§.13　印刷ライブラリ

13-1.機能

DT-9700 の内蔵プリンタ用印刷システム/印刷ライブラリです。
本ソフトウェアを使用することにより、GDI 関数を用いて印刷データを作成して印刷することが可能になります(*)。

*ただし、GDI 関数/印刷システムの前処理が行われるため、印刷の開始まで多少時間がかかります。

13-2.動作環境

■機種　DT-9700
■OS　Microsoft WindowsCE .NET 4.1

13-3.開発環境

■Microsoft eMbedded C++ Version4.0 + SP1
■Microsoft Visual Studio .NET 2003

13-4.提供ファイル

●SDK

■Cp780Lib.lib	インポートライブラリ
■CpLib.h	印刷ライブラリヘッダファイル
■CpExtDvm.h	デバイスモード情報構造体定義ヘッダファイル
■Cp780LibCS.dll	C#用クラスライブラリ
■Cp780LibVB.dll	VB.NET 用クラスライブラ

●SAMPLE

■C#用サンプルプログラム
■C++用サンプルプログラム
■VB.NET サンプルプログラム

●BIN

■CSFPS_BI.cab	印刷ライブラリの DT-9700 用インストーラ
■Setup.exe	自動セットアップ

13-5.使用方法

●eMbedded Visual C++4.0：
開発環境において、プログラムソース内にライブラリヘッダファイル「CpLib.h」と、使用するプリンタのヘッダファイル「CpExtDvm.h」をインクルードし、インポートライブラリ「Cp780Lib.lib」を使用するライブラリとして指定して下さい。
●Visual Studio .NET 2003：
C#をご利用の場合、Visul Studio.NET2003 にて Cp780LibCS.dll を参照設定してください。
同様に、VB.NETご利用の場合もCp780LibVB.dllを参照設定してください。

13-6.制約事項

VB、及び C#で CpPrintDlg 関数を使用する場合、「プロパティ」ボタンを押してもプリンタプロパティダイアログが表示されません。使用する場合は、フォームに「メニューコントロール」を貼り付けて使用して下さい。

13-7.サポート関数

GDI 関数は、「13-15-2.GDI 関数確認一覧」に記載している関数をサポートしています。
*CpCreateDC で作成する DC がパレットに対応していないため、パレットに関する関数は正常に動作しません。

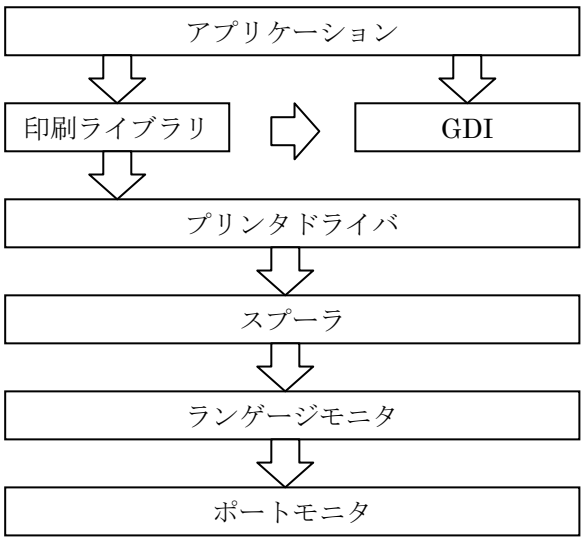
13-8.インストール手順

1）DT-9700 本体の任意の場所に CSFPS_BI.cab と Setup.exe を同じ場所にコピーします。
2）Setup.exe を実行します。
3）自動インストーラが起動し、CAB ファイル展開後、自動的にリセットします。
*インストールすると CAB ファイルは消去されます。消去したくない場合は、属性を”読取り専用”にしてください。

VB、C#で作成したアプリケーションを実行する場合事前に SDK フォルダの 「Cp780LibVB.dll」「Cp780LibCS.dll」を DT-9700 の Windows フォルダへコピーして下さい。

13-9.印刷システムを使用したプリンタへの印刷

アプリケーションから印刷ライブラリ関数を使用して印刷制御(印刷データの描画)を行います。基本的にはCpCreateDC で作成されたデバイスコンテキストに対して WinGDI 関数を使用して描画処理を行って、印刷を実現します。



13-10.印刷ライブラリ関数 及び印刷ライブラリ構造体一覧

関数名	概要
CpOpenPrinter	指定されたプリンタを識別するハンドルを取得します。
CpClosePrinter	指定されたプリンタオブジェクトをクローズします。
CpDocumentProperties	指定されたプリンタのプリンタ構成ダイアログボックスを表示します。
CpCreateDC	プリンタのデバイスコンテキスト(DC)を作成します。
CpResetDC	プリンタのデバイスコンテキスト(DC)を更新します。
CpDeleteDC	指定されたデバイスコンテキストを削除します。
CpGetDeviceCaps	プリンタの情報を取得します。
CpStartDoc	印刷ジョブを開始します。
CpAbortDoc	印刷ジョブを強制終了します。
CpEndDoc	印刷ジョブを終了します。
CpStartPage	1 ページ分の印刷を開始します。
CpEndPage	1 ページ分の印刷を終了します。
CpUnicodeOut	Unicode 文字列をプリンタ用コード(JIS 内部／Shift JIS)に変換して出力します。
CpExtEscape	アプリケーションが直接プリンタドライバに対してアクセスできるようにします。
CpPrintDlg	[プリンタ選択]ダイアログボックスを作成します。

構造体名	概要
CPDOCINFO	CpStartDoc 関数が使用する入力ファイル名と出力ファイル名を格納します。
CPDEVMODE	プリンタの初期化と環境データに関する情報を格納します。
CPEXTDVM	プリンタの詳細設定に関する情報を格納します。
CPCDEVMODE	CPDEVMODE と CPEXTDVM をメンバに持つ構造体です。

13-10-1.関数リファレンス

次項より関数リファレンスを記述します。

➢ CpOpenPrinter

機能　指定されたプリンタを識別するハンドルを取得します。

書式　**BOOL CpOpenPrinter (LPTSTR lpszPrinter, LPTSTR lpszPort, LPHANDLE lphPrinter)**

パラメータ

lpszPrinter	プリンタ名を指定します。
lpszPort	出力先ポート名を指定します。
lphPrinte	オープンされたプリンタのハンドルが返ります。

戻り値　関数が正常終了した場合は、TRUE を返します。それ以外の場合は、FALSE を返します。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpClosePrinter

機能　指定されたプリンタオブジェクトをクローズします。

書式　**BOOL CpClosePrinter (HANDLE hPrinter)**

パラメータ

hPrinter　　オープンされたプリンタのハンドルを指定します。

戻り値　関数が正常終了した場合は、TRUE を返します。それ以外の場合は、FALSE を返します。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpDocumentProperties

機能　指定されたプリンタのプリンタ構成ダイアログボックスを表示します。

書式　**LONG CpDocumentProperties(HWND hwnd, HANDLE hPrinter, LPCPDEVMODE lpdmOutput, LPCPDEVMODE lpdmInput, DWORD fMode)**

パラメータ

Hwnd	ダイアログボックスを表示するときの親ウィンドハンドル
hPrinter	オープンされたプリンタのハンドル
lpdmOutput	修正されたプリンタデータ
lpdmIntput	元のプリンタデータ
fMode	関数が実行する操作を示す値を指定します。このパラメータが 0 ならば、CPDEVMODE データ構造体に必要なバイト数を返します。それ以外の場合は、次に示す値を 1 つ以上組み合わせてこのパラメータに指定します。

CPDM_IN_BUFFER	関数への入力を指定します。
CPDM_IN_PROMPT	プリンタ設定ダイアログを表示します。
CPDM_OUT_BUFFER	関数からの出力を指定します。

戻り値　fMode パラメータが 0 か、lpdmOutput が NULL ならば、プリンタドライバの初期化データを設定するのに必要なバッファのサイズを返します。プリンタドライバが構造体にプライベートなデータを付加しているときは、このバッファは CPDEVMODE 構造体よりも大きくなります。
関数が初期化ダイアログ ボックスを表示したときは、ユーザーが選択したボタンに応じて、IDOK または IDCANCEL を返します。関数が正常に終了しなかったときは、0 未満の値を返します。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpCreateDC

機能　プリンタのデバイスコンテキスト(DC)を作成します。作成された DC に対して各 GDI 関数が使用できます。

書式　**HDC CpCreateDC(HANDLE hPrinter, LPCPDEVMODE lpInitData)**

パラメータ

hPrinter	オープンされたプリンタのハンドル
lpInitData	プリンタデータ

戻り値　関数が正常終了した場合は、指定されたプリンタ用のデバイスコンテキストのハンドルを返します。それ以外の場合は、NULL を返します。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpResetDC

機能　プリンタのデバイスコンテキスト(DC)を更新します。1 ドキュメント内での用紙方向切り替え、用紙サイズ変更時に使用します。

書式　**HDC CpResetDC(HANDLE hPrinter, HDC hdc, LPCPDEVMODE lpInitData)**

パラメータ

hPrinter	オープンされたプリンタのハンドル
Hdc	デバイスコンテキスト
lpInitData	プリンタデータ

戻り値　関数が正常終了した場合は、元のデバイスコンテキストのハンドルと元のプリンタ制御情報を返します。それ以外の場合は、NULL を返します。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpDeleteDC

機能　指定されたデバイスコンテキストを削除します。

書式　**BOOL CpDeleteDC(HANDLE hPrinter, HDC hdc)**

パラメータ

hPrinter	オープンされたプリンタのハンドル
Hdc	デバイスコンテキスト

戻り値　関数が正常終了した場合は、TRUE を返します。それ以外の場合は、FALSE を返します。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpGetDeviceCaps

機能　プリンタの情報を取得します。

書式　**int CpGetDeviceCaps(HANDLE hPrinter, HDC hdc, int nIndex)**

パラメータ

hPrinter	オープンされたプリンタのハンドル	
Hdc	デバイスコンテキスト	
nIndex	機能取得番号	
	CP_HORZRES	印刷可能領域幅(ピクセル単位)
	CP_VERTRES	印刷可能領域長さ(ピクセル単位)
	CP_HORZSIZE	印刷可能領域幅(0.1mm 単位)
	CP_VERTSIZE	印刷可能領域長さ(0.1㎜ 単位)
	CP_LOGPIXELSX	水平方向のインチ当たりのピクセル数
	CP_LOGPIXELSY	垂直方向のインチ当たりのピクセル数
	CP_LOGMMPIXELSX	水平方向の 1㎜ 当たりのピクセル数
	CP_LOGMMPIXELSY	垂直方向の 1㎜ 当たりのピクセル数
	CP_PRINTER	ページ制御プリンタ CP_DT_PAGE(1)か、ライン制御プリンタ CP_DT_LINE(2)かが返ります。
	CP_TEXTCAPS	CpUnicodeOut 描画の出力方式 直接出力：CP_TC_DIRECT(1)が返ります。

戻り値　要求された機能取得番号項目の値を返します。

➢ CpStartDoc

機能　印刷ジョブを開始します。

書式　**int CpStartDoc(HANDLE hPrinter, HDC hdc, LPCPDOCINFO lpdi)**

パラメータ

hPrinter	オープンされたプリンタのハンドル
Hdc	デバイスコンテキスト
Lpdi	文書名と出力ファイル名
lpdi->lpszDocName	印刷中ダイアログに表示される文書名を指定します
lpdi->lpszOutput	空文字(_T(“”))を指定して下さい。

戻り値　関数が正常終了した場合は、正の値を返します。それ以外は CP_ERROR を返します。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpAbortDoc

機能　印刷ジョブを強制終了します。

書式　**int CpAbortDoc(HANDLE hPrinter, HDC hdc)**

パラメータ

hPrinter	オープンされたプリンタのハンドル
Hdc	デバイスコンテキスト

戻り値　関数が正常終了した場合は、正の値を返します。それ以外は CP_ERROR を返します。

➢ CpEndDoc

機能　印刷ジョブを終了します。

書式　**int CpEndDoc(HANDLE hPrinter, HDC hdc)**

パラメータ

hPrinter	オープンされたプリンタのハンドル
Hdc	デバイスコンテキスト

戻り値　関数が正常終了した場合は、正の値を返します。それ以外は CP_ERROR を返します。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpStartPage

機能　１ページ分の印刷を開始します。

書式　**int CpStartPage(HANDLE hPrinter, HDC hdc)**

パラメータ

hPrinter	オープンされたプリンタのハンドル
Hdc	デバイスコンテキスト

戻り値　関数が正常終了した場合は、正の値を返します。それ以外は CP_ERROR を返します。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpEndPage

機能　１ページ分の印刷を終了します。

書式　**int CpEndPage(HANDLE hPrinter, HDC hdc)**

パラメータ

hPrinter	オープンされたプリンタのハンドル
Hdc	デバイスコンテキスト

戻り値　関数が正常終了した場合は、正の値を返します。それ以外は CP_ERROR を返します。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpUnicodeOut

機能　Unicode 文字列をプリンタ用コード(JIS 内部／Shift JIS)に変換して出力します。

書式　**int CpUnicodeOut(HANDLE hPrinter, HDC hdc, LPCTSTR lpszString, UINT cbCount)**

パラメータ

hPrinter	オープンされたプリンタのハンドル
Hdc	デバイスコンテキスト
lpszString	文字列のアドレス
cbCount	文字列中の文字数

戻り値　関数が正常終了した場合は、正の値を返します。それ以外は CP_ERROR を返します。

注意　本関数は Unicode 文字列をプリンタ用コード(JIS 内部／Shift JIS)に変換して出力する他にはプリンタコマンドを出力しません。したがって、ESC コマンドでの各種設定は CpExtEscape()で出力する必要があります。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpExtEscape

機能　アプリケーションが直接プリンタドライバに対してアクセスできるようにします。
文字、バーコードのプリンタコマンドを印字位置等の制御コマンドを使用して直接印刷できます。
ただし、プリンタ毎にプリンタコマンドが異なりますので注意が必要です。

書式　**int CpExtEscape(HANDLE hPrinter, HDC hdc, int nEscape, int cbInput,**
LPCSTR lpvInData, int cbOutput, LPVOID lpvOutData)

パラメータ

hPrinter	オープンされたプリンタのハンドル
Hdc	デバイスコンテキスト
nEscape	CPPASSTHROGH を指定して下さい。 これは、直接プリンタドライバにデータを送ることを意味します。
cbInput	lpvInData が指すデータのバイト数を指定します。
lpvInData	指定されたエスケープに必要なデータを指定します。
cbOutput	lpvOutData が指すデータのバイト数を指定します。
lpvOutData	指定されたエスケープからの出力を受け取るデータを指定します。

戻り値　関数が正常終了した場合は、正の値を返します。エスケープが実装されていない場合は、0 を返します。それ以外は CP_ERROR を返します。

参考　13-15 プログラミングガイドを参照して下さい。

➢ CpPrintDlg

機能　［プリンタ選択］ダイアログボックスを表示します。ユーザーは［プリンタ選択］ダイアログボックスを使って、『プリンタの選択』『ポートの選択』とプリンタのプロパティを設定することができます。
【CpPrintDlg】ではこの関数で【CpOpenPrinter】と【CpDocumentProperties】を同時に行ないます。

書式　**int CpPrintDlg(HWND hwnd, LPHANDLE lphPrinter, LPCPDEVMODE lpdmOutput, LPTSTR lpszPrinter, LPTSTR lpszPort)**

パラメータ

Hwnd	ダイアログボックスを表示する時の親ウィンドウハンドル
lphPrinter	オープンされたプリンタのハンドル
lpdmOutput	CPDEVMODE へのポインタを指定
lpszPrinter	出力するプリンタ名
lpszPort	出力するポート名

戻り値　lpdmOutput が NULL ならば、プリンタドライバの初期化データを設定するのに必要なバッファのサイズを返します。プリンタドライバが構造体にプライベートなデータを付加しているときは、このバッファは CPDEVMODE 構造体よりも大きくなります。
関数がダイアログ ボックスを表示したときは、ユーザーが選択したボタンに応じて、IDOK、または IDCANCEL を返します。関数が正常に終了しなかったときは、0 未満の値を返します。

13-10-2.印刷ライブラリ構造体リファレンス

➢ CPDOCINFO

CPDOCINFO 構造体は、CpStartDoc 関数が使用する入力ファイル名と出力ファイル名を格納します。

```
typedef struct {
    int         cbSize;
    TCHAR       DocName[MAX_DOCNAME+1];
    TCHAR       Output[MAX_OUTPUT+1];
} CPDOCINFO;
```

メンバ

cbSize	構造体のサイズをバイト単位で指定します。
lpszDocName	文書の名前を指定します。(MAX_DOCNAME=32)
lpszOutput	使用されません。空文字列を指定してください。(MAX_ OUTPUT =32)

➢ CPDEVMODE

CPDEVMODE 構造体は、プリンタデバイスの初期化と環境データに関する情報を格納します。

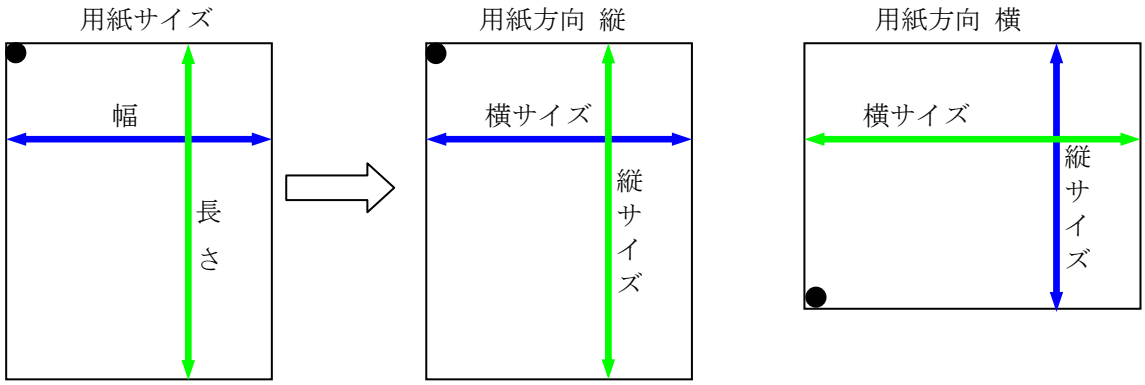
```
typedef struct {
    WORD        dmDriverVersion;
    WORD        dmSize;
    WORD        dmDriverExtra;
    WORD        dmOrientation;
    WORD        dmPaperSize;
    WORD        dmPaperLength;
    WORD        dmPaperWidth;
    WORD        dmCopies;
    WORD        dmPrintQuality;
    WORD        dmColor;
    WORD        dmBitsPerPel;
    WORD        dmLogmmPixelsX;
    WORD        dmLogmmPixelsY;
    WORD        dmCollate;
    WORD        dmDitherType;
} CPDEVMODE;
```

メンバ

dmDriverVersion	プリンタドライバの開発者により割り当てられる、プリンタドライバのバージョン番号を指定します。
dmSize	CPDEVMODE 構造体のサイズをバイト単位で指定します。プライベートなドライバデータのサイズは含まれません。
dmDriverExtra	この構造体の直後に続く、プライベートなドライバデータ(拡張情報 CPEXTDVM)のバイト数を指定します。
dmOrientation	用紙方向を指定します。下記のいずれかを指定します。 縦：CPDMORIENT_PORTRAIT(1) 横：CPDMORIENT_LANDSCAPE(2)
dmPaperSize	印刷用紙サイズ(CPDMPAPER_FREE)を指定します。これにより用紙の長さと幅を dmPaperLength と dmPaperWidth に設定します。
dmPaperLength	用紙の長さを 0.1mm 単位で指定します。
dmPaperWidth	用紙の幅を 0.1mm 単位で指定します。
dmCopies	印刷する部数を指定します。
dmPrintQuality	プリンタの解像度を 1mm 当たりのドット数で指定します。
dmColor	カラー／モノクロを指定します。下記のいずれかを指定します。 モノクロ：CPDMCOLOR_MONOCHROME(1) カラー　：CPDMCOLOR_COLOR(2)
dmBitsPerPel	色解像度をピクセル当たりのビット数で指定します。
dmLogmmPixelsX	横方向の 1mm 当たりのドット数を指定します。
dmLogmmPixelsY	縦方向の 1mm 当たりのドット数を指定します。
dmCollate	部単位印刷の指定をします。(有効(1)／無効(0))
dmDitherType	ディザリングの種類を指定します。下記のいずれかを指定します。 なし(完全 2 値)：　CPDMDITHER_NON パターン　　：　CPDMDITHER_ PATTERN 誤差拡散　　：　CPDMDITHER_ ERRORDIFFUSION

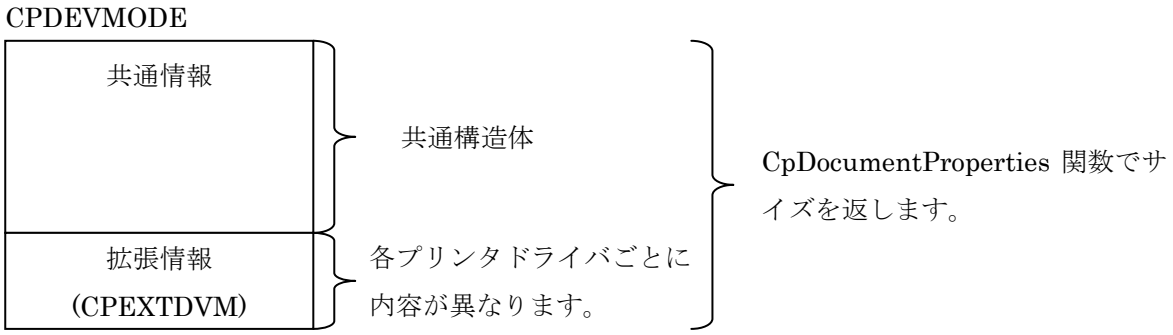
■ 用紙サイズと用紙方向との関係

用紙サイズの指定(幅／長さ)は物理サイズの指定になります。用紙方向による論理用紙サイズは以下の通りになります。



➢ CPEXTDVM(拡張情報)

CPDEVMODE の直後に続くプライベートなドライバデータは、各プリンタドライバ毎に内容が異なります。



■ CPEXTDVM(内蔵プリンタ設定項目)

内蔵プリンタ専用の設定項目の構造体です。

```
typedef struct {
    WORD        stSize;
    WORD        PrintSpeed;
    WORD        PrintDensity;
    WORD        PaperKind;
    WORD        bPreHeat;
    WORD        bAutoLoading;
    WORD        AutoLoading;
    WORD        bErrorContinuation;
    WORD        bMarkerDetection;
} CPEXTDVM
```

メンバ

stSize	CPEXTDVM 構造体のサイズをバイト単位で指定します。
PrintSpeed	印字速度の指定をします。以下の 3 種類を指定できます。
	高速印字　PRINTSPEED_FAST
	低速印字(高品位)　PRINTSPEED_LATE
	グラフィック印字　PRINTSPEED_GRAPHIC
PrintDensity	印字濃度の指定をします。9 段階で指定できます。
	Level 1：DENSITY_LV1 ～ Level 9：DENSITY_LV9
PaperKind	用紙種類の指定をします。以下に示す 5 種類を指定できます。
	１P(高感度)　PKIND_F200U9W6
	１P(標準)　PKIND_HS360
	１P(長期保存)　PKIND_AFP235
	ラベル　PKIND_HG56S
	２P　PKIND_TLC00
bPreHeat	プリヒートの指定をします。(有効(1)／無効(0))
bAutoLoading	オートローディングの指定をします。(有効(1)／無効(0))
AutoLoading	オートローディングでのペーパーフィード量を mm 単位で指定します。
	最大値　MAX_AUTOLOADING(96)
	最長値　MIN_AUTOLOADING(10)
bErrorContinuation	エラー時継続印字の指定をします。(有効(1)／無効(0))
bMarkerDetection	マーカ検出の指定をします。(有効(1)／無効(0))
	有効時は、ページ毎に印刷前にマーカ検出を行います

➢ CPCDEVMODE

CPDEVMODE と CPEXTDVM をメンバに持つ構造体です。CPEXTDVM 内の設定値を変更したい場合等に使用します。

```
typedef struct {
    CPDEVMODE    dm;
    CPEXTDVM     dlg;
} CPCDEVMODE;
```

メンバ

Dm	標準デバイスモード情報(13-7-2 参照)
Dlg	拡張デバイスモード情報(13-7-3 参照)

※13-15 プログラミングガイドのコメントを参考にして下さい。

13-11.プリンタ名と出力ポート名へについて

プリンタ名と対応する出力ポート名を以下に示します。

	プリンタ名	出力ポート名
内蔵プリンタ(日本語版)	BuiltIn	DT9700 BuiltIn
内蔵プリンタ(英語版)	BuiltIn	IT3000 BuiltIn

13-12.用紙サイズについて

プリンタの指定可能用紙サイズを以下に示します。
※このサイズはデバイスコンテキストに作成される描画イメージのサイズを指します。

	幅[0.1mm]	長さ[0.1mm]
内蔵プリンタ	480 or 720	20 ～ 9999

13-13.ランゲージモニタについて

スプーラからの制御により、プリンタポートとの送受信を行います。また、データ出力時の状態監視、エラー制御も行います。

13-13-1.インターフェースについて

印刷が開始されると、タスクトレイにプリンタアイコンが出力され、アイコンをダブルクリックすると印刷中を表すダイアログボックスが出力されます。印刷が終了すると、ダイアログボックスが出力されていれば、ダイアログボックスは閉じ、アイコンが出力されていれば、アイコンは削除されます。また、印刷中ダイアログボックスは以下の動作が実行可能です。

キャンセルボタン押下　　印刷を中断します。

印刷中にエラーが発生すると、エラーを表すダイアログボックスが出力されます。エラーダイアログが出力された際は、以下の動作の実行を選択します。

継続ボタン押下　　引き続き印刷を実行します。
中止ボタン　　印刷を中断します。

13-13-2 各種エラーについて

エラー表示	エラー内容	対処方法
用紙がありません	用紙切れ 用紙なし	用紙をセットする
プラテンがオープンしています	プラテンが開いている	プラテンを閉じる
VDETP が発生しました	ローバッテリー状態	バッテリーを充電、交換する
サスペンドが発生しました	サスペンド(印字中断)状態	エラー時継続印字を有効にして印刷する
ヘッド温度エラー	ヘッド温度上昇	ヘッド温度が下がるまで印字しない
マーカー検出エラー	マーカーが検出できない	・ 印刷を中止し、マーカー検出を無効にして再度印刷する ・ マーカー用紙で再度印刷する
80mm へ変更できません	用紙サイズ変更エラー	印刷を中止し、用紙幅の設定を 48.0mm に設定し、58mm 用紙で印刷する

13-14.プリンタへの直接印刷について

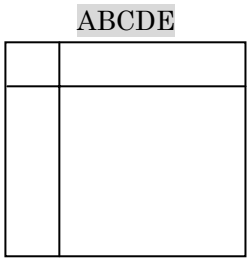
CpUnicodeOut の直接プリンタ印刷サポート時、または CpExtEscape でプリンタへ直接印刷を行う場合の印刷結果は、以下のようになります。

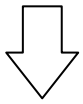
13-14-1.ライン制御プリンタの場合 (DT9700/IT3000 BuiltIn)

アプリケーションの処理

① CpUnicodeOut でテキストを描画 ----------►

② プリンタのデバイスコンテキストに対して描画 ----------►



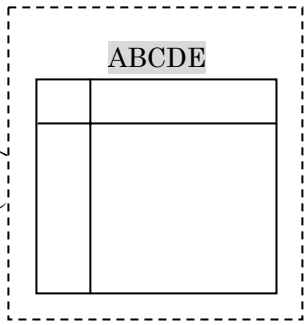


プリンタの処理

① CpUnicodeOut の内容を描画

② デバイスコンテキストのデータを描画

ライン単位で印刷するため、CpUnicodeOut で直接プリンタへ送信されると送信された時点で印刷データが作成されます。したがって、その後のデータと重ねることができません。



印刷結果

13-15.プログラミングガイド

13-15-1.サンプルプログラム

サンプルプログラムを用いて、ライブラリ関数の使用方法を解説します。

本サンプルプログラムではライブラリの関数を使用して，GDI 関数である”LineTo”により描画されたイメージを１ページ目に印刷し、用紙サイズを変更して、文字列を２ページ目に印刷します。

使用されるライブラリの関数を以下に示します。

- CpOpenPrinter
- CpDocumentProperties
- CpCreateDC
- CpStartDoc
- CpStartPage
- LineTo (デバイスコンテキスト用描画関数)
- CpEndPage
- CpResetDC
- CpStartPage
- CpUnicodeOut
- CpExtEscape
- CpEndPage
- CpEndDoc
- CpDeleteDC
- CpClosePrinter

実際のプログラムコードを以下に示します。

※ ヘッダファイル(Cp780Lib.h)がインクルードされており、ライブラリ(Cp780Lib.lib)がリンクされているものとします。

```
TCHAR     szUni[] = _T("UnicodeOut テスト");
char      szESC_E[] = "¥x1b""E";

void Sample(HWND hWnd)                    // hWnd  アプリケーションのウィンドウハンドル
{
          HANDLE hPrinter;
          HDC hDC, hDCOld;
          LPCPDEVMODE pInCDM = NULL;      // プリンタデータ入力値
          LPCPDEVMODE pCDM = NULL;        // プリンタデータ領域
          CPDOCINFO doc;
          int fMode = CPDM_OUT_BUFFER;    // プリンタデータ出力要求
          int size;

          /***** CpOpenPrinter：指定プリンタをオープンしてハンドルを取得します *****/
          // プリンタ名：BuiltIn(内蔵プリンタ)  出力ポート名：DT9700 BuiltIn(内蔵プリンタ)
          if (CpOpenPrinter(_T("BuiltIn"), _T("DT9700 BuiltIn"), &hPrinter) != TRUE)
                    return;

          /***** CpDocumentProperties：プリンタデータを取得します *****/
          // プリンタデータ保存用領域として必要なサイズを取得
          size = CpDocumentProperties(hWnd, hPrinter, pCDM, pInCDM, fMode);
          if (size < 0) {
                    CpClosePrinter(hPrinter);
                    return;
          }
          // 取得された必要サイズ分の領域を確保
          pCDM = (LPCPDEVMODE)malloc(size);
          // CPDEVMODE のデフォルト情報を取得
          if (CpDocumentProperties(hWnd, hPrinter, pCDM, pInCDM, fMode) < 0) {
                    CpClosePrinter(hPrinter);
                    free(pCDM);
                    return;
          }

          /***** 描画サイズ(長さ)を変更 *****/
          // 描画サイズ(長さ)を 200(20[mm])に変更
          pCDM->dmPaperLength = 200;

          // CpDocumentProperties の引数 fMode の CPDM_IN_PROMPT プリンタプロパティによる値
          //の変更以外にも上記コードのように値を直接指定することも可能です。

          /***** プリンタ毎の拡張情報を変更したい場合 *****/
          // プリンタ拡張情報定義のヘッダファイル(CpExtDvm.h)のインクルードが必要です。
          // 内蔵プリンタの【印字速度】を「高速印字」に変更する場合
          // ((LPCPCDEVMODE)pCDM)->dlg.PrintSpeed = PRINTSPEED_FAST;

          /***** CpCreateDC：描画用のデバイスコンテキストハンドルを取得します *****/
          hDC = CpCreateDC(hPrinter, pCDM);
          if (hDC == NULL) {
                    CpClosePrinter(hPrinter);
                    free(pCDM);
                    return;
          }
          /***** CpSartDoc：印刷ドキュメントの開始設定を行います *****/
          // CPDOCINFO 設定
          doc.cbSize = sizeof(CPDOCINFO);
          wcscpy((LPWSTR)doc.DocName, (LPWSTR)_T("ドキュメント名"));
          wcscpy((LPWSTR)doc.Output, (LPWSTR)_T(""));

          if (CpStartDoc(hPrinter, hDC, &doc) == CP_ERROR) {
                    CpDeleteDC(hPrinter, hDC);
                    CpClosePrinter(hPrinter);
                    free(pCDM);
                    return;
          }

          /***** CpStartPage：ページの開始設定を行います *****/
          if (CpStartPage(hPrinter, hDC) == CP_ERROR) {
```

```
            CpEndDoc(hPrinter, hDC);
            CpDeleteDC(hPrinter, hDC);
            CpClosePrinter(hPrinter);
            free(pCDM);
            return;
}

/***** イメージ描画：デバイスコンテキストハンドルに右のイメージを描画しています *****/
LineTo(hDC, 100, 100);          //
LineTo(hDC, 100, 0);            //
LineTo(hDC, 0, 0);              //

/***** CpEndPage：ページの終了設定を行います *****/
if (CpEndPage(hPrinter, hDC) == CP_ERROR) {
            CpEndDoc(hPrinter, hDC);
            CpDeleteDC(hPrinter, hDC);
            CpClosePrinter(hPrinter);
            free(pCDM);
            return;
}

/***** CpResetDC：用紙サイズを変更します *****/
hDCOld = hDC;
pCDM->dmPaperLength = 100;                  // 描画サイズ(長さ)を 200 から 100(10[mm])に変更
hDC = CpResetDC(hPrinter, hDC, pCDM);
if (hDC == NULL) {
            CpEndDoc(hPrinter, hDCOld);
            CpDeleteDC(hPrinter, hDCOld);
            CpClosePrinter(hPrinter);
            free(pCDM);
            return;
}
/***** CpStartPage：ページの開始設定を行います *****/
if (CpStartPage(hPrinter, hDC) == CP_ERROR) {
            CpEndDoc(hPrinter, hDC);
            CpDeleteDC(hPrinter, hDC);
            CpClosePrinter(hPrinter);
            free(pCDM);
            return;
}

/***** CpUnicodeOut：文字列を送ります *****/
if (CpUnicodeOut(hPrinter, hDC, szUni, lstrlen(szUni)) == CP_ERROR) {
            CpEndPage(hPrinter, hDC);
            CpEndDoc(hPrinter, hDC);
            CpDeleteDC(hPrinter, hDC);
            CpClosePrinter(hPrinter);
            free(pCDM);
            return;
}

/***** CpExtEscape：[ESC]E（未印字データの吐き出し）コマンドを送ります *****/
if (CpExtEscape(hPrinter, hDC, CPPASSTHROGH, (int)sizeof(szESC_E),
            (LPCSTR)szESC_E, 0, NULL) == CP_ERROR) {
            CpEndPage(hPrinter, hDC);
            CpEndDoc(hPrinter, hDC);
            CpDeleteDC(hPrinter, hDC);
            CpClosePrinter(hPrinter);
            free(pCDM);
            return;
}

/***** CpEndPage：ページの終了設定を行います *****/
if (CpEndPage(hPrinter, hDC) == CP_ERROR) {
            CpEndDoc(hPrinter, hDC);
            CpDeleteDC(hPrinter, hDC);
            CpClosePrinter(hPrinter);
            free(pCDM);
```

```
            return;
        }

        /***** CpEndDoc：ドキュメントの終了設定を行います *****/
        // CpEndDoc 実行により、印刷がスタートします。
        if (CpEndDoc(hPrinter, hDC) == CP_ERROR) {
            CpDeleteDC(hPrinter, hDC);
            CpClosePrinter(hPrinter);
            free(pCDM);
            return;
        }

        /***** CpDeleteDC：デバイスコンテキストを削除します *****/
        if (CpDeleteDC(hPrinter, hDC) != TRUE) {
            CpClosePrinter(hPrinter);
            free(pCDM);
            return;
        }

        /***** CpClosePrinter：指定されたプリンタを閉じます *****/
        if (CpClosePrinter(hPrinter) != TRUE) {
            free(pCDM);
            return;
        }

        // プリンタデータ保存領域を解放
        if(pCDM)
            free(pCDM);
}
```

13-15-2.GDI 関数確認一覧

Windows CE.Net 4.1 で CpCreateDC で作成された DC に対して描画等を行い、確認条件下において正常に動作する関数の一覧を示します。印刷システムにおいては、下表の GDI 関数を使用して下さい。

AddFontResource	ExtTextOut	OffsetRgn
BitBlt	FillRect	PatBlt
CombineRgn	FillRgn	Polygon
CreateBitmap	GetBkColor	Polyline
CreateCompatibleBitmap	GetBkMode	PtInRegion
CreateCompatibleDC	GetCharWidth32	Rectangle
CreateDC	GetClipBox	RectInRegion
CreateDIBPatternBrushPt	GetClipRgn	RectVisible
CreateDIBSection	GetCurrentObject	RestoreDC
CreateFontIndirect	GetCurrentPositionEx	RoundRect
CreatePatternBrush	GetDeviceCaps	SaveDC
CreatePen	GetDIBColorTable	SelectClipRgn
CreatePenIndirect	GetNearestColor	SelectObject
CreateRectRgn	GetObject	SetBitmapBits
CreateRectRgnIndirect	GetObjectType	SetBkColor
CreateSolidBrush	GetPixel	SetBkMode
DeleteDC	GetRegionData	SetBrushOrgEx
DeleteObject	GetRgnBox	SetDIBColorTable
DrawEdge	GetStockObject	SetDIBitsToDevice
DrawFocusRect	GetSysColorBrush	SetPixel
DrawText	GetTextAlign	SetRectRgn
Ellipse	GetTextColor	SetROP2
EnableEUDC	GetTextExtentExPoint	SetTextAlign
EnumFontFamilies	GetTextFace	SetTextColor
EnumFonts	GetTextMetrics	SetViewportOrgEx
EqualRgn	IntersectClipRect	StretchBlt
ExcludeClipRect	LineTo	StretchDIBits
ExtCreateRerion	MaskBlt	TranslateCharsetInfo
ExtEscape	MoveToEx	TransparentImage