

PM250TA(1次元モデル) スキャナー SDK

コマンドリファレンス・マニュアル

Ver 1.3



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	7
2.1 概要	7
2.1.1 ユーザーレイヤー	7
2.1.2 SDK	7
2.1.3 ハードウェアインタフェースレイヤー	7
3 定義、API関数、メッセージ	8
3.1 APIの構成	8
3.1.1 スキャナー関数	8
BcdOpen	8
BcdClose	8
BcdEnable	8
BcdIsEnabled	8
BcdSetHWND	9
BcdSetTerminator	9
BcdSetResultType	9
BcdGetResultType	9
BcdEnableBeep	10
BcdSetSuccessSoundFileName	10
BcdSetFailSoundFileName	10
BcdSetSuccessSound	10
BcdSetFailSound	10
BcdEnableVibrator	11
BcdSetVibratorInterval	11
BcdEnableLED	11
BcdEnablePrefix	11
BcdSetPrefix	11
BcdGetPrefix	12
BcdEnablePostfix	12
BcdSetPostfix	12
BcdGetPostfix	12

BcdSetTrigTimeout	1 2
BcdEanbleAutoScan.....	1 3
BcdSetAutoScanInterval	1 3
BcdSetConfig	1 3
BcdGetConfig	1 3
BcdGetBarTypeString	1 3
BcdGetEnableLength	1 4
BcdGetBarLength	1 4
BcdSetBarLength	1 4
BcdTransmitBarID	1 4
BcdGetBarID	1 5
BcdSetBarID	1 5
3.1.2 インジコン関数	1 6
BcdSetDefault	1 6
BcdEnableBarcode.....	1 6
BcdEnableALLBarcode.....	1 6
BcdIsBarcodeEnabled	1 6
BcdSetBarFlag.....	1 7
BcdGetBarFlagArrayPtr	1 7
BcdTrigTurnOn	1 7
BcdGetResult	1 7
BcdGetString.....	1 7
BcdGetBarType	1 8
BcdGetLength	1 8
BcdGetVersion	1 8
BcdSetLinearSecurityLevel	1 8
BcdSetBidirectionalMode	1 9
BcdTriggerEnable.....	1 9
BcdIsTriggerEnabled	1 9
3.1.3 その他の関数	2 0
BcdGetModelName	2 0
BcdGetModelSerialNumber.....	2 0
3.2 関数と関数ポインタ	2 1
3.3 スキャナ-APIの使い方	2 3
3.4 構造体と定義の構成	2 4
3.4.1 スキャナ-構造体と定義	2 4
3.5 メッセージ	3 0
(1) WM_SCANNED	3 0

4 BCD.NET.DLL	3 1
4.1 BCD.NET.dllとは	3 1
4.2 BCD.NET.dll の構成	3 1
4.3 BCD.NET.dll 使用サンプル	3 2

1 ドキュメントについて

このドキュメントは、PM250TA(PM250 1次元モデル)のレーザー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。

OSイメージ版数と1DスキャナーSDKとの対応を下表に示します。

このドキュメントの内容は、下表の最新版数に対応しています。

(版数・バージョンの確認方法は、PM250ユーザーマニュアルを参照して下さい。)

OSイメージ 版数	対応1Dスキャナ-SDK
25.04(Sep 01 2009)	PMx50_SDK_SE955_rev2.3(090806).zip
25.05(Nov 12 2009)	PMx50_SDK_SE955_rev2.8(091103).zip
25.06(Dec 15 2009)	PMx50_SDK_SE955_rev3.0(091208).zip
25.07(Dec 23 2009)	
25.08(Aug 30 2010)	PMx50_SDK_SE955_rev3.1(100111).zip

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)
- ・PM250TAH(2次元モデル)スキャナー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	2009年10月1日	1.日本語版 初版発行	浜松東亜電機株式会社
1.1	2009年12月1日	1. OS(V25.05)に対応 <ul style="list-style-type: none"> ・「BcdEnableVibrate」削除 ・「BcdSetVibrateTime」削除 ・「BcdVibrateOperation」削除 ・「BcdSetScanKey」削除 ・「BcdGetResultType」ハグ Fix ・「BcdSetPrefix」ハグ Fix ・「BcdGetPrefix」ハグ Fix ・「BcdSetPostfix」ハグ Fix ・「BcdGetPostfix」ハグ Fix ・「BcdSetCurBarType」削除 ・「BcdGetParam」削除 ・「BcdGetSettingSturcture」削除 ・「g_getbarcodeypestring」削除 ・「g_getbarcodevaluestring」削除 ・「BcdDEBUG」削除 	浜松東亜電機株式会社

		<ul style="list-style-type: none"> ・「BcdGetBarFlagArrayPtr」ハグ Fix ・「BcdGetModelName」ハグ Fix ・「BcdInit」削除 ・「BcdOEMInit」削除 ・Code39デフォルトハグコード長変更 ・DISCRETE_2_OF_5デフォルトハグコード長変更 ・「BcdGetModelSerialNumber」ハグ Fix ・「BcdTriggerEnable」追加 ・「BcdIsTriggerEnabled」追加 <p>2. 「BAR_FLAG」記述追加</p>	
1.2	2009年12月25日	<p>1. OS(V25.07)に対応</p> <ul style="list-style-type: none"> ・「BcdEnableALLBarcode」追加 ・「BcdSetBarLength」復帰値変更 <p>2. 「BCD.NET.dll」修正</p>	浜松東亜電機株式会社
1.3	2010年8月31日	<p>1. OS(V25.08)に対応</p> <ul style="list-style-type: none"> ・「BAR_FLAG」構造体の変数名変更 (「btDIntex」 「btDIndex」) <p>2. OSとの対応追記</p>	浜松東亜電機株式会社

2 序論

2.1 概要

このSDKは、PM250(1次元)のスキャナ制御を行うものです。

このSDKは、スキャナ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PM250の1次元スキャナを制御できます。

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、PM250 1次元スキャナ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PM250 1次元スキャナ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_CR: [CR]コード

TERMINATOR_LF: [LF]コード

TERMINATOR_SPACE: [SPACE]コード

TERMINATOR_TAB: [TAB]コード

TERMINATOR_STXETX: 先頭に[STX]コード,最後に[ETX]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。

BcdSetSuccessSoundFileName

.....

スキャン成功音のファイルを設定します。(Wavファイルは[Windows]ディレクトリに存在しなければならない。)

Void BcdSetSuccessSoundFileName(TCHAR *szSFileName)

復帰値

Void

パラメータ

szSFileName: スキャン成功時WAVファイル名

BcdSetFailSoundFileName

.....

スキャン失敗音のファイルを設定します。(Wavファイルは[Windows]ディレクトリに存在しなければならない。)

Void BcdSetFailSoundFileName(TCHAR * szSFileName)

復帰値

Void

パラメータ

szSFileName: スキャン失敗時WAVファイル名

BcdSetSuccessSound

.....

スキャン成功音のファイルを設定します。(Wavファイルは[Windows]ディレクトリに存在しなければならない。)

Void BcdSetSuccessSound(LPWSTR strSFileName)

復帰値

Void

パラメータ

strSFileName: スキャン成功WAVファイル名

BcdSetFailSound

.....

スキャン失敗音のファイルを設定します。(Wavファイルは[Windows]ディレクトリに存在しなければならない。)

Void BcdSetFailSound(LPWSTR strFFFileName)

Return Values

Void

Parameters

strFFFileName: スキャン失敗時WAVファイル名

BcdEnableVibrator

.....

スキャン後のバイブレータ使用を決定します。

Void BcdEnableVibrator(BOOL bVibrate)

復帰値

Void

パラメータ

bVibrate: [TRUE]の場合、バイブレータが使用されます。

BcdSetVibratorInterval

.....

スキャン成功後とスキャン失敗後のバイブレータ作動時間を設定します。

Void BcdSetVibratorInterval(DWORD dwSuccessInterval, DWORD dwFailInterval)

復帰値

Void

パラメータ

dwSuccessInterval: スキャン成功後のバイブレータ作動時間 (単位: ミリ秒)

dwFailInterval: スキャン失敗後のバイブレータ作動時間 (単位: ミリ秒)

(注意)

パラメータの最大値は、「3000」(3秒)です。それ以上の値を設定しても最大値で設定されます。

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。([BcdEnablePrefix]関数で、使用を有効化してから本関数にて設定します。)

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

スキャン後のホ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、スキャン後にホストフィックスを使用します。

BcdSetPostfix

ホストフィックスを使用する場合、ホストフィックス文字列を設定します。([BcdEnablePostfix]関数で、使用を有効化してから本関数にて設定します。)

Void BcdSetPostfix(TCHAR *szPostfix)

復帰値

Void

パラメータ

szPostfix: ホストフィックス文字列

BcdGetPostfix

ホストフィックスを使用する場合、ホ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: ホ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

[SCD_CONFIG]の値を設定します。

Void BcdSetConfig(PSCD_CONFIG pbcdConfig)

復帰値

Void

パラメータ

pbcdConfig: [SCD_CONFIG]へのポインタ

BcdGetConfig

[SCD_CONFIG]を取得します。

Void BcdGetConfig(PSCD_CONFIG pbcdConfig)

復帰値

Void

パラメータ

pbcdConfig: [SCD_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 (=1 ~ 22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 (=1 ~ 22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 (=1 ~ 22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 (=1 ~ 22が有効です。)

uMin: 最小長設定値

uMax: 最大長設定値

(注意)

uMin値およびuMax値は、「55」以下の値が有効です。

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 (=1 ~ 22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 (=1 ~ 22が有効です。)

barID: バーコードID設定値

3.1.2 インジク関数

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 (=1 ~ 22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 BcdEnableALLBarcode(BOOL bEnable)

復帰値

BOOL: [TRUE]の場合正常終了。(1 ~ 2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 (=1 ~ 22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 (=1 ~ 22が有効です。)

nIndex: 詳細設定のインデックス(=「0」以上)

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 (=1 ~ 22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

BcdSetLinearSecurityLevel

スキャナーのリニアセキュリティレベルを設定します。

Void BcdSetLinearSecurityLevel (BYTE btLevel)

復帰値

Void

パラメータ

btLevel: スキャナーセキュリティレベル (=「1」～「4」が有効。(注意)参照。)

(注意)

リニアコードタイプ(例えばCode39、Interleaved 2 of 5)に対して4レベルのデコードセキュリティを設定できます。

バーコード品質の低下に対してより高いセキュリティレベルを選択して下さい。

セキュリティレベルを上げるに従ってスキャンエンジンの活動性は減少します。

バーコード品質に応じて適当なセキュリティレベルを選択して下さい。

Linear Security Level 1:

バーコードタイプ(Codabar, MSI, D 2of5, I 2of5)についてはデコードする前に2回読み込み成功しなければならない。

Linear Security Level 2:

すべてのバーコードタイプにてデコードする前に2回読み込み成功しなければならない。

Linear Security Level 3:

バーコードタイプ((MSI, D 2of5, I 2of5)についてはデコードする前に3回読み込み成功しなければならない。

それ以外のタイプについては2回成功しなければならない。

Linear Security Level 4:

すべてのバーコードタイプにてデコードする前に3回読み込み成功しなければならない。

BcdSetBidirectionalMode

.....

スキャナーの双方向モードを設定します。

Void BcdSetBidirectionalMode(BOOL bBidirectional)

復帰値

Void

パラメータ

bBidirectional: [TRUE]の場合、双方向モードが有効です。

(注意)

このパラメータはリニアコードタイプセキュリティレベルが有効の場合にのみ有効です。

このパラメータが有効な場合、バーコードはデコードされる前に両方の方向(前方向と逆方向)でスキャンが成功しなければならない。

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

3.1.3 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

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);
BcdSetSuccessSoundFileName	typedef void (WINAPI *PFN_BcdSetSuccessSoundFileName)(TCHAR*);
BcdSetFailSoundFileName	typedef void (WINAPI *PFN_BcdSetFailSoundFileName)(TCHAR*);
BcdSetSuccessSound	typedef void (WINAPI *PFN_BcdSetSuccessSound)(LPWSTR);
BcdSetFailSound	typedef void (WINAPI *PFN_BcdSetFailSound)(LPWSTR);
BcdEnableVibrator	typedef void (WINAPI *PFN_BcdEnableVibrator)(BOOL);
BcdSetVibratorInterval	typedef void (WINAPI *PFN_BcdSetVibratorInterval)(DWORD, DWORD);
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)(PSCD_CONFIG);
BcdGetConfig	typedef void (WINAPI *PFN_BcdGetConfig)(PSCD_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*);
BcdSetLinearSecurityLevel	typedef void (WINAPI *PFN_BcdSetLinearSecurityLevel)(BYTE);

BcdSetBidirectionalMode	typedef void (WINAPI *PFN_BcdSetBidirectionalMode)(BOOL);
BcdTriggerEnable	typedef void(WINAPI *PFN_BcdTriggerEnable)(BOOL);
BcdIsTriggerEnabled	typedef BOOL(WINAPI *PFN_BcdIsTriggerEnabled)(void);
BcdGetModelName	typedef LPWSTR (WINAPI *PFN_BcdGetModelName)(void);
BcdGetModelSerialNumber	typedef LPWSTR (WINAPI *PFN_BcdGetModelSerialNumber)(void);

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 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_CR 0x02 // CR
#define TERMINATOR_LF 0x03 // LF
#define TERMINATOR_SPACE 0x04 // SPACE
#define TERMINATOR_TAB 0x05 // TAB
#define TERMINATOR_STXETX 0x06 // STX + ETX

//      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.

//      scan status : WM_SCANNED message s LPARAM value.
typedef enum _STATUS_SCD
{
    ERROR_NO_ERROR = 0, // Scan success.
    ERROR_NO_READ, // Scan fail.
} ERROR_BSCANNER;

//      scanner order : CODE_ID of scan driver IOCTL.
typedef enum
```



```
{  
    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_ISENBLED,  
  
    ORDER_TRIGGER_TIMEOUT,  
    ORDER_SCANNER_DEFAULT,  
    ORDER_SCANNER_GETPARAM,  
    ORDER_SCANNER_AUTOSCANINTERVAL,  
    ORDER_SETRESULTTYPE,  
    ORDER_SETTERMINATOR,  
    ORDER_SCANNER_GETTYPESTATUS,  
    SCD_SETCONFIG,  
    SCD_GETCONFIG,  
    SCD_ENABLEPREFIX,  
    SCD_ENABLEPOSTFIX,  
    SCD_SETPREFIX,  
    SCD_SETPOSTFIX,  
}
```

```
SCD_GETPREFIX,
SCD_GETPOSTFIX,
SCD_SETSCANKEY,
SCD_SETSFILENAME,
SCD_SETFFILENAME,
SCD_SETVIBRATETIME_SUCCESS,
SCD_SETVIBRATETIME_FAIL,
SCD_GETSETTING,
SCD_BARTYPE_DISABLE,
SCD_BARTYPE_ENABLE,
SCD_SET_CURBARCODE,
SCD_SET_BARID,
SCD_SET_BARLENGTH_MAX,
SCD_SET_BARLENGTH_MIN,
SCD_SET_BARFLAG_INDEX,
SCD_SET_BARFLAG_BFLAG,
SCD_GETBARFLAG,
SCD_DETAIL_SET,
SCD_VIBRATEOPERATION,
SCD_TRANSMIT_ID,
ORDER_GETSCANNERVERSION,
ORDER_GETFIRMWAREVERSION,
ORDER_IS_TURNON,
ORDER_SWITCH_POWER,
ORDER_OEM_INIT,
ORDER_POWER_MAX,
ORDER_POWER_LOW,
ORDER_SET_LINEARSECURITYLEVEL,
ORDER_SET_BIDIRECTIONALMODE,
ORDER_SET_TRIGGERENABLE,
ORDER_SET_TRIGGERDISABLE,
ORDER_SET_ISTRIGGERENABLE,
} Order_t;

//      barcode type
typedef enum _BAR_TYPE
```

```

{
    BAR_TYPE_ERROR            =0x00,
    BAR_UNKNOWN               =0x00,
    UPC_A                     =0x01,
    UPC_E                     =0x02,
    UPC_E1                    =0x03,
    EAN_8                     =0x04,
    EAN_13                    =0x05,
    BOOKLAND_EAN              =0x06,
    USS_128                   =0x07,
    EAN_128                   =0x08,
    ISBT_128                  =0x09,
    CODE_39                   =0x0A,
    TRIOPTIC_CODE_39          =0x0B,
    CODE_93                   =0x0C,
    CODE_11                   =0x0D,
    INTERLEAVED_2_OF_5        =0x0E,
    DISCRETE_2_OF_5           =0x0F,
    CODABAR                   =0x10,
    MSI                       =0x11,
    GS1_DATABAR_14            =0x12,
    GS1_DATABAR_LIMITED       =0x13,
    GS1_DATABAR_EXPANDED      =0x14,
    UCC_COUPON                =0x15,,
    CHINESE_2_OF_5            =0x16,
    BAR_ALL                   =0x17,
}BAR_TYPE;

```

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

```
typedef struct _SCAN_RESULT
```

```

{
    TCHAR          szScanValue[MAX_PATH];
    TCHAR          szScanType[MAX_PATH];
    BAR_TYPE       barType;
    UINT           unScanLength;
} SCAN_RESULT, *PSCAN_RESULT;

```

```
//      Each barcode type s detail flag
typedef struct _BAR_FLAG
{
    THCAR    wszflagName[MAX_PATH];           //      detail option subject
    BOOL     bFlag;                           //      detail option subject flag
    BYTE     btDIndex;                        //      detail option index
    BYTE     btDetail;                        //      just flag for programming(ignore)
}BAR_FLAG,*PBAR_FLAG;
```

```
//      structure for scanner configure
typedef struct _SCD_CONFIG
{
    BOOL     config_scannerenable;           // スキャナ-有効状態
    BOOL     config_bautoscan;              // 自動スキャン状態
    DWORD    config_dwtriggertimeout;        // トリガ-タイムアウト時間
    DWORD    config_dwautoscantriggertime;    // 自動スキャン トリガ-間隔時間
    BYTE     config_btresulttype;           // 結果タイプ値
    BYTE     config_btTerminator;           // タ-ミネ-タ-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;               // ホ-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];     // ホ-ストフィックス文字列
    BOOL     config_bbeep;                  // ビ-プ音使用状態
    BOOL     config_bvibrator;              // バイブレータ-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_dwvibsuccesstime;        // スキャン成功時バイブレータ-駆動時間
    DWORD    config_dwvibfailtime;          // スキャン失敗時バイブレータ-駆動時間
    BAR_TYPE config_BTcurbartype;            // 現在使用中バ-コードタイプ
    BYTE     config_btcurindex;             // 現在使用中インデックス
    BOOL     config_bBarID;                 // バ-コードID使用状態
    BYTE     config_btLinearSecurityLevel;   // リニア-セキュリティレベル値
```

```
        BOOL          config_bBidirectional;    // 双方向モード有効状態
} SCD_CONFIG, *PSCD_CONFIG;
```

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
```

(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。

下記記述は[WM_SCANNED]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->IParam == ERROR_NO_ERROR)
    {
        //      Success get scan result.
    }
    else
    {
        //      Fail get scan result.
    }
}
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);
3. スキャナ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)
 BcdSetSuccessSound(string strSFileName) : public void SetSuccessSound(string strSFileName)
 BcdSetFailSound(string strFFFileName) : public void SetFailSound(string strFFFileName)
 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);
            this.BScanner.EnableVibrator(false);
        }

        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。