

PM100 スキャナー SDK

コマンドリファレンス・マニュアル

Ver 1.1



ここに含まれる情報は、浜松東亜電機株式会社が独占的な権利を持ち、浜松東亜電機株式会社の書面による事前の許諾を得ずに全部もしくは一部を配布・複製または開示してはならないものとする。

目次

1 ドキュメントについて	5
1.1 改定履歴	5
2 序論	6
2.1 概要	6
2.1.1 ユーザーレイヤー	6
2.1.2 SDK	6
2.1.3 ハードウェアインタフェースレイヤー	6
3 定義、API関数、メッセージ	7
3.1 APIの構成	7
3.1.1 スキャナー関数	7
BcdOpen	7
BcdClose	7
BcdEnable	7
BcdIsEnabled	7
BcdSetHWND	8
BcdSetTerminator	8
BcdSetResultType	8
BcdGetResultType	8
BcdEnableBeep	9
BcdEnableLED	9
BcdEnablePrefix	9
BcdSetPrefix	9
BcdGetPrefix	9
BcdEnablePostfix	10
BcdSetPostfix	10
BcdGetPostfix	10
BcdSetTrigTimeout	10
BcdEanbleAutoScan	10
BcdSetAutoScanInterval	11
BcdSetConfig	11
BcdGetConfig	11
BcdGetBarTypeString	11
BcdGetEnableLength	11
BcdGetBarLength	12
BcdSetBarLength	12
BcdTransmitBarID	12
BcdGetBarID	12
BcdSetBarID	13
3.1.2 インジック関数	14
BcdSetDefault	14
BcdEnableBarcode	14
BcdEnableALLBarcode	14
BcdIsBarcodeEnabled	14
BcdSetBarFlag	14
BcdGetBarFlagArrayPtr	15

BcdTrigTurnOn.....	15
BcdGetResult	15
BcdGetString.....	15
BcdGetBarType	15
BcdGetLength.....	16
BcdGetVersion	16
BcdTriggerEnable	16
BcdIsTriggerEnabled	16
BcdSetContinuousMode	16
BcdGetContinuousMode.....	17
3.1.3 スキャナー2D IT5300専用関数	17
BcdGetDefaultBarLength.....	17
BcdGetOCRFlagEX	17
BcdSetOCRFlag.....	17
BcdGetDecoderVersion	18
BcdGetApiVersion	18
BcdGetScaDriverVersion.....	18
BcdSetSCANPCLK	18
BcdIs24MHzPCLK	18
BcdEnableCenterWindow	19
BcdSetDecodeMode	19
BcdSetDelayDecoding	19
BcdCreateDIBSection	19
BcdDestroyDIBSection	20
BcdGetCaptureImage.....	20
BcdImageStreamInit	20
BcdImageStreamStart	20
BcdImageStreamRead	21
BcdImageStreamStop.....	21
BcdAimerOn.....	21
BcdIlluminationOn.....	21
BcdSetScanLightsMode	22
BcdGetEngineConfig	22
BcdSetScanImageMode	22
3.1.4 スキャナー1D IS4813専用関数	22
BcdSetExtraReads	22
BcdGetExtraReads	23
BcdSetExtraTime	23
BcdSetExtraTimeEX	23
BcdGetExtraTime.....	23
BcdSetStitchingEnable	24
BcdSetStitchingEnable	24
BcdSetStitchingLevel.....	24
BcdSetStitchingLevel.....	24
3.1.4 その他の関数.....	25
BcdGetModelName	25
BcdGetModelSerialNumber.....	25
BcdGetBeepScale	25
BcdSetBeepScale.....	25
3.2 関数と関数ポインタ	26
3.3 スキャナーAPIの使い方.....	28
3.4 構造体と定義の構成	29
3.4.1 スキャナー構造体と定義.....	29

3.4.2 エンジン構造体と定義	41
3.5 メッセージ	43
(1) WM_SCANNED	43
(2) WM_IMAGE	43
4 BCD.NET.DLL	44
4.1 BCD.NET.dllとは	44
4.2 BCD.NET.dll の構成	44
4.3 BCD.NET.dll 使用サンプル	45

1 ドキュメントについて

このドキュメントは、PM100のイメージスキャナーに関するSDK(コマンドリファレンスマニュアル)です。

このドキュメントの内容は、下記の版数に対応します。

(版数の確認方法は、PM1000ユーザーマニュアルを参照して下さい。)

適用OSイメージ 版数	適用BCDCore. dll版数
10.03 (Aug 29 2011)	V1.4.0
10.04 (Dec 21 2011)	V1.?.?

SDKは、本書以外に下記のものがあります。

・UnitAPI SDK(コマンドリファレンスマニュアル)

1.1 改定履歴

バージョン	日付	説明	作者
1.0	2011年10月26日	1.日本語版 初版発行	浜松東亜電機株式会社
1.1	2012年1月25日	1. OS(V10.04)に対応 ・「BcdSetContinuousMode」追加 ・「BcdGetContinuousMode」追加 ・「BcdSetExtraReads」追加 ・「BcdGetExtraReads」追加 ・「BcdSetExtraTime」追加 ・「BcdGetExtraTime」追加 ・「BcdSetStitchingEnable」追加 ・「BcdSetAllowNonProportional」追加 ・「BcdSetStitchingLevel」追加 ・「BcdSetRoundingUncertaintyLevel」追加 誤植修正	浜松東亜電機株式会社

2 序論

2.1 概要

このSDKは、PM250(2次元)のスキャナー制御を行うものです。

このSDKは、スキャナー制御のための各種定義、API、メッセージから構成されます。

EVC++ 4.0(Embedded Visual C++), C#, VB.NETからこのSDKを使うことによりPM100のスキャナーを制御できます。

ユーザーレイヤー
SDK
ハードウェアインタフェースレイヤー

2.1.1 ユーザーレイヤー

ユーザーレイヤーはSDKを通してEVC++, C#, VB.NET で作られるスキャナーに関するユーザープログラムを意味します。

2.1.2 SDK

SDKは、PM100 2次元スキャナー制御のための各種定義、API、メッセージから構成されます。

(SDKの構成)

BCDApi.h : [BCDCore.dll]のヘッダーファイル。各定義、構造体、API関数を含んでいます。

BCDCore.dll : ユーザーレイヤー側とリンクするためのDLLです。(直接PDAのWindowsの中にあります)

2.1.3 ハードウェアインタフェースレイヤー

ハードウェアインタフェースレイヤーはPM100 スキャナー制御のためのハードウェア層です。

3 定義、API関数、メッセージ

3.1 APIの構成

3.1.1 スキャナー関数

BcdOpen

スキャナーをオープンし、ハンドルを取得します。

HANDLE BcdOpen(Void)

復帰値

HANDLE

パラメータ

Void

BcdClose

スキャナーをクローズし、ハンドルをクローズします。

Void BcdClose(Void)

復帰値

Void

パラメータ

Void

BcdEnable

スキャナーを有効化/無効化します。

Void BcdEnable(BOOL bEnable)

復帰値

Void

パラメータ

bEnable

この変数でスキャナーの有効/無効を設定します。
[TRUE]を設定するとスキャナーは有効となります。

BcdIsEnabled

スキャナー状態(有効/無効)を取得します。

BOOL BcdIsEnabled(Void)

復帰値

BOOL : スキャナーが有効の場合、[TRUE]が返ります。

パラメータ

Void

BcdSetHWND

スキャン結果を受け取るためのアプリケーションのWindowハンドルを設定します。

Void BcdSetHWND(HWND hWnd)

復帰値

Void

パラメータ

hWnd

アプリケーションWindowハンドル

BcdSetTerminator

スキャンが成功した時の終了コードを設定します。

Void BcdSetTerminator(BYTE cTerminator)

復帰値

Void

パラメータ

cTerminator

TERMINATOR_NONE: 無し

TERMINATOR_CRLF: [CR]コード+[LF]コード

TERMINATOR_SPACE: [SPACE]コード

TERMINATOR_TAB: [TAB]コード

BcdSetResultType

スキャンが成功した時のスキャン結果通知タイプを設定します。

Void BcdSetResultType(DWORD dwType)

復帰値

Void

パラメータ

dwType

RESULT_USERMSG: スキャナー使用アプリケーションのWindowハンドルにユーザーメッセージで伝える

RESULT_KBDMSG: キーボードメッセージで伝える

RESULT_COPYPASTE: コピー&ペーストで伝える

BcdGetResultType

スキャン成功時のスキャン結果通知タイプを取得します。

BYTE BcdGetResultType(Void)

復帰値

RESULT_USERMSG

RESULT_KBDMSG

RESULT_COPYPASTE

パラメータ

Void

BcdEnableBeep

スキャン後のビープ音を有効にします。

Void BcdEnableBeep(BOOL bBeep)

復帰値

Void

パラメータ

bBeep:[TRUE]の場合、ビープ音を鳴動します。

BcdEnableLED

スキャン後のLEDの使用を決定します。

Void BcdEnableLED(BOOL bLED)

復帰値

Void

パラメータ

bLED:[TRUE]の場合、スキャン後にLEDを点灯します。

BcdEnablePrefix

スキャン後のプレフィックスの使用を決定します。

Void BcdEnablePrefix(BOOL bPrefix)

復帰値

Void

パラメータ

bPrefix:[TRUE]の場合、スキャン後のプレフィックスを使用します。

BcdSetPrefix

プレフィックスを使用する場合、プレフィックス文字列を設定します。

Void BcdSetPrefix(TCHAR *szPrefix)

復帰値

Void

パラメータ

szPrefix:プレフィックス文字列

BcdGetPrefix

プレフィックスを使用する場合、プレフィックス文字列を取得します。

Void BcdGetPrefix(TCHAR *szPrefix)

復帰値

Void

パラメータ

szPrefix:プレフィックス文字列を読み込む変数

BcdEnablePostfix

スキャン後のポストフィックスの使用を決定します。

Void BcdEnablePostfix(BOOL bPostfix)

復帰値

Void

パラメータ

bPostfix: [TRUE]の場合、スキャン後にポストフィックスを使用します。

BcdSetPostfix

ポストフィックスを使用する場合、ポストフィックス文字列を設定します。

Void BcdSetPostfix(TCHAR *szPostfix)

復帰値

Void

パラメータ

szPostfix: ポストフィックス文字列

BcdGetPostfix

ポストフィックスを使用する場合、ポストフィックス文字列を取得します。

Void BcdGetPostfix(TCHAR *szPostfix)

復帰値

Void

パラメータ

szPostfix: ポストフィックス文字列を読み込む変数

BcdSetTrigTimeout

スキャン開始とスキャン終了の間隔を設定します。

Void BcdSetTrigTimeout(DWORD dwSeconds)

復帰値

Void

パラメータ

dwSeconds: 間隔時間(単位: ミリ秒)

BcdEanbleAutoScan

オートスキャンの使用を決定します。

Void BcdEanbleAutoScan(BOOL bAutoScan)

復帰値

Void

パラメータ

bAutoScan: [TRUE]なら、オートスキャンは有効です。

BcdSetAutoScanInterval

オートスキャンの間隔を設定します。

Void BcdSetAutoScanInterval(DWORD dwInterval)

復帰値

Void

パラメータ

dwInterval: オートスキャン間隔 (単位: ミリ秒)

BcdSetConfig

[BCD_CONFIG]の値を設定します。

Void BcdSetConfig(PBCD_CONFIG pbcdConfig)

復帰値

Void

パラメータ

pbcdConfig: [BCD_CONFIG]へのポインタ

BcdGetConfig

[BCD_CONFIG]を取得します。

Void BcdGetConfig(PBCD_CONFIG pbcdConfig)

復帰値

Void

パラメータ

pbcdConfig: [BCD_CONFIG]へのポインタ

BcdGetBarTypeString

バーコードタイプからバーコードタイプの文字列を取得します。

LPWSTR BcdGetBarTypeString(BAR_TYPE barType)

復帰値

LPWSTR: バーコードタイプ文字列

パラメータ

barType: バーコードタイプ (=0~46が有効です。)

BcdGetEnableLength

指定されたバーコードタイプがバーコード長を返すか否かを識別します。

BOOL BcdGetEnableLength(BAR_TYPE barType)

復帰値

BOOL: [TRUE]なら指定されたバーコードタイプはバーコード長を返します。

パラメータ

barType: バーコードタイプ (=0~46が有効です。)

BcdGetBarLength

指定されたバーコードタイプの最大長と最小長を取得します。

Void BcdGetBarLength(BAR_TYPE barType, int* puMin, int* puMax)

復帰値

Void

パラメータ

barType: バーコードタイプ (=0～46が有効です。)

puMin: 最小長

puMax: 最大長

BcdSetBarLength

指定されたバーコードタイプの最大長と最小長を設定します。

BOOL BcdSetBarLength(BAR_TYPE barType, int uMin, int uMax)

復帰値

BOOL: [TRUE]の場合、正常終了。

パラメータ

barType: バーコードタイプ (=0～46が有効です。)

uMin: 最小長設定値

uMax: 最大長設定値

(注意)

- ① uMin値およびuMax値は、BcdGetDefaultBarLength関数にて取得できるデフォルト最小長～デフォルト最大長の範囲内で指定して下さい。

BcdTransmitBarID

スキャン後のバーコードIDの送信を有効にします。

Void BcdTransmitBarID(BOOL bTransmit)

復帰値

Void

パラメータ

bTransmit: [TRUE]ならバーコードIDを送信します。(バーコードIDがバーコードデータの前に付加されます。)

BcdGetBarID

指定されたバーコードタイプのIDを取得します。

BYTE BcdGetBarID(BAR_TYPE barType)

復帰値

BYTE: 取得したバーコードID

パラメータ

barType: バーコードタイプ (=0～46が有効です。)

BcdSetBarID

指定したバーコードタイプにIDを設定します。

Void BcdSetBarID(BAR_TYPE barType, BYTE barID)

復帰値

Void

パラメータ

barType:バーコードタイプ (=0～46が有効です。)

barID:バーコードID設定値

3.1.2 エンジン関数

BcdSetDefault

工場出荷モードにリセットします。

Void BcdSetDefault(void)

復帰値

Void

パラメータ

Void

BcdEnableBarcode

指定されたバーコードタイプを有効化/無効化します。

Void BcdEnableBarcode(BAR_TYPE barType, BOOL bEnable)

復帰値

Void

パラメータ

barType: バーコードタイプ (=0~46が有効です。)

bEnable: [TRUE]の場合、有効化

BcdEnableALLBarcode

全バーコードタイプを有効化/無効化します。

BOOL BcdEnableBarcode(BOOL bEnable)

復帰値

BOOL: [TURE]の場合正常終了。(4~7秒後に復帰します。)

パラメータ

bEnable: [TRUE]の場合、有効化

BcdIsBarcodeEnabled

指定されたバーコードタイプの有効/無効状態を取得します。

BOOL BcdIsBarcodeEnabled(BAR_TYPE barType)

復帰値

BOOL

パラメータ

barType: バーコードタイプ (=0~46が有効です。)

BcdSetBarFlag

指定されたバーコードタイプの詳細構成情報を設定します。

Void BcdSetBarFlag(BAR_TYPE barType, int nIndex, BOOL bEnable)

復帰値

Void

パラメータ

barType: バーコードタイプ (=0~46が有効です。)

nIndex: 詳細設定のインデックス

bEnable: [TRUE]の場合、そのインデックスの詳細設定は有効化

BcdGetBarFlagArrayPtr

指定されたバーコードタイプの詳細構成情報を取得します。

PBAR_FLAG BcdGetBarFlagArrayPtr(BAR_TYPE barType)

復帰値

PBAR_FLAG

パラメータ

barType:バーコードタイプ (=0～46が有効です。)

BcdTrigTurnOn

スキャンを開始/終了します。

Void BcdTrigTurnOn(BOOL bTrigger)

復帰値

Void

パラメータ

bTrigger:[TRUE]の場合、スキャンを開始します。

BcdGetResult

[WM_SCANNED]メッセージを取得した時に、この関数にて[SCAN_RESULT]構造体を取得します。

PSCAN_RESULT BcdGetResult(void)

復帰値

PSCAN_RESULT

パラメータ

Void

BcdGetString

現在の[SCAN_RESULT]からバーコード値を取得します。

LPWSTR BcdGetString(void)

復帰値

LPWSTR

パラメータ

Void

BcdGetBarType

現在の[SCAN_RESULT]からバーコードタイプを取得します。

BAR_TYPE BcdGetBarType(void)

復帰値

BAR_TYPE

パラメータ

Void

BcdGetLength

現在の[SCAN_RESULT]からバーコードデータの長さを取得します。

UINT BcdGetBarType(void)

復帰値

UINT

パラメータ

Void

BcdGetVersion

スキャナーのデコーダーのバージョンを取得します。

Void BcdGetVersion(TCHAR *szVersion)

復帰値

Void

パラメータ

szVersion: スキャナーデコーダーのバージョン

BcdTriggerEnable

スキャントリガーを有効化/無効化します。

Void BcdTriggerEnable (BOOL bTrigger)

復帰値

Void

パラメータ

bTrigger: [TURE]の場合、スキャントリガーが有効です。

BcdIsTriggerEnabled

スキャントリガー状態を取得します。

BOOL BcdIsTriggerEnabled (Void)

復帰値

BOOL: [TRUE]の場合、スキャントリガーは有効です。

パラメータ

Void

BcdSetContinuousMode

連続スキャンモードの設定を行います。

Void BcdSetContinuousMode(BOOL bEnable)

復帰値

Void

パラメータ

bEnable: TRUEで連続スキャンモード FALSEで通常モード

BcdGetContinuousMode

連続スキャンモードの設定を取得します。

BOOL BcdGetContinuousMode(void)

復帰値

TRUEで連続スキャンモード FALSEで通常モード

パラメータ

Void

3.1.3 スキャナー2D IT5300専用関数

このスキャナー以外のデバイス(1D IS4813)でこの関数を使用しても正常に機能しません、この時GetLastError 関数は 'ERROR_NOT_SUPPORTED' を返します。

BcdGetDefaultBarLength

指定されたバーコードタイプのデフォルトの最大長と最小長を取得します。

Void BcdGetDefaultBarLength(BAR_TYPE barType, int* puMin, int* puMax)

復帰値

Void

パラメータ

barType: バーコードタイプ (=0~46が有効です。)

puMin: デフォルト最小長

puMax: デフォルト最大長

BcdGetOCRFlagEX

OCRバーコードタイプの詳細設定情報を取得します。

POCR_FLAG BcdGetOCRFlag(void)

復帰値

POCR_FLAG: OCRコードの詳細設定情報

パラメータ

Void

BcdSetOCRFlag

OCRバーコードタイプの詳細設定情報を設定します。

Void BcdSetOCRFlag(OCR_FLAG OCRFlag)

復帰値

Void

パラメータ

OCR_FLAG: OCRコードの詳細設定情報

BcdGetDecoderVersion

.....

スキャナーのデコーダーのバージョンを取得します。(BcdGetVersion() と同じ関数)

Void BcdGetDecoderVersion(TCHAR *szVersion)

復帰値

Void

パラメータ

szVersion: スキャナーデコーダーのバージョン

BcdGetApiVersion

.....

スキャナーのデコーダーのAPIバージョンを取得します。

Void BcdGetApiVersion(TCHAR *szVersion)

復帰値

Void

パラメータ

szVersion: スキャナーデコーダーのAPIバージョン

BcdGetScaDriverVersion

.....

スキャナーのドライバのバージョンを取得します。

Void BcdGetScaDriverVersion(TCHAR *szVersion)

復帰値

Void

パラメータ

szVersion: スキャナードライバのバージョン

BcdSetSCANPCLK

.....

スキャナーの[Pclock]を設定します。

Void BcdSetSCANPCLK(DWORD PCLK)

復帰値

Void

パラメータ

PCLK: [0]の場合[Pclock]は12MHz、[1]以上の場合[Pclock]は24MHz

BcdIs24MHzPCLK

.....

スキャナーの[Pclock]を取得します。

This function is get scanner's Pclock.

BOOL BcdIs24MHzPCLK(Void)

復帰値

BOOL: [TRUE]の場合、[Pclock]は24MHz、[FALSE]の場合[Pclock]は12MHz.

パラメータ

Void

BcdEnableCenterWindow

.....

[center window]機能の使用を決定します。

[center window]機能を使用する場合はスキャン時にバーコード位置が目標領域の中心でなければならない。

Void BcdEnableCenterWindow(BOOL bUse)

復帰値

Void

パラメータ

bUse:[TRUE]の場合、[Center window]機能を使用します。

BcdSetDecodeMode

.....

スキャナーのデコードモードを決定します。

Void BcdSetDecodeMode(BYTE btMode)

復帰値

Void

パラメータ

btMode

STANDARD :標準のモード

ADVANCED_LINEAR :1次元バーコードに適したモード

QUICK_OMNI :標準より早いモード

BcdSetDelayDecoding

.....

遅延時間後にデコードする場合の遅延時間を設定します。

Void BcdSetDelayDecoding(DWORD btDelay)

復帰値

Void

パラメータ

btDelay:遅延時間(単位:ミリ秒)

BcdCreateDIBSection

.....

イメージデータを書込むために[BmpFormat]からビットマップハンドルを作成します。

HBITMAP BcdCreateDIBSection(BmpFormat * pBmp, LONG IWidth, LONG IHeight)

復帰値

HBITMAP : Bitmap handle.

パラメータ

pBmp:[BmpFormat]のポインタ

IWidth: イメージの幅

IHeight: イメージの高さ

BcdDestroyDIBSection

ビットマップハンドルを削除します。

Void BcdDestroyDIBSection(HBITMAP hBitmap)

復帰値

Void

パラメータ

hBitmap: ビットマップハンドル

この関数は[BcdCreateDIBSection]関数とペアで使用しなければならない。

BcdGetCaptureImage

イメージデータを取得します。

BOOL BcdGetCaptureImage(PBYTE pImageBuffer,DWORD * pImageSize)

復帰値

BOOL:[TRUE]の場合、正常復帰

パラメータ

pImageBuffer: イメージデータバッファへのポインタ

pImageSize: イメージデータサイズへのポインタ

BcdImageStreamInit

イメージプレビューの準備を行う。

BOOL BcdImageStreamInit(Void)

復帰値

BOOL : [TRUE]の場合、正常復帰

パラメータ

Void

BcdImageStreamStart

イメージプレビューを開始します。

BOOL BcdImageStreamStart(Void)

復帰値

BOOL : [TRUE]の場合、正常復帰

パラメータ

Void

BcdImageStreamRead

連続的なイメージプレビューを維持します。

BOOL BcdImageStreamRead(unsigned char * pImageBuffer, DWORD * pdwSize)

復帰値

BOOL : [TRUE]の場合、正常復帰

パラメータ

pImageBuffer: イメージデータのバッファへのポインタ

pdwSize: イメージのサイズ

BcdImageStreamStop

イメージプレビューを停止します。

BOOL BcdImageStreamStop(Void)

復帰値

BOOL : [TRUE]の場合、正常復帰

パラメータ

Void

BcdAimerOn

照準の使用を決定します。

BOOL BcdAimerOn(BOOL enable)

復帰値

BOOL : [TRUE]の場合、正常復帰

パラメータ

Enable: [TRUE]の場合、照準をONします。[FALSE]の場合OFFします。

BcdIlluminationOn

照明の使用を決定します。

BOOL BcdIlluminationOn(BOOL enable)

復帰値

BOOL : [TRUE]の場合、正常復帰

パラメータ

Enable: [TRUE]の場合、照明をONします。[FALSE]の場合OFFします。

BcdSetScanLightsMode

イメージキャプチャあるいはプレビューのとき照準と照明の使用を決定します。(この関数は[Image Mode]時に使用できます。)

BOOL BcdSetScanLightsMode(SCANLIGHTSMODE mode)

復帰値

BOOL : [TRUE]の場合、正常復帰

パラメータ

mode

SCANLIGHTSMODE_OFF= 0,	: 照準、照明ともにOFF
SCANLIGHTSMODE_ILLUM_ONLY_ON,	: 照明のみON
SCANLIGHTSMODE_AIMER_ONLY_ON,	: 照準のみON
SCANLIGHTSMODE_ON,	: 照準、照明ともにON

BcdGetEngineConfig

スキャンエンジンの構成を取得します。

BOOL BcdGetEngineConfig(ENGINECONFIG * pEngineConfig)

復帰値

BOOL : [TRUE]の場合、正常復帰

パラメータ

pEngineConfig: [ENGINECONFIG]へのポインタ。([ENGINECONFIG]参照)

BcdSetScanImageMode

スキャナーモードの使用を決定します。(スキャンモードあるいはイメージモード)

Void BcdSetScanImageMode(int nMode)

復帰値

Void

パラメータ

nMode

MODE_SCAN	=0x00	: スキャンモード
MODE_IMAGE	=0x01	: イメージモード

3.1.4 スキャナー1D IS4813専用関数

このスキャナー以外のデバイス(2D IT5300)でこの関数を使用しても正常に機能しません、この時GetLastError 関数は'ERROR_NOT_SUPPORTED'を返します。

BcdSetExtraReads

拡張スキャン回数を設定します。

Void BcdSetExtraReads(BAR_TYPE barType, int nExtraRead)

復帰値

Void

パラメータ

barType: バーコードタイプを指定

nExtraRead: 読み込み回数

BcdGetExtraReads

拡張スキャン回数を取得します。

Void BcdGetExtraReads(BAR_TYPE barType, int* nExtraRead)

復帰値

Void

パラメータ

barType: バーコードタイプを指定

nExtraRead: 読み込み回数取得変数ポインタ

BcdSetExtraTime

拡張スキャン読み込み時間を設定します。

Void BcdSetExtraTime(BAR_TYPE barType, int nExtraTime)

復帰値

Void

パラメータ

barType: バーコードタイプを指定

nExtraTime: 読み込み時間

BcdSetExtraTimeEX

拡張スキャン読み込み時間を設定します。(拡張機能)

BOOL BcdSetExtraTime (BAR_TYPE barType, int nExtraTime)

復帰値

TRUE: 正常終了 FALSE: 拡張スキャン時間が設定されていません。
(Last error で内容を確認してください)

パラメータ

barType: バーコードタイプを指定

nExtraTime: 読み込み時間

BcdGetExtraTime

拡張スキャン読み込み時間を取得します。

Void BcdGetExtraTime (BAR_TYPE barType, int* nExtraTime)

復帰値

Void

パラメータ

barType: バーコードタイプを指定

nExtraTime: 読み込み時間取得変数

BcdSetStitchingEnable
.....

スティッチングモードを設定します。

Void BcdSetStitchingEnable(BOOL bStitching)

復帰値

Void

パラメータ

bStitching: TRUE:スティッチングモードを使用する

BcdSetStitchingEnable
.....

ノンプロポーショナルモードを設定します。

Void BcdSetAllowNonProportional (BOOL bAllow)

復帰値

Void

パラメータ

bAllow: TRUE: ノンプロポーショナルモードを使用する

BcdSetStitchingLevel
.....

スティッチングレベルを設定します。

Void BcdSetStitchingLevel (DWORD dwLevel)

復帰値

Void

パラメータ

dwLevel: スティッチングレベル

BcdSetStitchingLevel
.....

誤差レベルを設定します。

Void BcdSetRoundingUncertaintyLevel (DWORD dwLevel)

復帰値

Void

パラメータ

dwLevel: 誤差レベル

3.1.4 その他の関数

BcdGetModelName

デバイスモデル名を取得します。

LPWSTR BcdGetModelName(Void)

復帰値

LPWSTR : Model Name.

パラメータ

Void

BcdGetModelSerialNumber

シリアル番号を取得します。

LPWSTR BcdGetModelSerialNumber(Void)

復帰値

LPWSTR : シリアル番号

パラメータ

Void

BcdGetBeepScale

ビープ音の音階を取得します。

DWORD BcdGetBeepScale(PBCD_BEEP_LEVEL_SETTING)

復帰値

DWORD : [0]の場合、正常復帰

[0]以外の場合、異常(エラーコードは、「winerror.h」で定義)

パラメータ

PBCD_BEEP_LEVEL_SETTING

BcdSetBeepScale

ビープ音の音階を決定します。

DWORD BcdSetBeepScale(PBCD_BEEP_LEVEL_SETTING)

復帰値

DWORD : [0]の場合、正常復帰

[0]以外の場合、異常(エラーコードは、「winerror.h」で定義)

パラメータ

PBCD_BEEP_LEVEL_SETTING

3.2 関数と関数ポインタ

各関数は関数ポインタを持っています。

下表は各関数と関数ポインタの説明です。

BcdOpen	typedef HANDLE (WINAPI *PFN_BcdOpen)(void);
BcdClose	typedef void (WINAPI *PFN_BcdClose)(void);
BcdEnable	typedef void (WINAPI *PFN_BcdEnable)(BOOL);
BcdIsEnabled	typedef BOOL (WINAPI *PFN_BcdIsEnabled)(void);
BcdSetHWND	typedef void (WINAPI *PFN_BcdSetHWND)(HWND);
BcdSetTerminator	typedef void (WINAPI *PFN_BcdSetTerminator)(BYTE);
BcdSetResultType	typedef void (WINAPI *PFN_BcdSetResultType)(DWORD);
BcdGetResultType	typedef BYTE (WINAPI *PFN_BcdGetResultType)(void);
BcdEnableBeep	typedef void (WINAPI *PFN_BcdEnableBeep)(BOOL);
BcdEnableLED	typedef void (WINAPI *PFN_BcdEnableLED)(BOOL);
BcdEnablePrefix	typedef void (WINAPI *PFN_BcdEnablePrefix)(BOOL);
BcdSetPrefix	typedef BOOL (WINAPI *PFN_BcdSetPrefix)(TCHAR*);
BcdGetPrefix	typedef void (WINAPI *PFN_BcdGetPrefix)(TCHAR*);
BcdEnablePostfix	typedef void (WINAPI *PFN_BcdEnablePostfix)(BOOL);
BcdSetPostfix	typedef BOOL (WINAPI *PFN_BcdSetPostfix)(TCHAR*);
BcdGetPostfix	typedef void (WINAPI *PFN_BcdGetPostfix)(TCHAR*);
BcdSetTrigTimeout	typedef void (WINAPI *PFN_BcdSetTrigTimeout)(DWORD);
BcdEanbleAutoScan	typedef void (WINAPI *PFN_BcdEanbleAutoScan)(BOOL);
BcdSetAutoScanInterval	typedef void (WINAPI *PFN_BcdSetAutoScanInterval)(DWORD);
BcdSetConfig	typedef void (WINAPI *PFN_BcdSetConfig)(PBCD_CONFIG);
BcdGetConfig	typedef void (WINAPI *PFN_BcdGetConfig)(PBCD_CONFIG);
BcdGetBarTypeString	typedef LPWSTR (WINAPI *PFN_BcdGetBarTypeString)(BAR_TYPE);
BcdGetEnableLength	typedef BOOL (WINAPI *PFN_BcdGetEnableLength)(BAR_TYPE);
BcdGetBarLength	typedef void (WINAPI *PFN_BcdGetBarLength)(BAR_TYPE, int*, int*);
BcdSetBarLength	typedef BOOL (WINAPI *PFN_BcdSetBarLength)(BAR_TYPE, int, int);
BcdTransmitBarID	typedef void (WINAPI *PFN_BcdTransmitBarID)(BOOL);
BcdGetBarID	typedef BYTE (WINAPI *PFN_BcdGetBarID)(BAR_TYPE);
BcdSetBarID	typedef void (WINAPI *PFN_BcdSetBarID)(BAR_TYPE, BYTE);
BcdSetDefault	typedef void (WINAPI *PFN_BcdSetDefault)(void);
BcdEnableBarcode	typedef void (WINAPI *PFN_BcdEnableBarcode)(BAR_TYPE, BOOL);
BcdEnableALLBarcode	typedef BOOL (WINAPI *PFN_BcdEnableALLBarcode)(BOOL);
BcdIsBarcodeEnabled	typedef BOOL (WINAPI *PFN_BcdIsBarcodeEnabled)(BAR_TYPE);
BcdSetBarFlag	typedef void (WINAPI *PFN_BcdSetBarFlag)(BAR_TYPE, int, BOOL);
BcdGetBarFlagArrayPtr	typedef PBAR_FLAG (WINAPI *PFN_BcdGetBarFlagArrayPtr)(BAR_TYPE);
BcdTrigTurnOn	typedef void (WINAPI *PFN_BcdTrigTurnOn)(BOOL);
BcdGetResult	typedef PSCAN_RESULT (WINAPI *PFN_BcdGetResult)(void);
BcdGetString	typedef LPWSTR (WINAPI *PFN_BcdGetString)(void);
BcdGetBarType	typedef BAR_TYPE (WINAPI *PFN_BcdGetBarType)(void);
BcdGetLength	typedef UINT (WINAPI *PFN_BcdGetLength)(void);
BcdGetVersion	typedef void (WINAPI *PFN_BcdGetVersion)(TCHAR*);
BcdTriggerEnable	typedef void (WINAPI *PFN_BcdTriggerEnable)(BOOL);
BcdIsTriggerEnabled	typedef BOOL (WINAPI *PFN_BcdIsTriggerEnabled)(void);
BcdSetContinuousMode	typedef void (WINAPI *PFN_BcdSetContinuousMode)(BOOL);
BcdGetContinuousMode	typedef BOOL (WINAPI *PFN_BcdGetContinuousMode)(void);
	スキャナモジュール 2D IT5300専用関数
BcdGetDefaultBarLength	typedef void (WINAPI *PFN_BcdGetDefaultBarLength)(BAR_TYPE, int*, int*);
BcdGetOCRFlagEX	typedef POOCR_FLAG (WINAPI *PFN_BcdGetOCRFlag)(void);
BcdSetOCRFlag	typedef void (WINAPI *PFN_BcdSetOCRFlag)(OCR_FLAG);
BcdGetDecoderVersion	typedef void (WINAPI *PFN_BcdGetDecoderVersion)(TCHAR*);

BcdGetApiVersion	typedef void (WINAPI *PFN_BcdGetApiVersion)(TCHAR*);
BcdGetScaDriverVersion	typedef void (WINAPI *PFN_BcdGetScaDriverVersion)(TCHAR*);
BcdSetSCANPCLK	typedef void (WINAPI *PFN_BcdSetSCANPCLK)(DWORD);
BcdIs24MHzPCLK	typedef BOOL (WINAPI *PFN_BcdIs24MHzPCLK)(void);
BcdEnableCenterWindow	typedef void (WINAPI *PFN_BcdEnableCenterWindow)(BOOL);
BcdSetDecodeMode	typedef void (WINAPI *PFN_BcdSetDecodeMode)(BYTE);
BcdSetDelayDecoding	typedef void (WINAPI *PFN_BcdSetDelayDecoding)(DWORD);
BcdCreateDIBSection	typedef HBITMAP (WINAPI *PFN_BcdCreateDIBSection)(BmpFormat*);
BcdDestroyDIBSection	typedef void (WINAPI *PFN_BcdDestroyDIBSection)(HBITMAP);
BcdGetCaptureImage	typedef BOOL (WINAPI *PFN_BcdGetCaptureImage)(PBYTE, DWORD*);
BcdImageStreamInit	typedef BOOL (WINAPI *PFN_BcdImageStreamInit)(void);
BcdImageStreamStart	typedef BOOL (WINAPI *PFN_BcdImageStreamStart)(void);
BcdImageStreamRead	typedef BOOL (WINAPI *PFN_BcdImageStreamRead)(unsigned char*, DWORD*);
BcdImageStreamStop	typedef BOOL (WINAPI *PFN_BcdImageStreamStop)(void);
BcdAimerOn	typedef BOOL (WINAPI *PFN_BcdAimerOn)(BOOL);
BcdIlluminationOn	typedef BOOL (WINAPI *PFN_BcdIlluminationOn)(BOOL);
BcdSetScanLightsMode	typedef BOOL (WINAPI *PFN_BcdSetScanLightsMode)(SCANLIGHTSMODE);
BcdGetEngineConfig	typedef BOOL (WINAPI *PFN_BcdGetEngineConfig)(ENGINECONFIG*);
BcdSetScanImageMode	typedef void (WINAPI *PFN_BcdSetScanImageMode)(int);
	スキャナモジュール 1D IS4813専用関数
BcdSetExtraReads	typedef void (WINAPI *PFN_BcdSetExtraReads)(BAR_TYPE, int);
BcdGetExtraReads	typedef void (WINAPI *PFN_BcdGetExtraReads)(BAR_TYPE, int*);
BcdSetExtraTime	typedef void (WINAPI *PFN_BcdSetExtraTime)(BAR_TYPE, int);
BcdGetExtraTime	typedef void (WINAPI *PFN_BcdGetExtraTime)(BAR_TYPE, int*);
BcdSetStitchingEnable	typedef void (WINAPI *PFN_BcdSetStitchingEnable)(BOOL);
BcdSetAllowNonProportional	typedef void (WINAPI *PFN_BcdSetAllowNonProportional)(BOOL);
BcdSetStitchingLevel	typedef void (WINAPI *PFN_BcdSetStitchingLevel)(DWORD);
BcdSetRoundingUncertaintyLevel	typedef void (WINAPI *PFN_BcdSetRoundingUncertaintyLevel)(DWORD);
	その他関数
BcdGetModelName	typedef LPWSTR (WINAPI *PFN_BcdGetModelName)(void);
BcdGetModelSerialNumber	typedef LPWSTR (WINAPI *PFN_BcdGetModelSerialNumber)(void);
BcdGetBeepScale	typedef DWORD (WINAPI *PFN_BcdGetBeepScale)((PBCD_BEEP_LEVEL_SETTING);
BcdSetBeepScale	typedef DWORD (WINAPI *PFN_BcdSetBeepScale)((PBCD_BEEP_LEVEL_SETTING);

3.3 スキャナ-APIの使い方

アプリケーションは関数ポインタを使ってプログラミングできます。

下記コードは関数ポインタの使い方のサンプルです。

```
//      Include scanner's heder file.
#include "../BCDapi.h"

//      Load library and get function's process address.
HINSTANCE hInstance = NULL;
PFN_BcdEnable pBcdEnable = NULL;
hInstance=LoadLibrary(_T("BcdCore.dll"));
if(hInstance != NULL)
    pBcdEnable = (PFN_BcdEnable)GetProcAddress(hInstance, L"BcdEnable");

// IF you want use to scanner.
pBcdEnable(TRUE);
FreeLibrary(hInstance);
```

3.4 構造体と定義の構成

下記の記述がSDKを使う上での構造体と定義です。

3.4.1 スキャナ構造体と定義

About of scanner configurations.

```
//      define for user message
#define WM_SCANNED    WM_USER+0x7777  // Transmit message after scan.
#define WM_IMAGE      WM_USER+0x7778  // Transmit message after capture.

//      define for terminate : Terminate code is add the last character of scan value.
#define TERMINATOR_NONE    0x00  //      None.
#define TERMINATOR_CRLF    0x01  //      CR + LF
#define TERMINATOR_SPACE   0x02  //      SPACE
#define TERMINATOR_TAB     0x03  //      TAB

//      define for scan result type : How to transmit after scan.
#define RESULT_USERMSG     0x00  //      Each application get the WM_SCANNED.
#define RESULT_KBDMSG      0x01  //      Use to KBDMSG.
#define RESULT_COPYPASTE   0x02  //      Use to COPYPASTE.
#define RESULT_EVENT       0x03  //      NOT USE.
#define RESULT_CALLBACK    0x04  //      NOT USE.

//      define for mode scan or image : Select the decoding mode or image mode.
#define MODE_SCAN          0x00
#define MODE_IMAGE         0x01

//      define for Decode Mode : Select the engine's decode mode.
#define STANDARD           0x00  //      This decode mode is standard
#define ADVANCED_LINEAR    0x02  //      This decode mode is fit of 1D barcode.
#define QUICK_OMNI         0x01  //      This decode mode is fast more than standard.
```

Enums section

```
//      scan status : WM_SCANNED message' s LPARAM value.
```

```
typedef enum _STATUS_BCD
```

```
{
```

```
    ERROR_NO_ERROR = 0,      //      Scan success.
```

```
    ERROR_NO_READ,          //      Scan fail.
```

```
    ERROR_UNKNOWN_TYPE,    //      NOT USE.
```

```
    ERROR_WRONG_DATA,      //      NOT USE.
```

```
    ERROR_NO_SCAN_DATA,    //      NOT USE.
```

```
    ERROR_SCAN_TIMEOUT,    //      NOT USE.
```

```
} ERROR_BSCANNER;
```

```
//      barcode type
```

```
typedef enum _BAR_TYPE
```

```
{
```

```
    //      2D scanner(IT5300)
```

```
    BAR_SYM_AZTEC          = 0,
```

```
    BAR_SYM_CODABAR,
```

```
    BAR_SYM_CODE11,
```

```
    BAR_SYM_CODE128,
```

```
    BAR_SYM_CODE39,
```

```
    BAR_SYM_CODE49,
```

```
    BAR_SYM_CODE93,
```

```
    BAR_SYM_COMPOSITE,
```

```
    BAR_SYM_DATAMATRIX,
```

```
    BAR_SYM_EAN8,
```

```
    BAR_SYM_EAN13,
```

```
    BAR_SYM_INT25,
```

```
    BAR_SYM_MAXICODE,
```

```
    BAR_SYM_MICROPDF,
```

```
    BAR_SYM_OCR,
```

```
    BAR_SYM_PDF417,
```

```
    BAR_SYM_POSTNET,
```

```
    BAR_SYM_QR,
```

```
    BAR_SYM_RSS,
```

```
    BAR_SYM_UPCA,
```

BAR_SYM_UPCE0,
BAR_SYM_BPO,
BAR_SYM_CANPOST,
BAR_SYM_AUSPOST,
BAR_SYM_IATA25,
BAR_SYM_CODABLOCK,
BAR_SYM_JAPOST,
BAR_SYM_PLANET,
BAR_SYM_DUTCHPOST,
BAR_SYM_MSI,
BAR_SYM_TLCODE39,
BAR_SYM_TRIOPTIC,
BAR_SYM_CODE32,
BAR_SYM_MATRIX25,
BAR_SYM_PLESSEY,
BAR_SYM_CHINAPOST,
BAR_SYM_KOREAPOST,
BAR_SYM_TELEPEN,
BAR_SYM_CODE16K,
BAR_SYM_POSICODE,
BAR_SYM_GS1_128,
BAR_SYM_USPS4CB,
BAR_SYM_IDTAG,
BAR_SYM_ISBT,
BAR_SYM_STRT25,
BAR_SYM_COUPONCODE,
BAR_SYM_LABEL,
BAR_SYM_HANXIN,
BAR_SYM_GRIDMATRIX,
BAR_SYM_UPCE1,
BAR_NUM_SYMBOLOGIES,

// 1D scanner(IS4813)
IS4813_BAR_UNKNOWN = 0,
IS4813_UPC_A,
IS4813_UPC_E,
IS4813_EAN_8,

```

IS4813_EAN_13,
IS4813_CODE_39,
IS4813_CODE_93,
IS4813_CODE_ITF_2OF5,
IS4813_CODE_128,
IS4813_CODABAR,
IS4813_MSI_PLESSEY,          //      0x0A
IS4813_UK_PLESSEY,
IS4813_CODE_11,
IS4813_MATRIX_2OF5,
IS4813_STANDARD_2OF5,
IS4813_AIRLINE_2OF5_13_DIGIT,
IS4813_AIRLINE_2OF5_15_DIGIT,
IS4813_HONGKONG_2OF5,       //      0x11
IS4813_NEC_2OF5,
IS4813_TELEPEN,
IS4813_TRIOPTIC,
IS4813_GS1_DATABAR_14,
IS4813_GS1_DATABAR_LIMITED,
IS4813_GS1_DATABAR_EXPANDED,
IS4813_BAR_ALL              //      0x18

```

```

]BAR_TYPE;

```

```

//      scanner order : CODE_ID of scan driver IOCTL.

```

```

typedef enum

```

```

{

```

```

    //      common orders

```

```

    ORDER_DEBUG00 = 0,

```

```

    ORDER_DEBUG01,

```

```

    ORDER_DEBUG02,

```

```

    ORDER_PARAM_INIT,

```

```

    ORDER_HWTRIGGERON,

```

```

    ORDER_HWTRIGGEROFF,

```

```

    ORDER_SETHWND,

```

```

    ORDER_SCANNER_ENABLE,

```

```

    ORDER_SCANNER_DISABLE,

```

```

    ORDER_BEEP_ENABLE,

```


ORDER_BEEP_DISABLE,
ORDER_VIBRATE_ENABLE,
ORDER_VIBRATE_DISABLE,
ORDER_LED_ENABLE,
ORDER_LED_DISABLE,
ORDER_GETBARTYPE,
ORDER_GETBARSTRING,
ORDER_GETRESULT,
ORDER_AUTOSCAN_ON,
ORDER_AUTOSCAN_OFF,
ORDER_IENABLED,
ORDER_TRIGGER_TIMEOUT,
ORDER_SCANNER_DEFAULT,
ORDER_SCANNER_GETPARAM,
ORDER_SCANNER_AUTOSCANINTERVAL,
ORDER_SETRESULTTYPE,
ORDER_SETTERMINATOR,
ORDER_SCANNER_GETTYPESTATUS,
ORDER_SETCONFIG,
ORDER_GETCONFIG,
ORDER_ENABLEPREFIX,
ORDER_ENABLEPOSTFIX,
ORDER_SETPREFIX,
ORDER_SETPOSTFIX,
ORDER_GETPREFIX,
ORDER_GETPOSTFIX,
ORDER_SETSCANKEY,
ORDER_SETSFILENAME,
ORDER_SETFFILENAME,
ORDER_SETVIBRATETIME_SUCCESS,
ORDER_SETVIBRATETIME_FAIL,
ORDER_GETSETTING,
ORDER_BARTYPE_DISABLE,
ORDER_BARTYPE_ENABLE,
ORDER_SET_CURBARCODE,
ORDER_SET_BARID,
ORDER_SET_BARLENGTH_MAX,

ORDER_SET_BARLENGTH_MIN,
ORDER_SET_BARFLAG_INDEX,
ORDER_SET_BARFLAG_BFLAG,
ORDER_GETBARFLAG,
ORDER_DETAIL_SET,
ORDER_VIBRATEOPERATION,
ORDER_TRANSMIT_ID,
ORDER_GETSCANNERVERSION,
ORDER_IS_TURNON,
ORDER_SWITCH_POWER,
ORDER_OEM_INIT,

ORDER_SET_TRIGGERENABLE,
ORDER_SET_TRIGGERDISABLE,
ORDER_SET_ISTRIGGERENABLE,
//2D scanner(IT5300)
ORDER_ISSCANCLK,
ORDER_SETSCANCLK,
ORDER_GETCAPTURE,
ORDER_STOPSCAN,
ORDER_GET_DECODERREVISION,
ORDER_GET_APIREVISION,
ORDER_CENTERWINDOW_EANBLE,
ORDER_CENTERWINDOW_DISABLE,
ORDER_SET_DECODEMODE,
ORDER_SET_DELAYDECODING,
ORDER_IMAGESTREAM_INIT,
ORDER_IMAGESTREAM_START,
ORDER_IMAGESTREAM_READ,
ORDER_IMAGESTREAM_STOP,
ORDER_SETEXPOSURESETTINGS,
ORDER_AIMER_ON,
ORDER_AIMER_OFF,
ORDER_ILLUMINATION_ON,
ORDER_ILLUMINATION_OFF,
ORDER_SETSCANLIGHTSMODE,
ORDER_GETENGINECONFIG,

```

ORDER_GET_SCANDRVREVISION,
ORDER_SETSCANIMAGEMODE,
ORDER_GETDEFAULTSETTING,
ORDER_GETOCRFLAG,
ORDER_SETOCRFLAG,

ORDER_GETCONTINUOUSMODE,
ORDER_SETCONTINUOUSMODE,
//1D scanner(IS4813)
ORDER_SETEXTRAREAD,
ORDER_SETEXTRAREADTIME,
ORDER_SETSTITCHINGENABLE,
ORDER_SETALLOWNONPROPORTIONAL,
ORDER_SETSTITCHINGLEVEL,
ORDER_SETROUNDINGUNCERTAINTYLEVEL,
} Order_t;

//      order for image action : Select the capture start and stop.
enum
{
    IMAGE_GET_START = 0,
    IMAGE_GET_STOP,
};

//      capture function : for 2D scanner.
enum
{
    IMAGE_WIDTH = 752,
    IMAGE_HEIGHT = 480,
    IMAGE_SIZE = IMAGE_WIDTH * IMAGE_HEIGHT,
    NUM_OF_COLOR_INDEX = 256,
    COLOR_TABLE_SIZE = sizeof(RGBQUAD) * NUM_OF_COLOR_INDEX,
    BITMAPINFO_SIZE = sizeof(BITMAPINFOHEADER) + COLOR_TABLE_SIZE,
    BMP_HEADER_SIZE = sizeof(BITMAPFILEHEADER) + BITMAPINFO_SIZE,
    BMP_FILE_SIZE = BMP_HEADER_SIZE + IMAGE_SIZE,
    BMP_IMAGE_OFFSET = BMP_HEADER_SIZE,
};

```

```
//      Scan Light Mode : Select the light mode.
typedef enum SCANLIGHTSMODE SCANLIGHTSMODE;
enum SCANLIGHTSMODE
{
    SCANLIGHTSMODE_OFF= 0,           // Aimers and illumination off.
    SCANLIGHTSMODE_ILLUM_ONLY_ON,    // Illumination only on.
    SCANLIGHTSMODE_AIMER_ONLY_ON,    // Aimers only on.
    SCANLIGHTSMODE_ON,               // Both aimers and illumination on
};

//      Beep Sound Changed – Added in SDK 1.1
typedef enum
{
    BEEP_SCALE_NONE = 0,
    BEEP_SCALE_C4 = 1,
    BEEP_SCALE_D4 ,
    BEEP_SCALE_E4 ,
    BEEP_SCALE_F4 ,
    BEEP_SCALE_G4 ,
    BEEP_SCALE_A4 ,
    BEEP_SCALE_B4 ,
    BEEP_SCALE_C5 ,
    BEEP_SCALE_D5 ,
    BEEP_SCALE_E5 ,
    BEEP_SCALE_F5 ,
    BEEP_SCALE_G5 ,
    BEEP_SCALE_A5 ,
    BEEP_SCALE_B5 ,
    BEEP_SCALE_C6 ,
    BEEP_SCALE_D6 ,
    BEEP_SCALE_E6 ,
    BEEP_SCALE_F6 ,
    BEEP_SCALE_G6 ,
    BEEP_SCALE_A6 ,
    BEEP_SCALE_B6 ,
    BEEP_SCALE_C7 ,
}BEEP_BUZZER_SCALE;
```

Structures section

```
//      structure for scanner configure
```

```
typedef struct _BCD_CONFIG
```

```
{
```

BOOL	config_scannerenable;	//スキャナー有効状態
BOOL	config_bautoscan;	//自動スキャン状態
DWORD	config_dwtriggertimeout;	// トリガータイムアウト時間
DWORD	config_dwautoscantriggertime;	//自動スキャン トリガー間隔時間.
BYTE	config_btresulttype;	//結果タイプ値.
BYTE	config_btTerminator;	// ターミネーター値
BYTE	config_btscankey;	//使用しない.
BOOL	config_bprefix;	// プレフィックス使用状態.
BOOL	config_bpostfix;	// ポストフィックス使用状態
TCHAR	config_szprefix[MAX_PATH];	// プレフィックス文字列
TCHAR	config_szpostfix[MAX_PATH];	// ポストフィックス文字列.
BOOL	config_bbeep;	// ビープ音使用状態.
BOOL	config_bvibrator;	// バイブレーター使用状態.
BOOL	config_bLED;	// LED 使用状態.
TCHAR	config_szscansuccessfile[MAX_PATH];	// スキャン成功時のWAVファイル名.
TCHAR	config_szscanfailfile[MAX_PATH];	// スキャン失敗時のWAVファイル名
DWORD	config_dwvibsuccessstime;	// スキャン成功時バイブレーター駆動時間.
DWORD	config_dwvibfailtime;	// スキャン失敗時バイブレーター駆動時間.
BAR_TYPE	config_BTcurbartype;	//現在使用中バーコードタイプ
BYTE	config_btcurindex;	//現在使用中インデックス
BOOL	config_bBarID;	// バーコードID使用状態.
// 2D scanner(IT5300)		
BOOL	config_bCenterWindow;	// [center window]使用状態.
BYTE	config_btDecodeMode;	// 現在のデコードモード
DWORD	config_dwDelayDecoding;	// デコード前遅延時間
DWORD	config_dwMaxExposure;	// イメージキャプチャにおける最大露光値.
DWORD	config_dwMaxGain;	// イメージキャプチャにおける最大ゲイン値
DWORD	config_dwTargetWhite;	// イメージキャプチャにおけるターゲットホワイト値.
DWORD	config_dwTargetWhiteWindow;	//イメージキャプチャにおけるターゲットホワイトウィンドウ値.
int	config_nScanImageMode;	// スキャナーモード状態(スキャンモード,イメージモード)
BOOL	config_bImageAimer;	// 照準の使用状態
BOOL	config_bImageIllumination;	// 照明の使用状態

```

//      1D scanner(IS4813)

BOOL config_bStitchingEnable;          // スティッチングモード設定
BOOL config_bAllowNonProportional;     // ノンプロポーションナルモード設定
DWORD config_dwStitchingLevel;         // スティッチングレベル.
DWORD config_dwRoundingUncertaintyLevel; // 誤差レベル

//      予約領域

DWORD      config_Reserve01;           // 予約エリア01.
DWORD      config_Reserve02;           // 予約エリア02.
DWORD      config_Reserve03;           // 予約エリア03.
DWORD      config_Reserve04;           // 予約エリア04.

} BCD_CONFIG, *PBCD_CONFIG;

```

```
// Each barcode type's detail flag
```

```

typedef struct _BAR_FLAG
{
    TCHAR  wszFlagName[MAX_PATH]; // detail option subject
    BOOL   bFlag;                 // detail option subject flag
    BYTE   btDIndex;              // detail option index
    BYTE   btDetail;              // just flag for programming(ignore)

    // 1D scanner
    BYTE   btparam;               // just flag for programming(ignore)
    DWORD  dwDetailProperty;       // barcode property only use IS4813 scanner.

    // Reserve field
    DWORD  dwDetailReserve01;      // Reserve field 01.
    DWORD  dwDetailReserve02;      // Reserve field 02.

}BAR_FLAG, *PBAR_FLAG;

```

```
//      each barcode setting's configuration.
```

```

struct structSSCannerSetting
{
    TCHAR wszBarName[MAX_PATH]; // barcode name
    BYTE  btIndex;              // barcode index
    BYTE  btMark;               // barcode mark(ID)
    char  chparam;              // read only(each barcode's parameter)
    BOOL  bCodeEnable;          // barcode enable flag
    BOOL  bRange;               // barcode range flag
    int   btLengthMin;          // barcode length min

```

```

        int btLengthMax;                // barcode length max

        PBAR_FLAG pBarFlag;             // barcode detail setting structure.

        //          1D scanner

        int btRangeMin;                 // barcode range min
        int btRangeMax;                 // barcode range max
        DWORD dwSymID;                  // barcode SymID only used IS4813 scanner.
        DWORD dwProperty;               // barcode property only use IS4813 scanner.
        int nExtraRead;                 // barcode ExtraRead only used IS4813 scanner.
        int nExtraTime;                 // barcode ExtraTime only used IS4813 scanner.

        //          Reserve field

        DWORD dwSetReserve01;           // Reserve field 01.
        DWORD dwSetReserve02;           // Reserve field 02.
};

```

```
//          Scan result's structure.
```

```
typedef struct _SCAN_RESULT
```

```

{
    TCHAR          szScanValue[MAX_PATH]; // Scan Value.
    TCHAR          szScanType[MAX_PATH];  // Scan Type(String).
    TCHAR          tcCodeID;              // Scan Type ID.
    BAR_TYPE       barType;               // Scan Type(BAR_TYPE).
    TCHAR          tcSymLetter;           // Symbol Letter.(2D)
    TCHAR          tcSymModifier;         // Decoding time (2D)
    WORD           unScanLength;          // Scan Length.
    TCHAR          szScanMaxValue[MAX_PATH*15]; // barcode value.
    // Reserve field
    DWORD          dwDecodeTime;          // DecodeTime (ms)
    DWORD          dwResultReserve02;     // Reserve field 02.
} SCAN_RESULT, *PSCAN_RESULT;

```

```
// OCR's detail flag
```

```
typedef struct _OCR_FLAG
```

```

{
    TCHAR  szOCRTemplate[MAX_PATH];      // A null-terminated string that
    indicates one or more template patterns for the OCR decode.

    TCHAR  szOCRGroupG[MAX_PATH];        // A null-terminated string that defines
    the set of characters matching group "g" in a template.

```

```
    TCHAR  szOCRGroupH[MAX_PATH];           // A null-terminated string that defines
the set of characters matching group "h" in a template.

    TCHAR  szOCRCheckChar[MAX_PATH];        // A null-terminated string that defines
the legal characters for checksum computation in a decoded message.
}OCR_FLAG, *POCR_FLAG;

//      for bitmap format when you use to image.
typedef struct BmpFormat BmpFormat;
struct BmpFormat
{
    BITMAPFILEHEADER * pBitmapFileHeader;
    unsigned int uBitmapFileHeaderSize;
    BITMAPINFOHEADER * pBitmapInfoHeader;
    unsigned int uBitmapInfoHeaderSize;
    RGBQUAD * pRgbQuad;
    unsigned int uRgbQuadSize;
    unsigned char * pImageData;
    unsigned int uImageDataSize;
};
```


3.4.2 エンジン構造体と定義

```
typedef struct ENGINECONFIG ENGINECONFIG;
```

```
struct ENGINECONFIG
```

```
{
```

```
    DWORD dwIlluminationType;    // Writeable 0 = Green, Non Zero = Red
    DWORD dwLedControlMode;      // Read Only at this time (interlaced/concurrent mode)
    DWORD dwPixelFrequency;      // Read Only at this time(12Mh/24Mh)
    DWORD dwEngineId;            // Read Only(Engine ID Number)
                                // 0~2 BIT : Focus position of the lens(SF : 001, SR : 010)
                                // 3~4 BIT : Reserved.
                                // 5~6 BIT : Type of illumination populated(future use).
                                // 7~8 BIT : Reserved.
                                // 9~11 BIT : Aimer type(5000 LED : 000, 5100
                                // Bright Aimer : 001, 5300 Laser : 010)
                                // 12~14 BIT : Imager type(future use).
                                // 15 BIT : Reserved.

    DWORD dwFirmwareCksum;       // Read Only : FirmwareCksum
    DWORD dwFirmwareVersion;     // Read Only : FirmwareVersion
    DWORD dwAimerCenterX;        // Read Only: AimerCenterX
    DWORD dwAimerCenterY;        // Read Only : AimerCenterY
    DWORD dwEngineSensorID;      // Read Only : EngineSensorID
    DWORD dwEngineID;            //Type of Imager
                                // TYPE_NONE = 0, No imager hardware
                                // TYPE_IT4200 = 1,
                                // TYPE_IT4000 = 5,
                                // TYPE_IT4100 = 6,
                                // TYPE_IT4300 = 7,
                                // TYPE_IT5000VGA = 10,
                                // TYPE_IT5000VGA_P = 11

    DWORD dwImagerRows;          // Number of rows for a given imager.
    DWORD dwImagerCols;          // Number of rows for a given imager.
    DWORD dwBitsPerPixel;        // Number of rows for a given imager.
    DWORD dwRotation;            // RIGHT_SIDE_UP = 0, ROTATED_RIGHT,
                                // UPSIDE_DOWN, ROTATED_LEFT.
```

```
DWORD dwAimerXoffset;    // This value represents the X coordinate
                          // for the center of the aimer pattern.
DWORD dwAimerYoffset;    // This value represents the Y coordinate for the center
                          // of the aimer pattern.
DWORD dwYDepth;          // This value represents the Y Depth
};
```

```
typedef struct _BCD_BEEP_LEVEL_SETTING
```

```
{
    DWORD dwSize;          // Struct Size
    BYTE  nBeepType;        // Success Beep = 1 , Fail Beep = 2
    BYTE  nFirstBeep;       // First Beep Level
    BYTE  nSecondBeep;      // Second Beep Level
    BYTE  nValidBeep;       // Not Used
    WORD  dwBeepTime;       // First Beep and Second Beep Time value
    WORD  dwTimeReserve;    // TimeReserved
    DWORD dwReserved;       // Reserved
}BCD_BEEP_LEVEL_SETTING, *PBCD_BEEP_LEVEL_SETTING;
```

3.5 メッセージ

下記の記述はSDKで使用するメッセージです。

```
//      define for USER Message

#define          WM_SCANNED                      WM_USER+0x7777
#define          WM_IMAGE                        WM_USER+0x7778
```

(1) WM_SCANNED

スキャン結果タイプが[RESULT_USERMSG]の場合、アプリケーションはスキャン後に[WM_SCANNED]を取得します。

(2) WM_IMAGE

スキャナーモードがイメージモードの場合、アプリケーションはトリガーON/OFF後に[WM_IMAGE]を取得します。
[WM_IMAGE]メッセージの[LPARAM]はトリガーONの時、[IMAGE_GET_START]です。
[WM_IMAGE]メッセージの[LPARAM]はトリガーOFFの時、[IMAGE_GET_STOP]です。

下記記述は[WM_SCANNED]と[WM_IMAGE]メッセージを使用したサンプルコードです。

```
BOOL CCaptureDiagDlg::PreTranslateMessage(MSG* pMsg)
{
//      about scanner
if(pMsg->message == WM_SCANNED)
{
        if(pMsg->lParam == ERROR_NO_ERROR)
        {
                //      Success get scan result.
        }
        else
        {
                //      Fail get scan result.
        }
}

//      about image
if(pMsg->message == WM_IMAGE)
{
        if(pMsg->lParam == IMAGE_GET_START)
        {
                //      Image Capture Start
        }
        else//      pMsg->lParam == IMAGE_GET_STOP
        {
                //      Image Capture Stop
        }
}
return CDialog::PreTranslateMessage(pMsg);
}
```

4 BCD.NET.dll

4.1 BCD.NET.dllとは

[BCD.NET.dll]は、.NETユーザーに対して、DllImportを通して[BcdCore.dll]のいくつかの関数を使用できるようにしたものです。この関数は、まさにインポートしただけであり、[BcdCore.dll]の関数と同じです。
[BCD.NET.dll]はスキャナーを使う上で最小の関数のみ対応しています。

4.2 BCD.NET.dll の構成

1. スキャンデータを[OnScanned]関数に転送します。[OnScanned]関数はスキャナーイベントハンドラーです。
2. コンストラクターの中でスキャナーを制御するために必要な関数を呼び出します。
コンストラクター中に :
 BcdOpen();
 BcdEnable(true);
 BcdSetResultType(0);
 BcdSetHWND(scanMsgWnd.Hwnd);
 終了処理
 BcdClose();
3. スキャナーを制御するための主な関数の再定義(DllImportを使って)
 BcdEnable(bool bEnable) : public void Enable(bool bEnable)
 BcdSetHWND(IntPtr hWnd) : public void SetHWND(IntPtr hWnd)
 BcdEnableBeep(bool bEnable) : public void EnableBeep(bool bEnable)
 BcdEnableVibrator(bool enable) : public void EnableVibrator(bool enable)
 BcdEnableAutoScan(bool bEnable) : public void EnableAutoScan(bool bEnable)
 BcdEnableBarcode(int nBarType, bool bEnable) : public void EnableBarcode(int nBarType, bool bEnable)
 BcdTriggerTurnOn(bool bTurnOn) : public void TriggerTurnOn(bool bTurnOn)
 BcdSetScanTimeOut(uint uTimeOut) : public void SetScanTimeOut(uint uTimeOut)
 BcdSetAutoScanInterval(uint dwInterval) : public void SetAutoScanInterval(uint dwInterval)
 BcdGetBarTypeString(uint nBarType) : public unsafe string GetBarTypeString(uint nBarType)
 Bcdstring GetModelName() : public unsafe string GetModelName()
 Bcdstring GetModelSerialNumber() : public unsafe string GetModelSerialNumber()

4.3 BCD.NET.dll 使用サンプル

下記のコードは付属サンプルプログラム(tScanner)から抽出したものです。

```
...
using BCD.net;
...
namespace tScanner
{
    public partial class Form1 : Form
    {
        public CBScanner BScanner;
        public Form1()
        {
            InitializeComponent();
            // for scanner
            this.BScanner = new BCD.net.CBScanner();
            BScanner.Enable(true);
            this.BScanner.OnScanned += new
            BCD.net.ScannerEventHandler(this.FormScanTest_OnScanned);
            this.BScanner.EnableAutoScan(false);
            this.BScanner.EnableBeep(true);
        }

        public void FormScanTest_OnScanned(string strScanData, uint nLen, int barType, long errCode)
        {
            if (errCode == 0)
            {
                textBox1.Text = strScanData;
                textBox2.Text = this.BScanner.GetBarTypeString((uint)barType);
            }
            else
            {
                textBox1.Text = "SCAN FAIL";
                textBox2.Text = "SCAN FAIL";
            }
        }

        private void checkBox1_CheckStateChanged(object sender, EventArgs e)
        {
            this.BScanner.EnableAutoScan(checkBox1.Checked);
        }

        private void checkBox2_CheckStateChanged(object sender, EventArgs e)
        {
            this.BScanner.EnableBeep(checkBox2.Checked);
        }

        private void checkBox3_CheckStateChanged(object sender, EventArgs e)
        {
            this.BScanner.EnableVibrator(checkBox3.Checked);
        }

        private void button1_Click(object sender, EventArgs e)
    }
}
```

```
{
    this.BScanner.TriggerTurnOn(true);
}

private void button2_Click(object sender, EventArgs e)
{
    this.BScanner.EnableAutoScan(false);
    checkBox1.Checked = false;
    this.BScanner.TriggerTurnOn(false);
}
}
```

[BCD.NET.dll]はすべてのスキャナ関数に対応していません。

[BCD.NET.dll]はスキャナを使う上で最小の関数に対応しています。

もしその他の関数を使用したい場合は、[DllImport]を通して[BcdApi.h]の中のすべての関数を使用できます。