  DWORD dwWidth
)
```

```
[Visual Basic]
Public Shared Function PRNSetPaperWidth( _
  ByVal dwWidth As Int32 _
) As Int32
```

```
[C#]
public static Int32 PRNSetPaperWidth(
  Int32 dwWidth
);
```

パラメータ

dwWidth

印刷に使用する用紙の幅を指定します。

0 : 80mm (82.55mm)
1 : 58mm

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_PARAMETER_ERROR	: パラメータエラー
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません
PRN_NOTCHANGE	: 80mm 幅へ変更できません。
PRN_SUSPEND_OCCURRED	: 印刷中にサスペンドが発生し、印刷を中止しました。 Device Emulator では発生しません
FUNCTION_UNSUPPORT	: 未サポートエラー

補足

58mm 用紙で印刷後、80mm 用紙に変更して印刷すると印刷がかすれる可能性があります。
このため、用紙幅 58mm を設定し印刷した後は、80mmへ変更することはできません。
用紙幅を変更した場合、本関数の呼び出し前に送信され、まだ印字されていないデータは消去されます。また、左右マージンの設定は 0 にリセットされます。
本関数を実行する前には、未印字吐出しコマンドを実行して印刷データを全て印刷して下さい。

対応情報

機種 : DT-9800 / IT-9000
ヘッダ : PrinterLib.h
ライブラリ : PrinterLib.lib

3.14 PRNGetPaperWidth

現在設定されている用紙幅を取得します。

```
[C++]
DWORD PRNGetPaperWidth(
  DWORD *dwWidth
)
```

```
[Visual Basic]
Public Shared Function PRNGetPaperWidth( _
  ByRef dwWidth As Int32 _
) As Int32
```

```
[C#]
public static Int32 PRNGetPaperWidth(
  ref Int32 dwWidth
);
```

パラメータ

dwWidth
現在の用紙幅設定を格納するエリアへのポインタ

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません
PRN_SUSPEND_OCCURRED	: 印刷中にサスペンドが発生し、印刷を中止しました。 Device Emulator では発生しません
FUNCTION_UNSUPPORT	: 未サポートエラー

対応情報

機種	: DT-9800 / IT-9000
ヘッダ	: PrinterLib.h
ライブラリ	: PrinterLib.lib

3.15 PRNSetPrinterProperty

プリンタの各種設定を変更します。

```
[C++]
DWORD PRNSetPrinterProperty(
  DWORD dwPaperType,
  DWORD dwDepth,
  DWORD dwSpeed,
  DWORD dwAutoloading,
  DWORD dwAutoloadingLength,
  DWORD dwPreheat,
  DWORD dwPrintContinuation
)
```

```
[Visual Basic]
Public Shared Function PRNSetPrinterProperty( _
  ByVal dwPaperType As Int32, _
  ByVal dwDepth As Int32, _
  ByVal dwSpeed As Int32, _
  ByVal dwAutoloading As Int32, _
  ByVal dwAutoloadingLength As Int32, _
  ByVal dwPreHeat As Int32, _
  ByVal dwPrintContinuation As Int32 _
) As Int32
```

```
[C#]
public static Int32 PRNSetPrinterProperty(
  Int32 dwPaperType,
  Int32 dwDepth,
  Int32 dwSpeed,
  Int32 dwAutoloading,
  Int32 dwAutoloadingLength,
  Int32 dwPreHeat,
  Int32 dwPrintContinuation
);
```

解説

本関数は、プリンタの各種設定を変更します。
Device Emulatorでは、設定値を内部変数として格納するため、何も動作しません
が、PRNGetPrinterProperty関数を実行することにより、設定値を確認することができます。

パラメータ

dwPaperType
用紙の種類を設定します

■ DT-9800
0 : F-200U9W6
1 : HS360

2 : AFP-235
3 : HG56S
4 : TLC00
5 : Reserve

■ IT-9000
0 : F220VP
1 : HA220AA
2 : AFP-235
3 : HW54S
4 : Reserve
5 : Reserve
6 : ODT60TC-RAK

dwDepth
印字濃度を設定します
1≦dwDepth≦9

dwSpeed
印字速度を設定します
0 : 高速印字
1 : 低速印字(高品質)
2 : グラフィック

dwAutoloading
オートローディングの有効無効を設定します
0 : 無効
1 : 有効

dwAutoloadingLength
オートローディング指定量を設定します。
0Ah≦dwAutoloadingLength≦60h

dwPreheat
プリヒートの有効無効を設定します
0 : 無効
1 : 有効

dwPrintContinuation
エラー時継続印字の有効無効を設定します
0 : 無効
1 : 有効

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_PARAMETER_ERROR	: パラメータエラー
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません

PRN_SUSPEND_OCCURRED　：印刷中にサスペンドが発生し、印刷を中止しました。
Device Emulator では発生しません
FUNCTION_UNSUPPORT　：未サポートエラー

対応情報

機種　：DT-9800 / IT-9000
ヘッダ　：PrinterLib.h
ライブラリ　：PrinterLib.lib

3.16 PRNGetPrinterProperty

プリンタの各種設定状態を取得します。

```
[C++]
DWORD PRNGetPrinterProperty(
  DWORD *dwPaperType,
  DWORD *dwDepth,
  DWORD *dwSpeed,
  DWORD *dwAutoloading,
  DWORD *dwAutoloadingLength,
  DWORD *dwPreheat,
  DWORD *dwPrintContinuation
)
```

```
[Visual Basic]
Public Shared Function PRNGetPrinterProperty( _
  ByRef dwPaperType As Int32, _
  ByRef dwDepth As Int32, _
  ByRef dwSpeed As Int32, _
  ByRef dwAutoloading As Int32, _
  ByRef dwAutoloadingLength As Int32, _
  ByRef dwPreHeat As Int32, _
  ByRef dwPrintContinuation As Int32 _
) As Int32
```

```
[C#]
public static Int32 PRNGetPrinterProperty(
  ref Int32 dwPaperType,
  ref Int32 dwDepth,
  ref Int32 dwSpeed,
  ref Int32 dwAutoloading,
  ref Int32 dwAutoloadingLength,
  ref Int32 dwPreHeat,
  ref Int32 dwPrintContinuation
);
```

パラメータ

dwPaperType
現在の用紙種類設定を格納するエリアへのポインタ

dwDepth
現在の印字濃度設定を格納するエリアへのポインタ

dwSpeed
現在の印字速度設定を格納するエリアへのポインタ

dwAutoloading
現在のオートローディング設定を格納するエリアへのポインタ

dwAutoloadingLength
オートローディング指定量を格納するエリアへのポインタ
0Ah≦dwAutoloadingLength≦60h

dwPreheat
現在のプリヒート設定を格納するエリアへのポインタ

dwPrintContinuation
現在のエラー時継続印字設定を格納するエリアへのポインタ

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタがオープンされていません。
PRN_VDETP_OCCURRED	: VDETP が発生しました。 Device Emulator では発生しません
PRN_SUSPEND_OCCURRED	: 印刷中にサスペンドが発生し、印刷を中止しました。 Device Emulator では発生しません
FUNCTION_UNSUPPORT	: 未サポートエラー

対応情報

機種	: DT-9800 / IT-9000
ヘッダ	: PrinterLib.h
ライブラリ	: PrinterLib.lib

3.17 PRNDecodeBarcode

裏面バーコードのデコード結果を返します。

```
[C++]
DWORD PRNDecodeBarcode(
  DWORD *pdwLength,
  TCHAR *pszData
)
```

```
[Visual Basic]
Public Shared Function PRNDecodeBarcode( _
  ByRef pszData As string _
) As Int32
```

```
[C#]
public static Int32 PRNDecodeBarcode(
  out string pszData
)
```

解説

用紙裏面に印刷されているバーコードを読み取り、その結果を返します。
Device Emulatorでは、PRNOpen関数の実行確認のみを行います。

パラメータ

pdwLength
裏面バーコードのデコード結果の長さを格納するエリアへのポインタ。

pszData
裏面バーコードのデコード結果を格納するエリアへのポインタ。

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタをオープンしていません
PRN_DECODE_ERROR	: デコード失敗
FUNCTION_UNSUPPORT	: 未サポートエラー

補足

本関数は、印刷中に読み取った裏面バーコードの読取データをデコードします。裏面バーコードの読取データは、PRNResetDecoder 関数が実行されてから本関数が実行されるまでのデータを使用します。したがって、裏面バーコード部分を印刷する前に必ず PRNResetDecoder 関数を実行してください。
本関数は、(裏面バーコード部分を含む)帳票印刷終了後(PRNTextOut による未印字吐き出し後)に実行して下さい。

対応情報

機種 : IT-9000

ヘッダ　　：PrinterLib.h
ライブラリ　：PrinterLib.lib

3.18 PRNResetDecoder

裏面バーコードの読み取りデータをリセットします。

```
[C++]
DWORD PRNResetDecoder()
```

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

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

解説

裏面バーコードの読み取りデータをリセットします。
Device Emulatorでは、PRNOpen関数の実行確認のみを行います。

パラメータ

なし

戻り値

PRN_NORMAL	: 正常終了
PRN_NOTOPEN	: プリンタをオープンしていません
FUNCTION_UNSUPPORT	: 未サポートエラー

補足

マーカ検出（PRNCheckMarker 関数）正常終了後の裏面バーコード読取(印刷)開始前に、必ず実行してください。

対応情報

機種	: IT-9000
ヘッダ	: PrinterLib.h
ライブラリ	: PrinterLib.lib

3.19 PRNSetBarcodeType

裏面バーコードの読取方式を設定します。

```
[C++]
DWORD PRNSetBarcodeType(
  DWORD dwBarcodeType
)
```

```
[Visual Basic]
Public Shared Function PRNSetBarcodeType( _
  ByVal dwBarcodeType As Int32 _
) As Int32
```

```
[C#]
public static Int32 PRNSetBarcodeType(
  Int32 dwBarcodeType
)
```

解説

裏面バーコードの読取方式を設定します。
Device Emulatorでは、設定値を内部変数として格納するため、何も動作しません
が、PRNGetBarcodeType関数を実行することにより、設定値を確認することができます。

パラメータ

dwBarcodeType
裏面バーコードの読取方式を設定します。
0 : 無効(デフォルト)
1 : 6 桁読取モード
2 : 15 桁読取モード

戻り値

PRN_NORMAL	: 正常終了
PRN_DRIVER_NOTEXIST	: 他のアプリケーションで既に使用されているか、プリンタドライバが非常駐状態のためプリンタが使用できません Device Emulator では発生しません
PRN_ALREADY_OPEN	: オープン済エラー
PRN_PARAMETER_ERROR	: パラメータエラー
PRN_FEEDKEY_ERROR	: フィードキーでフィード中のため、プリンタが使用できません Device Emulator では発生しません
FUNCTION_UNSUPPORT	: 未サポートエラー

補足

本関数は PRNOpen関数実行前に実行してください。
設定した値は本体をソフトリセットするとデフォルトに戻ります。

対応情報

機種　　：IT-9000
ヘッダ　　：PrinterLib.h
ライブラリ　：PrinterLib.lib

3.20 PRNGetBarcodeType

裏面バーコードの読取方式を取得します。

```
[C++]
DWORD PRNGetBarcodeType(
  DWORD *dwBarcodeType
)
```

```
[Visual Basic]
Public Shared Function PRNGetBarcodeType( _
  ByRef dwBarcodeType As Int32 _
) As Int32
```

```
[C#]
public static Int32 PRNGetBarcodeType(
  ref Int32 dwBarcodeType
)
```

解説

裏面バーコードの読取方式を取得します。

パラメータ

dwBarcodeType

裏面バーコードの読取方式を取得します。取得する値については、PRNSetBarcodeType関数を参照してください。

戻り値

PRN_NORMAL	: 正常終了
PRN_DRIVER_NOTEXIST	: 他のアプリケーションで既に使用されているか、プリンタドライバが非常駐状態のためプリンタが使用できません Device Emulator では発生しません
PRN_PARAMETER_ERROR	: パラメータエラー
PRN_FEEDKEY_ERROR	: フィードキーでフィード中のため、プリンタが使用できません Device Emulator では発生しません
FUNCTION_UNSUPPORT	: 未サポートエラー

対応情報

機種	: IT-9000
ヘッダ	: PrinterLib.h
ライブラリ	: PrinterLib.lib

4. プログラミング上の注意点

4.1 文字コードの入力について

本プリンタには、下記のフォントを搭載しています。

表 4-1

種類	サイズ	概要
ANK	6x7 / 6x12 / 8x16 / 12x24	拡張グラフィック文字コード カタカナ文字コード
漢字	12x12 / 16x16 / 24x24	JIS 第 1 水準／JIS 第 2 水準 （1984年に制定された JIS X9051 に準拠）
OCR-B		

本プリンタで ANK/OCR フォントの文字を印字する場合、文字コードの入力方法として下記の 2 種類があります。下記の指定は、ESC コマンドの「入力文字コード選択」で変更可能です。

- Unicode 指定
- ANK 指定

※ 漢字は、Unicode 指定時は「Unicode」、ANK 指定時は「Shift-JIS」で入力して下さい。

※ Unicode 指定／ANK 指定に関わらず、ANK/OCR フォントを印字する場合は ESC コマンドの「漢字変換指定」で漢字変換を無効にし、漢字を印字する場合は有効にして下さい。

ANK／OCR フォントと Unicode の対応は、次ページ以降の「フォントと文字コードの対応表①～④」を参照して下さい。

A) Unicode 指定

印字する文字(ANK／OCR)を Unicode で入力する場合、ESC コマンドの「入力文字コード選択」で Unicode 指定を選択します。

例 PRNTextOut(4, TEXT("ABC"));

※ 「ä」は、Unicode で「00E4」や「04D3」に割り当てられています。
この場合、TEXT("ä")で ä が「04D3」に変換されると正しく印字されません。
このように、フォントに複数の文字コードが割り当てられている場合は下記の方法で正しく印字できます

◆ 「フォントと文字コードの対応表」の Unicode を直接入力する

◆ ANK 指定で文字コード(0x0020～0x00FF)を入力する

B) ANK 指定

印字する文字(ANK／OCR)を 1 バイトコード(0x0020～0x00FF)で入力する場合、ESC コマンドの「入力文字コード選択」で ANK 指定を選択します。

例 「ä」「漢字」を印刷する場合

```
TCHAR szBuff1[10], szBuff2[10];
szBuff1[0] = 0x84;
szBuff1[1] = 0x0D;
szBuff2[0] = 0x8ABF; // 漢の Shift-JIS コード ※1TCHAR に Shift-JIS コードを入力
szBuff2[1] = 0x8E9A; // 字の Shift-JIS コード ※1TCHAR に Shift-JIS コードを入力
szBuff2[2] = 0x0D;
PRNTextOut(2, szBuff1);
PRNTextOut(3, szBuff2);
```

4.2 フォントテーブル

表の見方

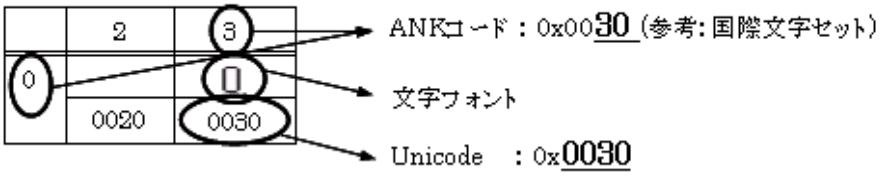


表 4-2 ANK/OCR-B のフォントテーブルを記載します。

	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		0	@	P	`	p	Ç	É	á	▒	└	╨	α	≡
	0020	0030	0040	0050	0060	0070	00C7	00C9	00E1	2591	2514	2568	03B1	2261
1	!	1	A	Q	a	q	ü	æ	í	▓	┴	╤	β	±
	0021	0031	0041	0051	0061	0071	00FC	00E6	00ED	2592	2534	2564	00DF	00B1
2	"	2	B	R	b	r	é	Æ	ó	▓	┬	╥	Γ	≥
	0022	0032	0042	0052	0062	0072	00E9	00C6	00F3	2593	252C	2565	0393	2265
3	#	3	C	S	c	s	â	ô	ú	│	├	╙	π	≤
	0023	0033	0043	0053	0063	0073	00E2	00F4	00FA	2502	251C	2559	03C0	2264
4	$	4	D	T	d	t	ä	ö	ñ	┤	─	╘	Σ	⌠
	0024	0034	0044	0054	0064	0074	00E4	00F6	00F1	2524	2500	2558	03A3	2320
5	%	5	E	U	e	u	à	ò	Ñ	╡	┼	╒	σ	⌡
	0025	0035	0045	0055	0065	0075	00E0	00F2	00D1	2561	253C	2552	03C3	2321
6	&	6	F	V	f	v	å	û	ª	╢	╞	╓	µ	÷
	0026	0036	0046	0056	0066	0076	00E5	00FB	00AA	2562	255E	2553	00B5	00F7
7	'	7	G	W	g	w	ç	ù	º	╖	╟	╫	τ	≈
	0027	0037	0047	0057	0067	0077	00E7	00F9	00BA	2556	255F	256B	03C4	2248
8	(	8	H	X	h	x	ê	ÿ	¿	╕	╚	╪	Φ	°
	0028	0038	0048	0058	0068	0078	00EA	00FF	00BF	2555	255A	256A	03A6	00B0
9	)	9	I	Y	i	y	ë	Ö	⌐	╣	╔	┘	Θ	•
	0029	0039	0049	0059	0069	0079	00EB	00D6	2310	2563	2554	2518	0398	2219
A	*	:	J	Z	j	z	è	Ü	¬	║	╩	┌	Ω	−
	002A	003A	004A	005A	006A	007A	00E8	00DC	00AC	2551	2569	250C	03A9	00AD
B	+	;	K	[	k	{	ï	¢	½	╗	╦	█	δ	√
	002B	003B	004B	005B	006B	007B	00EF	00A2	00BD	2557	2566	2588	03B4	221A
C	,	<	L	¥	l	|	î	£	¼	╝	╠	■	∞	ⁿ
	002C	003C	004C	*	006C	007C	00EE	00A3	00BC	255D	2560	2584	221E	207F
D	−	=	M	]	m	}	ì	¥	¡	╜	═	▌	ø	²
	002D	003D	004D	005D	006D	007D	00EC	*	00A1	255C	2550	258C	00F8	00B2
E	.	>	N	^	n	~	Ä	₧	«	╛	╬	▐	∈	▪
	002E	003E	004E	005E	006E	007E	00C4	20A7	00AB	255B	256C	2590	03B5	FFED
F	/	?	O	_	o	⌂	Å	ƒ	»	┐	╧	▀	∩	€
	002F	003F	004F	005F	006F	007F	00C5	0192	00BB	2510	2567	2580	2229	20AC

	\	§	¨	Ø	¤
	*	00A7	00A8	00D8	00A4

ANK Code 0x005Cと0x009Dについては、日本語版と英語版で Unicode が異なります。

¥	\
005C	E041

Japanese Version

¥	\
00A5	005C

English Version

表 4-3 ANK:文字コード表(カタカナ)

	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		0	@	P	`	p								
	0020	0030	0040	0050	0060	0070	E000	E010	E040	FF70	FF80	FF90	E020	E030
1	!	1	A	Q	a	q			｡	ｱ	ﾁ	ﾑ		
	0021	0031	0041	0051	0061	0071	E001	E011	FF61	FF71	FF81	FF91	E021	E031
2	"	2	B	R	b	r			｢	ｲ	ﾂ	ﾒ		
	0022	0032	0042	0052	0062	0072	E002	E012	FF62	FF72	FF82	FF92	E022	E032
3	#	3	C	S	c	s			｣	ｳ	ﾃ	ﾓ		
	0023	0033	0043	0053	0063	0073	E003	E013	FF63	FF73	FF83	FF93	E023	E033
4	$	4	D	T	d	t			､	ｴ	ﾄ	ﾔ		
	0024	0034	0044	0054	0064	0074	E004	E014	FF64	FF74	FF84	FF94	E024	E034
5	%	5	E	U	e	u			･	ｵ	ﾅ	ﾕ		
	0025	0035	0045	0055	0065	0075	E005	E015	FF65	FF75	FF85	FF95	E025	E035
6	&	6	F	V	f	v			ｦ	ｶ	ﾆ	ﾖ		
	0026	0036	0046	0056	0066	0076	E006	E016	FF66	FF76	FF86	FF96	E026	E036
7	'	7	G	W	g	w			ｧ	ｷ	ﾇ	ﾗ		
	0027	0037	0047	0057	0067	0077	E007	E017	FF67	FF77	FF87	FF97	E027	E037
8	(	8	H	X	h	x			ｨ	ｸ	ﾈ	ﾘ		
	0028	0038	0048	0058	0068	0078	E008	E018	FF68	FF78	FF88	FF98	E028	E038
9	)	9	I	Y	i	y			ｩ	ｹ	ﾉ	ﾙ		
	0029	0039	0049	0059	0069	0079	E009	E019	FF69	FF79	FF89	FF99	E029	E039
A	*	:	J	Z	j	z			ｪ	ｺ	ﾊ	ﾚ		
	002A	003A	004A	005A	006A	007A	E00A	E01A	FF6A	FF7A	FF8A	FF9A	E02A	E03A
B	+	;	K	[	k	{			ｫ	ｻ	ﾋ	ﾛ		
	002B	003B	004B	005B	006B	007B	E00B	E01B	FF6B	FF7B	FF8B	FF9B	E02B	E03B
C	,	<	L	¥	l	|			ｬ	ｼ	ﾌ	ﾜ		
	002C	003C	004C	*	006C	007C	E00C	E01C	FF6C	FF7C	FF8C	FF9C	E02C	E03C
D	-	=	M	]	m	}			ｭ	ｽ	ﾍ	ﾝ	°	
	002D	003D	004D	005D	006D	007D	E00D	E01D	FF6D	FF7D	FF8D	FF9D	E02D	E03D
E	.	>	N	^	n	~			ｮ	ｾ	ﾎ	ﾞ		
	002E	003E	004E	005E	006E	007E	E00E	E01E	FF6E	FF7E	FF8E	FF9E	E02E	E03E
F	/	?	O	_	o				ｯ	ｿ	ﾏ	ﾟ		
	002F	003F	004F	005F	006F	007F	E00F	E01F	FF6F	FF7F	FF8F	FF9F	E02F	E03F

	\	§	¨	Ø	¤
	*	00A7	00A8	00D8	00A4

ANK Code 0x005C については、日本語版と英語版で Unicode が異なります。

¥	\
005C	E041

Japanese Version

¥	\
00A5	005C

English Version

表 4-4　OCR-B サイズ I

	2	3	4	5	6	7	8
0		0	@	P	`	p	ı
	0020	0030	0040	0050	0060	0070	E050
1	!	1	A	Q	a	q	
	0021	0031	0041	0051	0061	0071	
2	"	2	B	R	b	r	
	0022	0032	0042	0052	0062	0072	
3	#	3	C	S	c	s	
	0023	0033	0043	0053	0063	0073	
4	$	4	D	T	d	t	
	0024	0034	0044	0054	0064	0074	
5	%	5	E	U	e	u	
	0025	0035	0045	0055	0065	0075	
6	&	6	F	V	f	v	
	0026	0036	0046	0056	0066	0076	
7	'	7	G	W	g	w	
	0027	0037	0047	0057	0067	0077	
8	(	8	H	X	h	x	
	0028	0038	0048	0058	0068	0078	
9	)	9	I	Y	i	y	
	0029	0039	0049	0059	0069	0079	
A	*	:	J	Z	j	z	
	002A	003A	004A	005A	006A	007A	
B	+	;	K	[	k	{	
	002B	003B	004B	005B	006B	007B	
C	,	<	L	¥	l	¦	
	002C	003C	004C	※	006C	007C	
D	-	=	M	]	m	}	
	002D	003D	004D	005D	006D	007D	
E	.	>	N	^	n	¤	
	002E	003E	004E	005E	006E	00A4	
F	/	?	O	_	o	£	
	002F	003F	004F	005F	006F	00A3	

表 4-5　OCR-B サイズⅣ

	2	3	4	5	6	7	8
0		0	@	P	`	p	
	0020	0030	0040	0050	0060	0070	
1	!	1	A	Q	a	q	
	0021	0031	0041	0051	0061	0071	
2	"	2	B	R	b	r	
	0022	0032	0042	0052	0062	0072	
3	#	3	C	S	c	s	
	0023	0033	0043	0053	0063	0073	
4	$	4	D	T	d	t	
	0024	0034	0044	0054	0064	0074	
5	%	5	E	U	e	u	
	0025	0035	0045	0055	0065	0075	
6	&	6	F	V	f	v	
	0026	0036	0046	0056	0066	0076	
7	'	7	G	W	g	w	
	0027	0037	0047	0057	0067	0077	
8	(	8	H	X	h	x	
	0028	0038	0048	0058	0068	0078	
9	)	9	I	Y	i	y	
	0029	0039	0049	0059	0069	0079	
A	*	:	J	Z	j	z	
	002A	003A	004A	005A	006A	007A	
B	+	;	K	[	k	{	
	002B	003B	004B	005B	006B	007B	
C	,	<	L	¥	l	¦	
	002C	003C	004C	※	006C	007C	
D	-	=	M	]	m	}	
	002D	003D	004D	005D	006D	007D	
E	.	>	N	^	n	¤	
	002E	003E	004E	005E	006E	00A4	
F	/	?	O	_	o	£	
	002F	003F	004F	005F	006F	00A3	

国際文字セット (入力コードが ANK 指定時に有効)

文字コードが同じでも、国によってフォントが異なるものがあります。そのため、ANK コードの一部を置き換えて各国に対応します。下表は国際文字選択機能によって置き換えられる文字と国の関係を表しています。横軸の数字は次のように国を意味します。

表 4-6

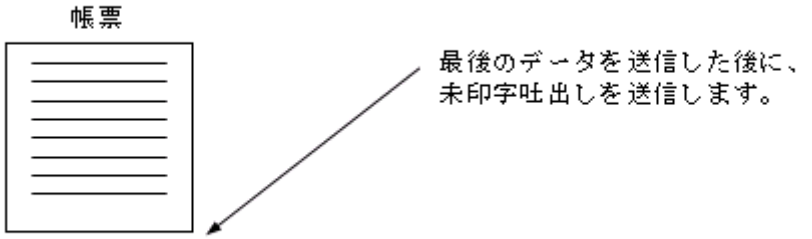
0.	アメリカ	1.	フランス	2.	ドイツ	3.	イギリス
4.	デンマーク	5.	スウェーデン	6.	イタリア	7.	スペイン
8.	日本						

表 4-7

	0	1	2	3	4	5	6	7	8
23h	#	#	#	£	#	#	#	₧	#
24h	$	$	$	$	$	¤	$	$	$
40h	@	à	§	@	@	É	@	@	@
5Bh	[	°	Ä	[	Æ	Ä	°	¡	[
5Ch	\	ç	Ö	\	ø	Ö	\	Ñ	¥
5Dh	]	§	Ü	]	Å	Å	é	¿	]
5Eh	^	^	^	^	^	Ü	^	^	^
60h	`	`	`	`	`	é	ù	`	`
7Bh	{	é	ä	{	æ	ä	à	¨	{
7Ch	|	ù	ö	|	ø	ö	ò	ñ	|
7Dh	}	è	ü	}	å	å	è	}	}
7Eh	~	¨	β	~	~	ü	ì	~	~

4.3 印刷プログラムの作成について

本機のプリンタは、印字データをバッファにスプールしながら印字を行っています。
したがって、バッファにたまった印字データを吐出すために、帳票の最後で「未印字吐出し:ESC E」(印字継続コマンド)コマンドを送信します。
このコマンドを送信しないと、印字中に発生したエラーを取りこぼす場合があります。帳票の最後で必ずこのコマンドを送信して下さい。



4.4 裏面バーコードの読み取りについて

裏面バーコードの読取手順について

裏面バーコード読取時は、下記の手順で印刷・読取を行う必要があります。

1. プリンタの電源を入れる（PRNOpen を実行する）前に裏面バーコード読取方式（6 桁 or15 桁）を設定する
2. プリンタの電源を入れる
3. マーカ検出で位置合わせを行う
4. 前回の裏面バーコード読取データをリセットする
5. 印刷する
6. 印刷の完了を待つ
7. 裏面バーコードを読み取る

位置合せについて

印刷データを印刷する前に必ずマーカ検出を行って印字位置(裏面バーコード読取開始位置)を合わせる必要があります。

印刷の長さについて

印刷しながらバーコードを読み取るために、バーコードの長さよりも長く印刷する必要があります。また、最後のストップバーを認識するためにバーコード長(バーコード・マーカ間スペース＋バーコード長)よりもバー3 本分（分解能が 1.0mm の場合は 3.0mm）程長く印刷する必要があります。

例：15 桁のバーコード、分解能が 1.0mm の場合

印刷する長さ = 3.0＋34.0 + 3.0 = 40.0mm

印刷データについて

裏面バーコードを読み取る場合は、文字フォント、ビットイメージ、ビットマップ、及びフィードコマンド(ESC コマンド)を組み合わせずに単独で印刷する必要があります。また印刷完了待ちは、全ての印刷データを送信した後に行う必要があります。

例 1：文字フォントのみの印刷で 40mm のデータの印刷実行後に印刷完了待ち

例 2：縦 40mm のビットマップの印刷実行後に印刷完了待ち

印刷中のエラーについて

印刷が最後まで正常に終了した場合に裏面バーコードを読み取ることが出来ます。したがって、印刷時はエラー時継続設定を無効に設定し、プリンタのエラー(VDETP、ヘッド温度エラーも含む) が発生した場合は、再度位置合わせからやり直す必要があります。

4.5 サンプルプログラム

エラー時非継続の場合

```
TCHAR ESCE[2] = {0x1B, 'E'};          // 未印字データの吐き出しコマンド
TCHAR ESCR[3] = {0x1B, 'R', 0};       // エラー時印字継続指定（非継続設定）
TCHAR CAN     = 0x18;

#define IMAGE_HEIGHT  96
#define IMAGE_WIDTH   72
#define FEED_LENGTH   0

// テキスト印刷
void PrintText()
{
    if (PRNOpen() != PRN_NORMAL)
    {
        MessageBox(NULL, L"Open error", L"Error", MB_OK);
        return;
    }

    if (PRNTextOut(3, ESCR) != PRN_NORMAL)
    {
        CheckError();
        PRNClose();
        return;
    }

    if (PRNTextOut(11, L"0123456789¥r") != PRN_NORMAL)
    {
        CheckError();
        PRNClose();
        return;
    }

    if (PRNTextOut(2, ESCE) != PRN_NORMAL)
    {
        CheckError();
        PRNClose();
        return;
    }

    PRNClose();
}

// イメージ印刷（ビットマップ印刷、バーコード印刷、画面印刷等も同等）
void PrintImage()
{
    HANDLE hFile;
    DWORD dwReadNum, dwSize;
    BYTE *pbData;
```

```
    hFile = CreateFile(L"¥¥Image.bin", GENERIC_READ | GENERIC_WRITE, 0,
NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
    if (hFile == INVALID_HANDLE_VALUE)
    {
        MessageBox(NULL, L"File not found", L"Error", MB_OK);
        return;
    }

    dwSize = IMAGE_WIDTH * IMAGE_HEIGHT;

    pbData = (BYTE *)LocalAlloc(0, dwSize);
    if (pbData == NULL)
    {
        MessageBox(NULL, L"Failed to get memory", L"Error", MB_OK);
        CloseHandle(hFile);
        return;
    }

    if (!ReadFile(hFile, pbData, dwSize, &dwReadNum, NULL))
    {
        MessageBox(NULL, L"Failed to read file", L"Error", MB_OK);
        LocalFree(pbData);
        CloseHandle(hFile);
        return;
    }

    CloseHandle(hFile);

    if (PRNOpen() != PRN_NORMAL)
    {
        MessageBox(NULL, L"Open error", L"Error", MB_OK);
        LocalFree(pbData);
        return;
    }

    if (PRNTextOut(3, ESCR) != PRN_NORMAL)
    {
        CheckError();
        PRNClose();
        LocalFree(pbData);
        return;
    }

    if (PRNImageOut(IMAGE_WIDTH, IMAGE_HEIGHT, FEED_LENGTH, pbData) != PRN_NORMAL)
    {
        CheckError();
        PRNClose();
        LocalFree(pbData);
        return;
    }

    if (PRNTextOut(2, ESCE) != PRN_NORMAL)
    {
```

```
        CheckError();
        PRNClose();
        LocalFree(pbData);
        return;
    }

    PRNClose();

    LocalFree(pbData);
}

void CheckError()
{
    DWORD dwRet;
    TCHAR szBuff[128];

    dwRet = PRNGetLastError();

    switch (dwRet) {
    case PRN_NOTOPEN:
        wcscpy(szBuff, L"Not open.¥r");
        break;

    case PRN_NOTFOUND:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"Not found.¥r");
        break;

    case PRN_NOTCHANGE:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"Not change.¥r");
        break;

    case PRN_FILE_NOTEXIST:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"File not exist.¥r");
        break;

    case PRN_FILEFORMAT_ERROR:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"File format error.¥r");
        break;

    case PRN_FILEOPEN_ERROR:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"File open error.¥r");
        break;

    case PRN_PARAMETER_ERROR:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"Parameter error.¥r");
        break;
```

```
    case PRN_HARDWARE_ERROR:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"Hardware error.¥r");
        break;

    case PRN_PLATEN_OPEN:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"Platen is opened.¥r");
        break;

    case PRN_PAPER_END:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"Paper end.¥r");
        break;

    case PRN_VDETP_OCCURRED:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"VDETP occurred.¥r");
        break;

    case PRN_SUSPEND_OCCURRED:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"Suspend occurred.¥r");
        break;

    case PRN_HEADTEMP_ERROR:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"HeadTemp error.¥r");
        break;

    case PRN_COVER_CLOSE:
        PRNTextOut(1, &CAN);
        wcscpy(szBuff, L"Splash cover is closed.¥r");
        break;

    }

    MessageBox(NULL, szBuff, L"Error", MB_OK);

    return;
}
```

エラー時継続の場合

```
TCHAR ESCE[2] = {0x1B, 'E'};        // 未印字データの吐き出しコマンド
TCHAR ESCR[3] = {0x1B, 'R', 1};     // エラー時印字継続指定（継続設定）
TCHAR CAN     = 0x18;

#define IMAGE_HEIGHT  96
#define IMAGE_WIDTH   72
#define FEED_LENGTH   0

void PrintText()
{
    if (PRNOpen() != PRN_NORMAL)
    {
        MessageBox(NULL, L"Open error", L"Error", MB_OK);
        return;
    }

    if (PRNTextOut(3, ESCR) != PRN_NORMAL)
    {
        if (CheckError() != TRUE)
        {
            PRNClose();
            return;
        }
    }

    if (PRNTextOut(11, L"0123456789¥r") != PRN_NORMAL)
    {
        if (CheckError() != TRUE)
        {
            PRNClose();
            return;
        }
    }

    if (PRNTextOut(2, ESCE) != PRN_NORMAL)
    {
        if (CheckError() != TRUE)
        {
            PRNClose();
            return;
        }
    }

    PRNClose();
}

void PrintImage()
{
    HANDLE hFile;
```

```
    DWORD dwReadNum, dwSize;
    BYTE *pbData;

    hFile = CreateFile(L"¥¥Image.bin", GENERIC_READ | GENERIC_WRITE, 0,
NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
    if (hFile == INVALID_HANDLE_VALUE)
    {
        MessageBox(NULL, L"File not found", L"Error", MB_OK);
        return;
    }

    dwSize = IMAGE_WIDTH * IMAGE_HEIGHT;

    pbData = (BYTE *)LocalAlloc(0, dwSize);
    if (pbData == NULL)
    {
        MessageBox(NULL, L"Failed to get memory", L"Error", MB_OK);
        CloseHandle(hFile);
        return;
    }

    if (!ReadFile(hFile, pbData, dwSize, &dwReadNum, NULL))
    {
        MessageBox(NULL, L"Failed to read file", L"Error", MB_OK);
        LocalFree(pbData);
        CloseHandle(hFile);
        return;
    }

    CloseHandle(hFile);

    if (PRNOpen() != PRN_NORMAL)
    {
        MessageBox(NULL, L"Open error", L"Error", MB_OK);
        LocalFree(pbData);
        return;
    }

    if (PRNTextOut(3, ESCR) != PRN_NORMAL)
    {
        if (CheckError() != TRUE)
        {
            PRNClose();
                   LocalFree(pbData);
            return;
        }
    }

    if (PRNImageOut(IMAGE_WIDTH, IMAGE_HEIGHT, FEED_LENGTH, pbData) != PRN_NORMAL)
    {
        if (CheckError() != TRUE)
        {
            PRNClose();
```

```
            LocalFree(pbData);
            return;
        }
    }

    if (PRNTextOut(2, ESCE) != PRN_NORMAL)
    {
        if (CheckError() != TRUE)
        {
            PRNClose();
            LocalFree(pbData);
            return;
        }
    }

    PRNClose();

    LocalFree(pbData);
}

BOOL CheckError()
{
    DWORD dwRet;
    int iRet;
    BOOL bContinuation;
    TCHAR szBuff[128];

    while (1) {
        dwRet = PRNGetLastError();

        switch (dwRet) {
        case PRN_NOTOPEN:
            wcscpy(szBuff, L"Not open.¥r");
            bContinuation = FALSE;
            break;

        case PRN_NOTFOUND:
            wcscpy(szBuff, L"Not found.¥r");
            bContinuation = FALSE;
            break;
        case PRN_NOTCHANGE:
            wcscpy(szBuff, L"Not change.¥r");
            bContinuation = FALSE;
            break;
        case PRN_FILE_NOTEXIST:
            wcscpy(szBuff, L"File not exist.¥r");
            bContinuation = FALSE;
            break;
        case PRN_FILEFORMAT_ERROR:
            wcscpy(szBuff, L"File format error.¥r");
            bContinuation = FALSE;
            break;
        case PRN_FILEOPEN_ERROR:
```

```
            wcscpy(szBuff, L"File open error.¥r");
            bContinuation = FALSE;
            break;
        case PRN_PARAMETER_ERROR:
            wcscpy(szBuff, L"Parameter error.¥r");
            bContinuation = FALSE;
            break;
        case PRN_HARDWARE_ERROR:
            wcscpy(szBuff, L"Hardware error.¥r");
            bContinuation = FALSE;
            break;
        case PRN_PLATEN_OPEN:
            wcscpy(szBuff, L"Platen is opened.¥r");
            bContinuation = FALSE;
            break;
        case PRN_SUSPEND_OCCURRED:
            wcscpy(szBuff, L"Suspend occurred.¥r");
            bContinuation = FALSE;
            break;

        case PRN_PAPER_END:
            wcscpy(szBuff, L"Paper end.¥r");
            bContinuation = TRUE;
            break;
        case PRN_VDETP_OCCURRED:
            wcscpy(szBuff, L"VDETP occurred.¥r");
            bContinuation = TRUE;
            break;
        case PRN_HEADTEMP_ERROR:
            wcscpy(szBuff, L"HeadTemp error.¥r");
            bContinuation = TRUE;
            break;
        case PRN_COVER_CLOSE:
            wcscpy(szBuff, L"Splash cover is closed.¥r");
            bContinuation = TRUE;
            break;
        }

        if (bContinuation != TRUE)
        {
            if (dwRet != PRN_NOTOPEN)
            {
                PRNTextOut(1, &CAN);
            }
            wcscat(szBuff, L"Printing was stopped, because Printer can't
continue.¥r");
            MessageBox(NULL, szBuff, L"Error", MB_OK | MB_ICONERROR);
            return FALSE;
        }
        else
        {
            wcscat(szBuff, L"Do you want to continue ?¥r");
            iRet = MessageBox(NULL, szBuff, L"Error", MB_RETRYCANCEL);
```

```
            if (iRet == IDRETRY)
            {
                if (PRNTextOut(2, ESCE) == PRN_NORMAL)
                {
                    return TRUE;
                }
            }
            else
            {
                PRNTextOut(1, &CAN);
                return FALSE;
            }
        }
    }
}
```

5. ESCコマンド

5.1 ESCコマンド一覧

コマンド	名称	初期値
CR、LF、FF	印刷・紙送り	－
BS	一文字削除	－
CAN	バッファクリア	－
ESC B n1 n2	n(mm)紙送り	－
ESC b n1 n2	n(dot)紙送り	－
ESC r n	右マージン指定	0x00
ESC s n	左マージン指定	0x00
ESC A n	改行ピッチ指定	0x08
ESC W n	文字間ピッチ指定	0x00
ESC J n	自動改行指定	0x01(有効)
ESC Y n	入力文字コード選択	0x00(Unicode)
ESC C n	漢字変換指定	0x01 (有効)
ESC u n	国際文字選択	0x08 (日本)
ESC t n	文字コード表選択	0x01 (カタカナ)
ESC F n	文字フォント指定	0x02 (16dot font)
ESC S n1 n2	文字サイズ指定	n1=0x00 (1 倍) n2=0x00 (1 倍)
ESC O n	文字装飾指定／解除	0x00 (解除)
ESC L n	横印字(回転)指定／解除	0x00 (解除)
ESC m n Data	外字定義(登録)	－
ESC K n1 n2 Data	スタンプ定義(登録)	縦横 96dot の白いスタンプ
ESC G	スタンプ印刷	－
ESC Q n	イメージデータ合成	0x00 (解除)
ESC I n	ビットイメージサイズ指定	0x00 (1 倍)
ESC V n	印字速度指定	0x01 (低速(高品位))
ESC D n	印字濃度指定	0x05 (標準)
ESC P n	用紙指定	0x00 (F200)
ESC H n	プリヒート指定	0x00 (無効)
ESC p n	ヘッド温度保持指定	0x00 (無効) 0x01～0x0C(時間設定:5 分単位) 0xFF (常時有効)
ESC T n	オートローディング指定	0x00 (無効)
ESC R n	エラー時継続印字指定	0x00 (無効)
ESC M	マーカ検出	－
ESC v n1 n2	マーカ検出モード	n1=0x00 (終了検出) n2=0x00 (0mm)
ESC E	未印字データの吐出し(印字継続)	－
ESC h n	バーコードの高さ指定	12(mm)
ESC c n	バーコードの C/D 指定	0x01(付加する)
ESC f n	バーコードのデータ印刷指定	0x01(8x16 フォント)
ESC e n	バーコード印刷時のマージン指定	0x00

ESC d n	バーコードの印刷方向指定	0x00(縦)
ESC g n1 n2 Data	バーコード印刷	－
ESC Z n	初期化	－

5.2 ESCコマンド詳細

- CR、LF、FF
  【名称】　印刷・紙送り
  【コード】　0x0D、0x0A、0x0C
  【機能】　印刷後、紙送り動作をします
  【動作】　ラインバッファにデータがある場合は、ラインバッファ内のデータを印刷し、改行ピッチ分の紙送りを行います。ラインバッファにデータが無い場合は、文字の高さ＋改行ピッチ分の紙送りを行います。

- BS
  【名称】　一文字削除
  【コード】　0x08
  【機能】　印刷データの最終文字を削除します。
  【動作】　ラインバッファにデータがある場合は、ラインバッファ内の最終データを削除します。ラインバッファにデータが無い場合は、何も処理しません。

- CAN
  【名称】　バッファクリア
  【コード】　0x18
  【機能】　印刷データを全て消去します。
  【動作】　CAN コマンド前に送信されていて、まだ印刷されていないデータを全て消去します。必ず送信データの先頭に CAN コマンドを入れて、PRNTextOut で送信して下さい。

- ESC B n1 n2
  【名称】　n mm 順方向、または逆方向紙送り
  【コード】　0x1B 0x42 n1 n2
  【定義域】　n1=0x00（順方向）、n1=0x01（逆方向）
  n1=0x00 の場合、0x00≦n2≦0xFF(mm)
  n1=0x01 の場合、0x00≦n2≦0x08(mm)
  ※逆方向紙送りは、IT-9000 のみ有効
  【機能】　印刷後、順方向または逆方向に設定量分(mm 単位)の紙送り動作をします
  【動作】　ラインバッファにデータがある場合は、ラインバッファ内のデータを印刷し、n2(mm)分の紙送りを行います。ラインバッファにデータが無い場合は、n2(mm)分の紙送りを行います。
  ※本コマンドは、単独で送信してください。

```
TCHAR ESCB[4] = {0x1B, 0x42, 0x00, 0x01};
PRNTextOut(4, ESCB);
```

- ESC b n1 n2

【名称】 n dot 順方向、または逆方向紙送り
【コード】 0x1B 0x62 n1 n2
【定義域】 n1=0x00(順方向)、n1=0x01(逆方向)
n1=0x00 の場合、0x00≦n2≦0xFF(dot)
n1=0x01 の場合、0x00≦n2≦0x40(dot)
※逆方向紙送りは、IT-9000 のみ有効
【機能】 印刷後、順方向または逆方向に設定量分(dot 単位)の紙送り動作をします
【動作】 ラインバッファにデータがある場合は、ラインバッファ内のデータを印刷し、n2(dot)分の紙送りを行います。ラインバッファにデータが無い場合は、n2(dot)分の紙送りを行います。
※本コマンドは、単独で送信してください。

```
TCHAR ESCb[4] = {0x1B, 0x62, 0x00, 0x01};
PRNTextOut(4, ESCb);
```

- ESC r n

【名称】 右マージン指定
【コード】 0x1B 0x72 n
【定義域】 0x00≦n≦0x3C (用紙幅 80mm(82.55mm)の場合)
0x00≦n≦0x24 (用紙幅 58mm の場合)
【機能】 右マージンを n×8 ドットに設定します。
【動作】 印刷有効範囲に対し、右側余白を 8 ドット単位で設定します。用紙幅 80mm(82.55mm)において n>0x3C の場合、用紙幅 58mm において n>0x24 の場合は無視されます。本コマンドは行頭での設定のみ有効であり、それ以外での設定は次行から有効になります。
また、左右マージン指定が以下の条件を満たさない場合は、無視されます。
プリンタ総ドット数(576 or 384) － (左マージン+右マージン) > 96
※PRNSetPaperWidth で用紙幅を切り替えた場合、左右マージンは 0 にリセットされます。

- ESC s n

【名称】 左マージン指定
【コード】 0x1B 0x73 n
【定義域】 0x00≦n≦0x3C (用紙幅 80mm(82.55mm)の場合)
0x00≦n≦0x24 (用紙幅 58mm の場合)
【機能】 左マージンを n×8 ドットに設定します。
【動作】 印刷有効範囲に対し、左側余白を 8 ドット単位で設定します。用紙幅 80mm(82.55mm)において n>0x3C の場合、用紙幅 58mm において n>0x24 の場合は無視されます。本コマンドは行頭での設定のみ有効であり、それ以外での設定は次行から有効になります。
また、左右マージン指定が以下の条件を満たさない場合は、無視されます。
プリンタ総ドット数(576 or 384) － (左マージン+右マージン) > 96
※PRNSetPaperWidth で用紙幅を切り替えた場合、左右マージンは 0 にリセットされます。

- ESC A n

  【名称】　改行ピッチ指定
  【コード】　0x1B 0x41 n
  【定義域】　0x00≦n≦0x60
  【機能】　改行ピッチを n ドットラインに設定します。
  改行ピッチは行頭での設定のみ有効であり、それ以外での設定は次行から有効になります。

- ESC W n

  【名称】　文字間ピッチ指定
  【コード】　0x1B 0x57 n
  【定義域】　0x00≦n≦0x60
  【機能】　文字間ピッチを n ドットに設定します。
  【動作】　文字の右側にスペースを設定します。文字印刷後の文字間スペースが印刷有効範囲を超えた場合、超えたスペース分は無視されます。
  本文字間は、半角(ANK)が基準となり全角(漢字)では設定値の 2 倍の文字間になります。ただし、文字サイズ指定で文字サイズを変更しても文字間ピッチは設定値通りとなります。文字間ピッチは行頭での設定のみ有効であり、それ以外での設定は次行から有効になります。

- ESC J n

  【名称】　自動改行指定
  【コード】　0x1B 0x4A n
  【定義域】　n=0x00(無効)、0x01(有効)
  【機能】　自動改行を有効とするか、無効とするかを指定します。
  【動作】　自動改行が有効であれば、ラインバッファがフルになった場合に自動改行を行います。
  自動改行が無効であれば、ラインバッファがフルになっても自動改行を行いません。この場合、印刷範囲を超えた受信データは切り捨てます。

- ESC Y n

  【名称】　入力文字コード選択
  【コード】　0x1B 0x59 n
  【定義域】　n=0x00(Unicode)、0x01(ANK 0x20～0xFF)
  【機能】　PRNTextOut にて入力する文字コードを選択します。
  例「Ç」を入力する場合(文字を印刷する前に、漢字変換を無効にする)
  Unicode ---> 0xC7　ANK ---> 0x80(文字コード表：拡張グラフィック選択時)
  ※ANK 指定で漢字を印刷する場合は、1TCHAR に対して漢字の Shift-JIS コードを入力してください。

```
TCHAR szBuff[10];
 szBuff[0] = 0x8ABF; //漢の Shift-JIS コード
```

- ESC C n
  【名称】　漢字変換指定
  【コード】　0x1B 0x43 n
  【定義域】　n=0x00(無効)、0x01(有効)
  【機能】　印刷において、指定されたデータ(Unicode)を Shift-JIS へ変換し、漢字変換を行います。漢字以外のフォントや 0x80 以降のキャラクタを印刷する場合は、無効を指定して下さい。

- ESC u n
  【名称】　国際文字選択
  【コード】　0x1B 0x75 n
  【定義域】　0x00≦n≦0x08
  【機能】　n の値により、下記の文字セットを選択します。
  0:USA　1:フランス　2:ドイツ　3:イギリス　4:デンマーク　5:スウェーデン
  6:イタリア　7:スペイン　8:日本

- ESC t n
  【名称】　文字コード表選択
  【コード】　0x1B 0x74 n
  【定義域】　n=0x00(拡張グラフィック)、0x01(カタカナ)
  【機能】　n の値により、0x80 以降の文字コード表を選択します。

- ESC F n
  【名称】　文字フォント指定
  【コード】　0x1B 0x46 n
  【定義域】　0x00≦n≦0x05
  【機能】　n の値により、文字フォントを指定します。

n	ANK	漢字	OCR-B
0	6×7 ドット		
1	6×12 ドット	12×12 ドット	
2	8×16 ドット	16×16 ドット	
3	12×24 ドット	24×24 ドット	
4			16×30 ドット
5			24×45 ドット

- ESC S n1 n2

【名称】　文字サイズ指定
【コード】　0x1B 0x53 n1 n2
【定義域】　0x00≦n1≦0x04 (横サイズ)
0x00≦n2≦0x04 (縦サイズ)
【機能】　n1 と n2 の組合せで、文字サイズ(倍率)を変更します。

n1	横サイズ	n2	縦サイズ
0x00	1.0 倍	0x00	1.0 倍
0x01	1.5 倍	0x01	1.5 倍
0x02	2.0 倍	0x02	2.0 倍
0x03	3.0 倍	0x03	3.0 倍
0x04	4.0 倍	0x04	4.0 倍

【動作】　OCR-B フォントを除く ANK(半角文字)、漢字、外字に対して文字サイズの変更が可能です。1 行中に各種文字フォント・文字サイズ・文字修飾を混在して印刷できますが、その場合の行幅はその行に含まれる最大の文字の高さになり、印刷位置は下揃えとなります。

- ESC O n

【名称】　文字修飾指定／解除
【コード】　0x1B 0x4F n
【定義域】　0x00≦n≦0x0F
【機能】　n の値により、OCR-B フォントを除く ANK、漢字に対して文字修飾を指定します。(ただしスムージングは、24dot フォント指定時のみ有効です.)

n のビット	機能	設定	
bit 0	スムージング	0:無効	1:有効
bit 1	強調	0:無効	1:有効
bit 2	反転	0:無効	1:有効
bit 3	淡調	0:無効	1:有効
bit 4	予約		
bit 5			
bit 6			
bit 7			

- ESC L n

【名称】　横印字(回転)指定／解除
【コード】　0x1B 0x4C n
【定義域】　0x00≦n≦0x03
【機能】　n の値により、OCR-B･6×7 フォントを除く ANK、漢字に対して字をその場で回転します。

n	回転指定
0x00	解除
0x01	90° 回転(90° 右回転)
0x02	180° 回転(180° 右回転)
0x03	270° 回転(270° 右回転)

【動作】　1 行中に 1 種類の指定のみ有効で、同一行中に異なった回転は混在できません。したがって、本コマンドは行頭での指定のみ有効で、行中での指定は無視されます。

- ESC m n Data

【名称】　外字定義(登録)
【コード】　0x1B 0x6D n Data
【定義域】　0xE100≦n≦0xE17F
【機能】　外字データを定義します。
【動作】　文字フォント指定(ESC F n)で選択されたフォントに対して、登録します。登録に必要なデータサイズは、24dot 系フォントの場合は TCHAR 型で 72、16dot 系フォントの場合は TCHAR 型で 32、12dot 系フォントの場合は TCHAR 型で 24 となります。
TCHAR 型のデータに1つに対し、1バイト分の外字データを入れて下さい。
例) 12dot フォントの場合

```
TCHAR UDC[72+3], Data[24] ={外字データ...};
TCHAR CAN = 0x0018;
UDC[0] = 0x1B;
UDC[1] = 0x6D;
UDC[2] = 0xE100;
memcpy(&UDC[3], Data, 24);
If (PRNGetStatus() != PRN_NORMAL)     } ※
    PRNTextOut(1, &CAN);
PRNTextOut(3+24, UDC);
```

同一文字コードに 24dot 系フォント、16dot 系フォント、12dot 系フォントをそれぞれ定義する場合は、文字フォント指定と組み合せて別々に登録する必要があります。
未定義の文字コードにアクセスした場合は、全角スペースが印刷されます。

※プリンタドライバ内のバッファがフルになった場合、ESC コマンドの解析を停止します。エラーが発生している場合はバッファ内のデータを印刷できないため、データをクリアするために CAN を送信しています。
外字登録 ESC コマンドは分割せずに送信してください。
外字登録は必ず Unicode(0xE100～0xE17F)で指定して下さい。

- ESC K n1 n2 Data

【名称】　スタンプ定義(登録)

【コード】　0x1B 0x4B n1 n2 Data

【定義域】　横方向指定(8ドット単位)
0x01≦n1≦0x48(用紙幅 80mm(82.55mm)の場合)
0x01≦n1≦0x30(用紙幅 58mm の場合)
縦方向指定(1ドットライン単位)
0x01≦n2≦0x60

【機能】　スタンプデータを定義します。

【動作】　横方向のサイズはスタンプデータの横ドット数÷8 とし、8 ドット単位で指定します。
Data には 1 バイト単位でスタンプデータを指定します。
既に登録されているデータがある場合は、そのデータに対して上書きします。
スタンプ登録時は、スタンプデータ1バイトを TCHAR 型のデータ1つに入力して下さい。
例) 縦横 24dot スタンプの場合

```
TCHAR CAN = 0x0018;
TCHAR Stamp[6912+4], Data[72] = {スタンプデータ...};
Stamp[0] = 0x1B;
Stamp[1] = 0x4B;
Stamp[2] = 0x03;
Stamp[3] = 0x24;
memcpy(&Stamp[4], Data, 72);
If (PRNGetStatus() != PRN_NORMAL)     } ※
    PRNTextOut(1, &CAN);
PRNTextOut(4+72, Stamp);
```

プリンタドライバ内のバッファがフルになった場合、ESC コマンドの解析を停止します。エラーが発生している場合はバッファ内のデータを印刷できないため、データをクリアするために CAN を送信しています。
スタンプ登録 ESC コマンドは分割せずに送信してください。

- ESC G

【名称】　スタンプ印刷

【コード】　0x1B 0x47

【機能】　スタンプデータを印刷データとして展開します。
スタンプを印刷する場合は、ESC G の後に CR や LF を送信して下さい。
例　TCHAR cmd[] = {0x1B, 0x47, 0x0D};

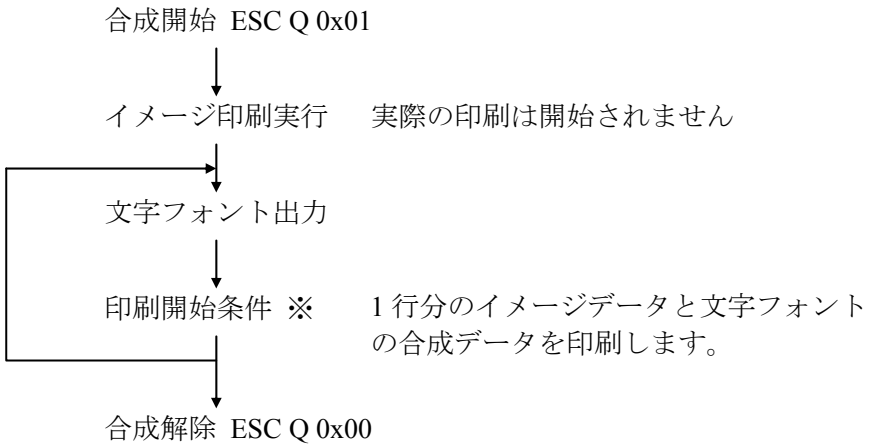
```
TCHAR cmd2[] = {0x20, 0x20, 0x20, 0x20, 0x1B, 0x47, 0x0D};
PRNTextOut(3, cmd);
PRNTextOut(7, cmd2); /* スペースでスタンプの印刷位置を右へずらして印刷 */
```

注意:スタンプの大きさ(横幅)と左右マージン設定の関係が下記のような場合は、スタンプを印刷することはできません。

スタンプの横幅 ＞ 印刷領域(576dot or 384dot) － (左マージン＋右マージン)

- ESC Q n

【名称】　イメージデータ合成
【コード】　0x1B 0x51 n
【定義域】　n=0x00(解除)、0x01(合成開始)
【機能】　イメージデータとテキストデータの合成の開始、解除を指定します。
【動作】　イメージデータとテキストデータの合成手順は、下記の通りです。



※この時点でビットイメージ印刷が実行された場合、実行されたビットイメージデータがセットされます。
イメージ印刷で確保した領域は、合成解除後に開放して下さい。合成解除前に開放するとアプリケーションがフリーズしてしまいます。

- ESC I n

【名称】　ビットイメージサイズ指定
【コード】　0x1B 0x49 n
【定義域】　n=0x00(送信されたイメージをそのまま印刷するように指定します。)
n=0x01(送信されたイメージを 2 倍に拡大するように指定します。)
【機能】　ビットイメージの印刷サイズを指定します。
【動作】　ビットイメージを印刷する場合に、イメージを拡大して印刷するかそのまま印刷するかの指定を行います。

- ESC V n

【名称】　印字速度指定
【コード】　0x1B 0x56 n
【定義域】　n=0x00 (高速印字)
n=0x01 (低速印字(高品位))
n=0x02 (グラフィック印字)
【機能】　印字速度を指定します。
【動作】　プリンタの印字速度を指定します。
高速印字:印刷速度を重視した印字速度指定です。低速印字と比較し、印字品位が低下します。
低速印字:印字品位を重視した印字速度指定です。高速印字に比べて印字速度が低下します。
グラフィック印字:ビットイメージ印字を重視した印字速度です。ビットイメージの印字品位を向上させるため、低速印字よりも更に遅い速度での印字となります。
※印字速度指定は、印字停止(未印字吐き出し実行)後に行ってください。

- ESC D n

【名称】　印字濃度指定
【コード】　0x1B 0x44 n
【定義域】　0x01≦n≦0x09　(0x05:標準。n<0x05　標準より薄い。n>0x05　標準より濃い。)
【機能】　印字濃度を指定します。
【動作】　印字濃度は9段階の指定が可能であり、各印字速度指定に対して標準的に印加するエネルギーに対してエネルギーを増加、または減少させることにより印字濃度を制御します。
※印字濃度指定は、印字停止(未印字吐き出し実行)後に行ってください。

- ESC P n

【名称】　用紙指定
【コード】　0x1B 0x50 n
【定義域】　0x00≦n≦0x05
【機能】　印刷する用紙を指定します。

n	用紙種類		
0x00	F-200U9W6	(1P ロール:高感度)	推奨用紙
0x01	HS360	(1P ロール:標準)	
0x02	AFP-235	(1P ロール:長期保存)	
0x03	HG56S	(ラベル紙)	
0x04	TLC00	(2P)※	
0x05	Reserve		

※2P 紙を使用する場合は、印字速度(ESC V)もグラフィック印字に設定して下さい。
【動作】　各種用紙に対して最適なエネルギーを印加するように制御を行うため、印刷する用紙を指定します。
※用紙指定は、印字停止(未印字吐き出し実行)後に行ってください。

- ESC H n

【名称】　プリヒート指定
【コード】　0x1B 0x48 n
【定義域】　n=0x00(無効)、0x01(有効)
【機能】　プリヒート機能の有効／無効を指定します。
【動作】　紙に熱を加えることで発色(印刷)させていますが、低温環境下では熱を加えるヘッドも低温になるため発色しにくくなります。したがって、ヘッドが低温である場合は印刷前にヘッドに対してエネルギーを与えてヘッドを暖めることで、発色しにくい状態を防ぎます。

- ESC p n

【名称】　ヘッド温度保持機能
【コード】　0x1B 0x70 n
【定義域】　n=0x00(無効)
n=0x01～0x0C(ヘッド温度保持一定時間有効:(5 分単位)　例:0x01→5 分間)
n=0x0F(ヘッド温度保持常時有効)
【機能】　ヘッド温度保持機能の有効／無効を指定します。
【動作】　ヘッド温度を常時、または一定時間適温に保ちます。
本コマンドは、印刷停止状態で送信して下さい。
例

```
TCHAR ESCE[2] = {0x1B, 0x45}, ESCp[3] = {0x1B, 0x70, 0x0F};
 PRNTextOut(2, ESCE);    // 未印字吐き出し
 PRNTextOut(3, ESCp);    //  ヘッド温度保持機能:常時有効
```

- ESC T n

【名称】　オートローディング指定
【コード】　0x1B 0x54 n
【定義域】　n=0x00　　(無効)
0x0A≦n≦0x60 (有効。この場合 n はローディング量(mm 単位)を指定します。)
【機能】　オートローディングの有効、無効を指定します。
有効の場合は、ローディング量の指定も兼ねています。
【動作】　本体に専用の単票用紙ホルダを装着している状態で、単票用紙ホルダから用紙が挿入された場合に指定されたローディング量の紙送りを行います。ローディングの指定が 10mm 以下の場合は強制的に 10mm とします。
※ DT-9800 のみ有効です

- ESC R n

【名称】　エラー時継続印字指定
【コード】　0x1B 0x52 n
【定義域】　n=0x00(無効)、0x01(有効)
【機能】　エラー発生時の継続印字の有効、無効を指定します。
【動作】　エラーが発生した場合に既に受信済みの印字データをクリア、または保持します。
エラーが発生した場合に最初から帳票を印刷する場合は、予めエラー時継続印字指定を無効とし、エラー発生後は帳票データを最初から送信して下さい。

- ESC M
  【名称】　マーカ検出
  【コード】　0x1B 0x4D
  【機能】　マーカを検出します
  【動作】　マーカ検出を実行すると、紙送りを行ってマーカを検出します。マーカを検出した時点で紙送りを停止しますが、30cm 以内にマーカを検出できなかった場合も紙送りを自動的に停止します。
  本コマンドは、単独で送信してください。また、本コマンドは印字完了後(未印字吐き出し終了後)に送信して下さい。
  TCHAR ESCM[2] = {0x1B, 0x4D};
  PRNTextOut(2, ESCM);

- ESC v n1 n2
  【名称】　マーカ検出モード
  【コード】　0x1B 0x76 n1 n2
  【定義域】　モード n1=0x00(マーカ検出終了)、0x01(マーカ開始検出)
  開始位置から検出完了までの距離(mm 指定) 0≦n2≦12
  【機能】　マーカ検出実行時の検出モードを指定します
  【動作】　マーカ終了検出を設定すると、マーカ検出実行時に”マーカがマーカセンサ上を通過したこと”を検出するようになります。マーカ終了検出設定は n2 を無視します。
  マーカ開始検出を設定すると、マーカ検出実行時に”マーカがマーカセンサ上に入ってから n2 だけフィードしたこと”を検出します。また、本設定はマーカ幅を満たさないマーカについても検出します。そのため、裏面バーコードのようにマーカ以外の黒がある用紙では使用しないでください。
  ※ IT-9000 のみ有効です

- ESC E
  【名称】　未印字データの吐出し(印字継続)
  【コード】　0x1B 0x45
  【機能】　未印字データを印字します。
  【動作】　印字データを全て送信した後、全てのデータの印字完了を待つためにこのコマンドを送信します。これにより、全ての印字データが印刷終了するまでのエラーを把握できます。
  またエラーで印字中断後、エラーを解除して印字を継続する場合はこのコマンドを送信して印字を継続します。

- ESC h n
  【名称】　バーコードの高さ指定
  【コード】　0x1B 0x68 n
  【定義域】　1≦n≦63(mm)
  【機能】　バーコードの高さを指定します。

- ESC c n
  【名称】　バーコードの C/D 指定
  【コード】　0x1B 0x63 n
  【定義域】　n=0x00(無効)、0x01(有効)
  【機能】　バーコードのチェックデジットの有無を指定します。

- ESC f n
  【名称】　バーコードのデータ印刷指定
  【コード】　0x1B 0x66 n
  【定義域】　n=0x00(付加文字無し)、0x01(8×16dot フォント)、0x02(6×7dot フォント)、0x03(OCR-B16×30dot フォント)
  【機能】　バーコードの下に印刷するデータのフォントを設定します。
  ※0x00 を設定した場合は、文字データは印刷されません

- ESC e n
  【名称】　バーコード印刷時のマージン指定
  【コード】　0x1B 0x65 n
  【定義域】　0≦n≦69(mm)、0xFFFF (用紙幅 80mm(82.55mm)の場合)
  0≦n≦45(mm)、0xFFFF (用紙幅 58mm の場合)
  【機能】　バーコード印刷時の左マージンを設定します。
  ※0xFFFF を設定した場合は、バーコードをセンタリングします。

- ESC d n
  【名称】　バーコードの印刷方向指定
  【コード】　0x1B 0x64 n
  【定義域】　n=0x00(縦方向)、0x01(横方向)
  【機能】　バーコードを印刷する方向を指定します。

- ESC g n1 n2 Data
  【名称】　バーコード印刷
  【コード】　0x1B 0x67 n1 n2 Data
  【定義域】　バーコードの種類指定
  n1=0x00(JAN)、0x01(NW7)、0x02(CODE39)、0x03(ITF)、0x04(UPC-E)、0x05(CODE128)
  バーコードのデータ長の指定
  n2 の範囲はバーコードの種類に依存
  【機能】　指定されたバーコードの種類とデータに従って、バーコードを印刷します。バーコード印刷の各種設定は、本コマンド実行前に行ってください。
  ※バーコード印刷時は、印字速度をグラフィックモードに設定してから印刷して下さい。

- ESC Z n
  【名称】　初期化
  【コード】　0x1B 0x5A n
  【定義域】　n＝0x00(全ての設定を初期化します。)
  n＝0x01(レジストリに保存されていない設定を初期化します。)
  【機能】　プリンタの設定を初期化します。
  【動作】　n の値により、初期化する内容が変化します。

6. DeviceEmulator

ここでは、次の 2 点について説明します。

- Device Emulator で印刷関数を実行したときに出力される印刷イメージファイル
- 印刷イメージファイルを bmp 形式ファイルに変換するツール

6.1 PrinterImage.bin

Device Emulator で印刷関数を実行すると、PC 上の下記のファイルに印刷イメージが保存されます。

C:¥Program Files¥Windows CE Tools¥wce500¥DT-9800¥
Emulation¥Ctrl¥Printer¥PrinterImage.bin

※ 既にファイルが存在する場合は、印刷イメージを追加保存します。

6.2 makePrnData.bat

印刷イメージファイル PrinterImage.bin の内容は、以下のツールにより、bmp 形式のファイルに変換できます。

makePrnData.bat
makePrnData.exe（makePrnData.bat から呼び出されます）

ツールはあらかじめ、下記のフォルダにインストールされています。

C:¥Program Files¥Windows CE Tools¥wce500¥DT-9800¥Emulation¥Ctrl¥Printer

バッチファイル makePrnData.bat を実行すると、PrinterImage.bin の内容が bmp 形式に変換され、同じフォルダに保存されます。bmp ファイルの名称は、Temp.bmp です。
※ 既に Temp.bmp が存在する場合は、上書き保存します。

カシオ計算機お問い合わせ窓口

製品に関する最新情報

●製品サポートサイト（カシオペア・ハンディターミナル）

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

製品の取扱い方法のお問い合わせ

●情報機器コールセンター

0570-022066
市内通話料金でご利用いただけます。
携帯電話・PHS 等をご利用の場合、042-503-7241

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