

Programmer's guide

Uni SDK Version 1.2.0 Manual

Abstract

This document describes the methods and properties of the M3 MOBILE UNI SDK version 1.2.0. UNI SDK supports all M3 MOBILE products unless written separately.

Modules covered:

1D Scanner, 2D Imager, Long-range Scanner, Camera, RFID (LF/HF), UHF Gun, WLAN, Bluetooth, System SDK, POS peripheral (Printer, MSR, IC Card, RFID)

Copyright and Agreement

WARNING: All contents of this SDK manual are protected by the copyright laws and all rights are reserved. Unauthorized distribution or copying is strictly prohibited.

M3 Mobile does not guarantee the quality and performance of the programs written in unsupported programming language. For supported development tools and languages, please refer to Development Tool and Requirements section.

Development Tools and Requirements

Supported Development Tools

- Visual Studio 2005/2008 – Visual C++, Visual C#, Visual Basic .NET

Development System Requirements

- Pentium – 1 Gigahertz (GHz) processor or higher
- Microsoft Windows 98 / ME / 2000 / XP / 2003 / 7 / 10
- ActiveSync / Windows Mobile Device Center (WMDC)

Development Platform Requirements

- "M3 Plus Platform SDK" and "Windows Mobile 5.0 SDK for Pocket PC" must be installed on your computer to be able to develop software using this SDK.
- M3Plus Platform SDK : [DOWNLOAD](#)
- Windows Mobile 5.0 SDK for Pocket PC : [LINK](#)

Release History

UNI SDK Version 1.2.0 Date: December 2016

- Scanner (1D)
 - Added M3 UL10, M3 BK10 to supported model
 - Added SE965 Module of Zebra to supported model
 - Added N4313 Module of Honeywell to supported model
 - Minor bug fixes
- Imager (2D)
 - Added M3 UL10, M3 BK10 to supported model
 - Added N5600 Module of Honeywell to supported model
 - Added SE4500 Module of Zebra to supported model
 - Minor bug fixes
- LR Scanner (Long Range)
 - Added M3 UL10, M3 BK10 to supported model
 - Added M3 UL10 CE to supported model
 - Minor bug fixes
- Wlan
 - Added M3 UL10, M3 BK10 to supported model
 - Added 45N Module of Wireless lan to supported model
 - Minor bug fixes
- System SDK
 - Added M3 UL10, M3 BK10 to supported model
 - Minor bug fixes
- Added UHF Gun SDK
- Remove the GPS SDK and recommend using the MS GPS API(Application programming interface)
 - Please contact our online support web page to receive our GPS SDK.

UNI SDK Version 1.1.1 Date: November 2013

- Added M3 ORANGE PLUS CE
- System SDK
 - DLL names have been changed
 - System.DLL => M3system.DLL

UNI SDK Version 1.1.0 Date: June 2013

- Scanner (1D)
 - Added M3 ORANGE PLUS WM to supported model
 - Modified to include Symbol H/W decoder
 - Minor bug fixes
- Imager (2D)
 - Added M3 ORANGE PLUS WM to supported model
 - IQ setting save error fixed
- Camera (Windows CE only)
 - M3 SMART CE has been included
 - GetBrightness function added
- RFID (LF/HF)
 - Improved port open procedure
 - SendCommandData function added
- System SDK
 - Added M3 POS, MM3 to supported model
 - Gyro sensor control added for M3 SMART

UNI SDK Version 1.0.0 Date: January 2012

First release of combined SDK called UNI SDK

Contents

1	Introduction	15
2	Samples.....	16
3	Tutorial.....	17
3.1	SCANNER (1D)	19
3.2	IMAGER (2D)	22
3.3	LRSCANNER(Long-Range)	25
3.4	CAMERA (CE)	28
3.5	CAMERA (WM).....	32
3.6	RFID (LF/HF)	35
3.7	UHF Gun Reader	37
3.8	WLAN	41
3.9	BLUETOOTH	44
3.10	SYSTEM SDK	55
4	References	57
4.1	SCANNER (1D)	58
4.1.1	SCAN_Close	64
4.1.2	SCAN_GetAdaptiveScanning.....	64
4.1.3	SCAN_GetCODABAR	65
4.1.4	SCAN_GetCODE11.....	66
4.1.5	SCAN_GetCODE128.....	66
4.1.6	SCAN_GetCODE25.....	67
4.1.7	SCAN_GetCODE39.....	68
4.1.8	SCAN_GetCODE93.....	69
4.1.9	SCAN_GetDeviceType	70
4.1.10	SCAN_GetEAN13	70
4.1.11	SCAN_GetEAN8.....	71
4.1.12	SCAN_GetEngineType	72
4.1.13	SCAN_GetGS1	73
4.1.14	SCAN_GetKOREAPOST	73
4.1.15	SCAN_GetMSI	74
4.1.16	SCAN_GetOption	75
4.1.17	SCAN_GetScanData.....	76
4.1.18	SCAN_GetSymbology	76
4.1.19	SCAN_GetTELEPEN.....	78
4.1.20	SCAN_GetUPCA.....	78
4.1.21	SCAN_GetUPCE	79
4.1.22	SCAN_GetVersionInfo	80

4.1.23	SCAN_Open	81
4.1.24	SCAN_Read	81
4.1.25	SCAN_ReadCancel	82
4.1.26	SCAN_SetAdaptiveScanning	82
4.1.27	SCAN_SetCODABAR	83
4.1.28	SCAN_SetCODE11	84
4.1.29	SCAN_SetCODE128	85
4.1.30	SCAN_SetCODE25	85
4.1.31	SCAN_SetCODE39	86
4.1.32	SCAN_SetCODE93	87
4.1.33	SCAN_SetEAN13	88
4.1.34	SCAN_SetEAN8	89
4.1.35	SCAN_SetGS1	90
4.1.36	SCAN_SetHWnd	90
4.1.37	SCAN_SetKOREAPOST	91
4.1.38	SCAN_SetMSI	91
4.1.39	SCAN_SetOption	92
4.1.40	SCAN_SetSymbology	93
4.1.41	SCAN_SetSymbologyAll	94
4.1.42	SCAN_SetSymbologyDefault	95
4.1.43	SCAN_SetTELEPEN	95
4.1.44	SCAN_SetUPCA	96
4.1.45	SCAN_SetUPCE	97
4.2	IMAGER (2D)	99
4.2.1	IMAGER_CAMCapture	111
4.2.2	IMAGER_CAMGetOption	111
4.2.3	IMAGER_CAMInit	112
4.2.4	IMAGER_CAMPreviewStart	113
4.2.5	IMAGER_CAMPreviewStop	113
4.2.6	IMAGER_CAMSetOption	114
4.2.7	IMAGER_CAMUnInit	115
4.2.8	IMAGER_Close	115
4.2.9	IMAGER_GetAZTEC	116
4.2.10	IMAGER_GetCenteringWindow	117
4.2.11	IMAGER_GetCHINAPOST	118
4.2.12	IMAGER_GetCODABAR	118
4.2.13	IMAGER_GetCODABLOCK	119
4.2.14	IMAGER_GetCODE11	120
4.2.15	IMAGER_GetCODE128	121

4.2.16	IMAGER_GetCODE16K	122
4.2.17	IMAGER_GetCODE39	122
4.2.18	IMAGER_GetCODE49	123
4.2.19	IMAGER_GetCODE93	124
4.2.20	IMAGER_GetCOMPOSITE	125
4.2.21	IMAGER_GetDATAMATRIX	126
4.2.22	IMAGER_GetDecOption	126
4.2.23	IMAGER_GetDeviceType	127
4.2.24	IMAGER_GetEAN13	128
4.2.25	IMAGER_GetEAN8	129
4.2.26	IMAGER_GetIATA25	129
4.2.27	IMAGER_GetINT25	130
4.2.28	IMAGER_GetKOREAPOST	131
4.2.29	IMAGER_GetMATRIX25	132
4.2.30	IMAGER_GetMAXICODE	132
4.2.31	IMAGER_GetMICROPDF	133
4.2.32	IMAGER_GetMSI	134
4.2.33	IMAGER_GetOCR	135
4.2.34	IMAGER_GetOption	136
4.2.35	IMAGER_GetPDF417	137
4.2.36	IMAGER_GetPLANET	138
4.2.37	IMAGER_GetPLESSEY	138
4.2.38	IMAGER_GetPOSICODE	139
4.2.39	IMAGER_GetPOSTNET	140
4.2.40	IMAGER_GetQR	141
4.2.41	IMAGER_GetRSS	141
4.2.42	IMAGER_GetScanByteData	142
4.2.43	IMAGER_GetScanData	143
4.2.44	IMAGER_GetSTR25	144
4.2.45	IMAGER_GetSymbology	144
4.2.46	IMAGER_GetTELEPEN	147
4.2.47	IMAGER_GetUPCA	147
4.2.48	IMAGER_GetUPCE	148
4.2.49	IMAGER_GetVersionInfo	149
4.2.50	IMAGER_IQGetBarcodeData	150
4.2.51	IMAGER_IQGetOption	150
4.2.52	IMAGER_IQImagingStart	151
4.2.53	IMAGER_IQImagingStop	152
4.2.54	IMAGER_IQInit	152

4.2.55	IMAGER_IQSetOption	153
4.2.56	IMAGER_IQUnInit	154
4.2.57	IMAGER_Open	154
4.2.58	IMAGER_Read	155
4.2.59	IMAGER_ReadCancel	155
4.2.60	IMAGER_SetAZTEC	156
4.2.61	IMAGER_SetCenteringWindow	157
4.2.62	IMAGER_SetCHINAPOST	158
4.2.63	IMAGER_SetCODABAR	158
4.2.64	IMAGER_SetCODABLOCK	159
4.2.65	IMAGER_SetCODE11	160
4.2.66	IMAGER_SetCODE128	161
4.2.67	IMAGER_SetCODE16K	162
4.2.68	IMAGER_SetCODE39	162
4.2.69	IMAGER_SetCODE49	163
4.2.70	IMAGER_SetCODE93	164
4.2.71	IMAGER_SetCOMPOSITE	165
4.2.72	IMAGER_SetDATAMATRIX	166
4.2.73	IMAGER_SetDecOption	166
4.2.74	IMAGER_SetEAN13	167
4.2.75	IMAGER_SetEAN8	168
4.2.76	IMAGER_SetIATA25	169
4.2.77	IMAGER_SetINT25	170
4.2.78	IMAGER_SetKOREAPOST	170
4.2.79	IMAGER_SetMATRIX25	171
4.2.80	IMAGER_SetMAXICODE	172
4.2.81	IMAGER_SetMICROPDF	173
4.2.82	IMAGER_SetMSI	173
4.2.83	IMAGER_SetOCR	174
4.2.84	IMAGER_SetOption	175
4.2.85	IMAGER_SetPDF417	176
4.2.86	IMAGER_SetPLANET	177
4.2.87	IMAGER_SetPLESSEY	178
4.2.88	IMAGER_SetPOSICODE	178
4.2.89	IMAGER_SetPOSTNET	179
4.2.90	IMAGER_SetQR	180
4.2.91	IMAGER_SetRSS	181
4.2.92	IMAGER_SetSTRT25	181
4.2.93	IMAGER_SetSymbology	182

4.2.94	IMAGER_SetSymbologyAll	184
4.2.95	IMAGER_SetSymbologyDefault	185
4.2.96	IMAGER_SetTELEPEN	186
4.2.97	IMAGER_SetUPCA	186
4.2.98	IMAGER_SetUPCE	187
4.3	LRSCANNER (Long-Range)	189
4.3.1	LRSCAN_Close	195
4.3.2	LRSCAN_GetCODABAR	195
4.3.3	LRSCAN_GetCODABLOCK	196
4.3.4	LRSCAN_GetCODE11	197
4.3.5	LRSCAN_GetCODE128	197
4.3.6	LRSCAN_GetCODE39	198
4.3.7	LRSCAN_GetDeviceType	199
4.3.8	LRSCAN_GetEAN13	200
4.3.9	LRSCAN_GetEAN13	200
4.3.10	LRSCAN_GetEAN8	201
4.3.11	LRSCAN_GetGS1COMPOSITE	202
4.3.12	LRSCAN_GetGS1DATABAR	202
4.3.13	LRSCAN_GetINT25	203
4.3.14	LRSCAN_GetMSI	204
4.3.15	LRSCAN_GetOption	205
4.3.16	LRSCAN_GetPLESSEY	206
4.3.17	LRSCAN_GetPOSTNET	206
4.3.18	LRSCAN_GetScanData	207
4.3.19	LRSCAN_GetSTANDARD2OF5	208
4.3.20	LRSCAN_GetSymbology	209
4.3.21	LRSCAN_GetTELEPEN	210
4.3.22	LRSCAN_GetUPCA	211
4.3.23	LRSCAN_GetUPCE	212
4.3.24	LRSCAN_GetVersionInfo	213
4.3.25	LRSCAN_Open	213
4.3.26	LRSCAN_Read	214
4.3.27	LRSCAN_ReadCancel	215
4.3.28	LRSCAN_SetCODABAR	215
4.3.29	LRSCAN_SetCODABLOCK	216
4.3.30	LRSCAN_SetCODE11	217
4.3.31	LRSCAN_SetCODE128	217
4.3.32	LRSCAN_SetCODE39	218
4.3.33	LRSCAN_SetEAN13	219

4.3.34	LRSCAN_SetEAN8	220
4.3.35	LRSCAN_SetGS1COMPOSITE	221
4.3.36	LRSCAN_SetGS1DATABAR	221
4.3.37	LRSCAN_SetINT25	222
4.3.38	LRSCAN_SetMSI	223
4.3.39	LRSCAN_SetOption	224
4.3.40	LRSCAN_SetPLESSEY	225
4.3.41	LRSCAN_SetPOSTNET	225
4.3.42	LRSCAN_SetSTANDARD2OF5	226
4.3.43	LRSCAN_SetSymbology	227
4.3.44	LRSCAN_SetSymbologyAll	229
4.3.45	LRSCAN_SetSymbologyDefault	229
4.3.46	LRSCAN_SetTELEPEN	230
4.3.47	LRSCAN_SetUPCA	231
4.3.48	LRSCAN_SetUPCE	231
4.4	CAMERA (CE)	233
4.4.1	CAM_Open	237
4.4.2	CAM_Close	237
4.4.3	CAM_Capture	238
4.4.4	CAM_PreviewStart	239
4.4.5	CAM_PreviewStop	239
4.4.6	CAM_GetLastSaveFilePath	240
4.4.7	CAM_AutoFocus	240
4.4.8	CAM_EnableAutoAF	241
4.4.9	CAM_Zoom	242
4.4.10	CAM_SetCameraOption	242
4.4.11	CAM_GetCameraOption	243
4.4.12	CAM_Brightness	244
4.4.13	CAM_FlashOn	244
4.4.14	CAM_FlashOff	245
4.4.15	CAM_RawData	245
4.4.16	CAM_InsertExifInformation	246
4.4.17	CAM_UseGPSExifData	247
4.4.18	CAM_RegisterMsgWnd	247
4.4.19	CAM_GetScannerType	248
4.4.20	CAM_GetVersion	248
4.5	CAMERA (WM)	250
4.5.1	CAM_Open	254
4.5.2	CAM_Close	254

4.5.3	CAM_SetPreviewWindow	255
4.5.4	CAM_PreviewStart	256
4.5.5	CAM_PreviewStop	256
4.5.6	CAM_GetRegCapturePath	257
4.5.7	CAM_GetRegCapturePathEx.....	258
4.5.8	CAM_SetRegCapturePath.....	258
4.5.9	CAM_Capture	259
4.5.10	CAM_CaptureEx	260
4.5.11	CAM_EnableShutterSound	261
4.5.12	CAM_VideoStart	261
4.5.13	CAM_VideoStop	262
4.5.14	CAM_VideoStopEx	263
4.5.15	CAM_GetFlashState	263
4.5.16	CAM_AlwaysFlash	264
4.5.17	CAM_CaptureFlash	265
4.5.18	CAM_AutoFocus	265
4.5.19	CAM_SetAfMode	266
4.5.20	CAM_Zoom	266
4.5.21	CAM_SetResolution	267
4.5.22	CAM_GetResolution	268
4.5.23	CAM_SetQuality	268
4.5.24	CAM_GetQuality	269
4.5.25	CAM_SetBrightness.....	269
4.5.26	CAM_GetBrightness	270
4.5.27	CAM_SetWhiteBalance	270
4.5.28	CAM_GetWhiteBalance.....	271
4.5.29	CAM_SetContrast.....	272
4.5.30	CAM_GetContrast	272
4.5.31	CAM_SetSharpness	273
4.5.32	CAM_GetSharpness	273
4.5.33	CAM_SetHistoEqual	274
4.5.34	CAM_GetHistoEqual.....	274
4.5.35	CAM_RegisterPreview.....	275
4.5.36	CAM_RegisterMsgWnd	276
4.5.37	CAM_InsertExifInformation	276
4.5.38	CAM_GetRegExifEnable	277
4.5.39	CAM_UseGPSExifData	277
4.5.40	CAM_GetRegExifGpsEnable.....	278
4.5.41	CAM_GetRegInsertDateTimeEnable	279

4.5.42	CAM_SetInsertDateTimeEnable	279
4.5.43	CAM_GetModelType.....	280
4.5.44	CAM_GetScannerType	280
4.5.45	CAM_GetVersion	281
4.6	RFID (LF/HF).....	282
4.6.1	RFID_Open.....	285
4.6.2	RFID_Close.....	285
4.6.3	RFID_PowerSupply.....	286
4.6.4	RFID_GetType	287
4.6.5	RFID_SelectTag	288
4.6.6	RFID_HighSelectTag	288
4.6.7	RFID_LoginTag	289
4.6.8	RFID_ReadBlock	290
4.6.9	RFID_WriteBlock	291
4.6.10	RFID_ReadMultiBlock.....	292
4.6.11	RFID_WriteMultiBlock.....	292
4.6.12	RFID_SetAntenna	293
4.6.13	RFID_GetVersion.....	294
4.6.14	RFID_EnableMultiTag	295
4.6.15	RFID_SendReadMultiTag	295
4.6.16	RFID_SendTransferCommand.....	296
4.6.17	RFID_SendContinuousRead	297
4.6.18	RFID_SendCommand	298
4.6.19	RFID_SendCommandGetData.....	299
4.6.20	RFID_GetData	299
4.6.21	RFID_CheckResult	300
4.6.22	RFID_SoundPlay	301
4.6.23	RFID_SetTagType	302
4.6.24	RFID_GetTagType.....	303
4.6.25	RFID_GetTagTypeToString.....	303
4.6.26	RFID_TagItInventory	304
4.6.27	RFID_TagItReadBlock	304
4.6.28	RFID_TagItWriteBlock	305
4.7	UHF GUN READER	307
4.7.1	UHF_GetError	313
4.7.2	UHF_IsReady.....	314
4.7.3	UHF_Init	314
4.7.4	UHF_Close	315
4.7.5	UHF_Inventory.....	316

4.7.6	UHF_InventoryStop.....	317
4.7.7	UHF_Read.....	317
4.7.8	UHF_Write.....	318
4.7.9	UHF_Lock.....	320
4.7.10	UHF_Kill.....	321
4.7.11	UHF_GetData.....	322
4.7.12	UHF_SetRegionFrequency.....	323
4.7.13	UHF_GetRegionFrequency.....	324
4.7.14	UHF_ReadBattery.....	324
4.7.15	UHF_ReadBatteryStatus.....	325
4.7.16	UHF_Version.....	326
4.8	WLAN.....	328
4.8.1	WLAN_ActivateConfig.....	331
4.8.2	WLAN_Close.....	331
4.8.3	WLAN_ConnectAP.....	332
4.8.4	WLAN_ConnectAPEX.....	333
4.8.5	WLAN_DeleteConfig.....	334
4.8.6	WLAN_ExportConfig.....	335
4.8.7	WLAN_ImportConfig.....	335
4.8.8	WLAN_Init.....	336
4.8.9	WLAN_GetAllConfigName.....	336
4.8.10	WLAN_GetCurrentAPInfo.....	337
4.8.11	WLAN_GetBssidList.....	338
4.8.12	WLAN_GetPowerStatus.....	338
4.8.13	WLAN_PowerOn.....	339
4.8.14	WLAN_PowerOff.....	340
4.9	BLUETOOTH.....	341
4.9.1	BLUETOOTH_Close.....	351
4.9.2	BLUETOOTH_Connect.....	351
4.9.3	BLUETOOTH_CreateConnection.....	352
4.9.4	BLUETOOTH_CreateConnectionEx.....	352
4.9.5	BLUETOOTH_DeleteConnection.....	353
4.9.6	BLUETOOTH_Disconnect.....	354
4.9.7	BLUETOOTH_FindConnectionClose.....	354
4.9.8	BLUETOOTH_FindDeviceClose.....	355
4.9.9	BLUETOOTH_FindFirstConnection.....	356
4.9.10	BLUETOOTH_FindFirstConnectionEx.....	356
4.9.11	BLUETOOTH_FindFirstDevice.....	357
4.9.12	BLUETOOTH_FindFirstDeviceEx.....	358

4.9.13	BLUETOOTH_FindFirstService	359
4.9.14	BLUETOOTH_FindFirstServiceEx	360
4.9.15	BLUETOOTH_FindLocalDevice	361
4.9.16	BLUETOOTH_FindNextConnection.....	361
4.9.17	BLUETOOTH_FindNextConnectionEx	362
4.9.18	BLUETOOTH_FindNextDevice	363
4.9.19	BLUETOOTH_FindNextService	364
4.9.20	BLUETOOTH_FindNextServiceEx.....	364
4.9.21	BLUETOOTH_FindServiceClose	365
4.9.22	BLUETOOTH_GetBluetoothState	366
4.9.23	BLUETOOTH_GetSCOConnectionState.....	366
4.9.24	BLUETOOTH_GetSecurityMode	367
4.9.25	BLUETOOTH_Open.....	368
4.9.26	BLUETOOTH_PerformAction	368
4.9.27	BLUETOOTH_SetAuthenticationCallback	369
4.9.28	BLUETOOTH_SetBluetoothState	370
4.9.29	BLUETOOTH_SetConnectionCallback	371
4.9.30	BLUETOOTH_SetIncomingPIN.....	372
4.9.31	BLUETOOTH_SetOutgoingPIN.....	373
4.9.32	BLUETOOTH_SetSecurityMode.....	374
4.10	SYSTEM.....	376
4.10.1	SYSTEM_BacklightOn	379
4.10.2	SYSTEM_Cleanboot	379
4.10.3	SYSTEM_GetBacklightlevel	380
4.10.4	SYSTEM_GetBacklightTimeOut	381
4.10.5	SYSTEM_GetBatteryLifePercent	382
4.10.6	SYSTEM_GetBatteryState	383
4.10.7	SYSTEM_GetBluetooth	383
4.10.8	SYSTEM_GetCpuClock	384
4.10.9	SYSTEM_GetDeviceInfo	385
4.10.10	SYSTEM_GetGUID	386
4.10.11	SYSTEM_GetGSensorValue.....	387
4.10.12	SYSTEM_GetOSVersionInfo	388
4.10.13	SYSTEM_GetPhoneVolumeLevel.....	388
4.10.14	SYSTEM_GetPowerTimeOut	390
4.10.15	SYSTEM_GetSerialNumber	390
4.10.16	SYSTEM_GetVersionInfo	391
4.10.17	SYSTEM_GetVolumeLevel	392
4.10.18	SYSTEM_GetWlan	393

4.10.19	SYSTEM_KeypadLock.....	394
4.10.20	SYSTEM_Reboot.....	394
4.10.21	SYSTEM_SetBacklightlevel.....	395
4.10.22	SYSTEM_SetBacklightTimeOut.....	395
4.10.23	SYSTEM_SetBluetooth.....	396
4.10.24	SYSTEM_SetCpuClock.....	397
4.10.25	SYSTEM_SetGSensorValue	398
4.10.26	SYSTEM_SetPhoneVolumeLevel	398
4.10.27	SYSTEM_SetPowerTimeOut.....	399
4.10.28	SYSTEM_SetSleepMode	400
4.10.29	SYSTEM_SetVolumeLevel	400
4.10.30	SYSTEM_SetWlan	401
4.10.31	SYSTEM_Vibrate	401
4.10.32	SYSTEM_VolumeMute	402
5	Demo Manual.....	403
5.1	WLAN	403
5.2	BLUETOOTH	406
5.3	POS.....	407
5.4	SYSTEM.....	409

1 Introduction

This document is a reference guide for the software developer's kit (SDK) for M3 OX10 Devices. This handheld terminal may include up to 7 different modules depending on the specification of the device. For module list and brief description of each module, see below table.

Module Name	Description
SCANNER	Read 1D barcodes depend on the scanner module option.
IMAGER	Read 1D/2D barcodes depend on the scanner module option.
LRSCANNER	Read 1D/2D barcodes in long distance. Typically around 5 to 10 meters.
CAMERA	Used to take a picture of an object.
RFID/ UHF GUN	Read data from tags or write to the tags through Reader.
WLAN	Enable network connection using Wi-Fi. Summit WLAN module is used.
BLUETOOTH	Mainly used to connect to a Bluetooth head set for phone calls or printer.
SYSTEM	System status can be control or read.

This guide document provides comprehensive SDK manual by providing Tutorials, Samples and References including function lists.

2 Samples

This chapter is illustrate the demo programs included in the DLL, SDK and current version of the SDK as well as the development tool used to write the sample program.

Module	Demo Application	Sample Source	Version	Date
SCANNER	ScanTest.exe Scanner.dll	SCANNER_TEST	1.1.2 1.3.1.2	2016-11-30 2013-11-23
IMAGER	ImagerTest.exe Imager.dll	IMAGER_TEST	1.2.0 1.2.2.1	2014-01-15 2016-04-18
LRSCANNER	LRScanTest.exe LRScanner.dll	LRSCANNER_TEST	1.0.1 1.0.4	2014-11-07 2016-11-28
CAMERA_CE	CamTest.exe CAM.dll	CAMERA_TEST	1.5.2 1.5.3	2016-09-27 2016-09-27
CAMERA_WM	CamTest.exe CAM.dll	CAMERA_TEST	1.3.0 1.3.2	2013-12-18 2015-03-09
RFID	RfidTest.exe RFID.dll	RFID_TEST	1.1.0 1.2.0	2013-10-17 2016-08-04
UGR GUN READER	UHF_GunTest.exe RFID_UHF.dll	RFID_TEST	1.0.1	
WLAN	WlanTest.exe WLAN.dll	WLAN_TEST	1.0.2 1.2.2	2013-11-26 2016-09-28
BLUETOOTH	BluetoothTest.exe BLUETOOTH.dll	BLUETOOTH_TEST	1.0.1 1.0.0	
SYSTEM	SystemTest.exe M3System.dll	SYSTEM_TEST	1.1.8 1.3.1	2015-01-26 2016-09-27

3 Tutorial

This chapter describes the basic usage of M3 Mobile SDK functions in a step-by-step manner. In this tutorial section, the following topics are treated.

Section	Topic
SCANNER	Initialization ScanRead Get Window Message for getting Scan Data Close Scanner
IMAGER	Initialization ScanRead Get Window Message for getting Scan Data Close Scanner
LRSCANNER	Initialization ScanRead Get Window Message for getting Scan Data Close Scanner
CAMERA_CE	Open Close Capture Still Shot Preview Insert EXIF Information
CAMERA_WM	Open Close Capture Still Preview Insert EXIF Information
RFID	Open Antenna On/Off Tag Select Data Read Data Write
UHF GUN READER	Init UHF Start Inventory Get Data Stop Inventory Close UHF
WLAN	Initialization Searching AP Connect Config Delete Config Change Config Configuration Import/Export Close Wlan
BLUETOOTH	Initialization Device Find Service Find Connection Management Connect Disconnect Close
Printer	Init Printer Close Printer About Spool Add Command isReady Printer. Get Status

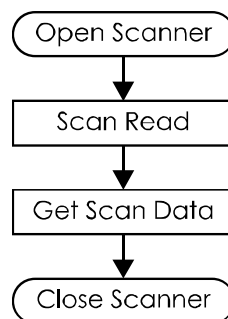
MSR	Init MSR Close MSR Start or Stop scanning Card.Read MSR Data
IC Card	Open ICCard Close ICCard ICCard/Sam Check
RFID	Open Rfid Close Rfid Start or Stop scanning Tag

3.1 SCANNER (1D)

Supported PDA:

Brand	Restriction
M3 BK10	No restriction
M3 MT10	No restriction
M3 OX10	No restriction
M3 PS10	No restriction
M3 ST10	No restriction
M3 UL10	No restriction

Basic scanner flow chart:



Tip !!

- I. Scanner communicates through serial so that other programs cannot use scanner while scanner is being run. For example, it occurs an error when other program access scanner while ScanEmul is running.
- II. Combined version of Imager.dll can be available for all 1D Scanner regardless of model type, Scanner Engine and OS.
- III. Its Virtual Keys are VK_F22 for WinCE and VK_F14 for Windows Mobile.
- IV. Scan data can be obtained through Windows Message and it can be checked in Scanner.h.

```
#define WM_SCAN_DATA WM_APP+350
```

Open Scanner

```
// Scanner Open
if(SCAN_Open() == FALSE)
{
    ::MessageBox(NULL, L"Error : SCAN_Open()", NULL, MB_TOPMOST);
}
m_DeviceType = SCAN_GetDeviceType();
// Set Scanner Key
if((m_DeviceType == DEVICE_M3SMART_CE) || (m_DeviceType == DEVICE_M3GREEN) || (m_DeviceType == DEVICE_M3T))
```

```

|| (m_DeviceType == DEVICE_M3POS))
{
    // Only for WinCE
    MoveWindow(0, 0, 240, 320);
    m_bWinCE = TRUE;
    RegisterHotKey(this->m_hWnd, HOTKEY_ONE, MOD_KEYUP, VK_F22);
    RegisterHotKey(this->m_hWnd, HOTKEY_TWO, MOD_KEYUP, 0);
    RegisterHotKey(this->m_hWnd, HOTKEY_THREE, MOD_KEYUP, 0);
}
else
{
    m_bWinCE = FALSE;
    RegisterHotKey(this->m_hWnd, HOTKEY_ONE, MOD_KEYUP, VK_F14);
    RegisterHotKey(this->m_hWnd, HOTKEY_TWO, MOD_KEYUP, 0);
    RegisterHotKey(this->m_hWnd, HOTKEY_THREE, MOD_KEYUP, 0);
}

```

Scan Read

```

void CScanTestDlg::OnBnClickedBtnScan()
{
    SCAN_Read();
}

```

Get ScanData

```

BEGIN_MESSAGE_MAP(CScanTestDlg, CDialog)
    //}}AFX_MSG_MAP
    ON_MESSAGE(WM_SCAN_DATA, OnScanRead)
END_MESSAGE_MAP()

:

LRESULT CScanTestDlg::OnScanRead(WPARAM wParam, LPARAM lParam)
{
    if(m_bNewForm == TRUE)
        return FALSE;

    TCHAR szBarType[1024] = {0, };
    TCHAR szBarData[1024] = {0, };
    SCAN_GetScanData(szBarType, szBarData);
    m_strStaticType = szBarType;
    m_strEditData = szBarData;
    UpdateData(FALSE);
    return LRESULT();
}

```

Close Scanner

```

void CScanTestDlg::OnDestroy()
{
    CDialog::OnDestroy();
}

```

```
UnregisterHotKey(this->m_hWnd, HOTKEY_ONE);  
UnregisterHotKey(this->m_hWnd, HOTKEY_TWO);  
UnregisterHotKey(this->m_hWnd, HOTKEY_THREE);  
SCAN_Close();
```

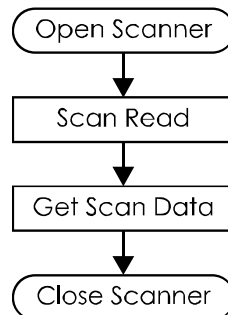
```
}
```

3.2 IMAGER (2D)

Supported PDA:

Brand	Restriction
M3 BK10	No restriction
M3 MT10	OS higher than 5050xx
M3 OX10	No restriction
M3 PS10	Not supported
M3 ST10 WM	No restriction
M3 ST10 CE	Not supported
M3 UL10	No restriction

Basic imager flow chart:



Tip !!

- I. Imager Driver and Camera Driver use the same Quick Capture Interface of CPU. This interface cannot be used at the same time. Because of unloading imager driver while running the camera program so that Imager cannot be used.
- II. Its Virtual Keys are VK_F22 for WinCE and VK_F14 for Windows Mobile.
- III. Scan Data can be obtained through Windows Message and it can be checked in Scanner.h.
#define WM_SCAN_DATA WM_APP+350

Open Scanner

```
// Scanner Open
if(IMAGER_Open () == FALSE)
{
    ::MessageBox(NULL, L"Error : IMAGER_Open ()", NULL, MB_TOPMOST);
}
m_DeviceType = IMAGER_GetDeviceType ();
// Enable All Symbolologies
IMAGER_SetSymbologyAll();
// Set Scanner Key
```

```

if((m_DeviceType == DEVICE_M3SMART_CE) || (m_DeviceType == DEVICE_M3GREEN) || (m_DeviceType == DEVICE_M3T)
|| (m_DeviceType == DEVICE_M3POS))
{
    // Only for WinCE
    MoveWindow(0, 0, 240, 320);
    m_bWinCE = TRUE;
    RegisterHotKey(this->m_hWnd, HOTKEY_ONE, MOD_KEYUP, VK_F22);
    RegisterHotKey(this->m_hWnd, HOTKEY_TWO, MOD_KEYUP, 0);
    RegisterHotKey(this->m_hWnd, HOTKEY_THREE, MOD_KEYUP, 0);
}
else
{
    m_bWinCE = FALSE;
    RegisterHotKey(this->m_hWnd, HOTKEY_ONE, MOD_KEYUP, VK_F14);
    RegisterHotKey(this->m_hWnd, HOTKEY_TWO, MOD_KEYUP, 0);
    RegisterHotKey(this->m_hWnd, HOTKEY_THREE, MOD_KEYUP, 0);
}

```

Scan Read

```

void CImagerTestDlg::OnBnClickedBtnScan()
{
    IMAGER_Read();
}

```

Get ScanData

```

BEGIN_MESSAGE_MAP(CScanTestDlg, CDialog)
    //}}AFX_MSG_MAP
    ON_MESSAGE(WM_SCAN_DATA, OnScanRead)
END_MESSAGE_MAP()

:

LRESULT CImagerTestDlg::OnScanRead(WPARAM wParam, LPARAM lParam)
{
    if(m_bNewForm == TRUE)
        return FALSE;
    TCHAR szBarType[1024] = {0, };
    TCHAR szBarData[1024] = {0, };
    IMAGER_GetScanData(szBarType, szBarData);
    m_strStaticType = szBarType;
    m_strEditData = szBarData;
    UpdateData(FALSE);
    return LRESULT();
}

```

Close Scanner

```

void CImagerTestDlg::OnDestroy()
{

```



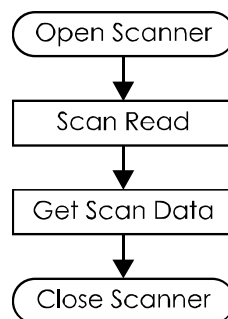
```
CDialog::OnDestroy();  
UnregisterHotKey(this->m_hWnd, HOTKEY_ONE);  
UnregisterHotKey(this->m_hWnd, HOTKEY_TWO);  
UnregisterHotKey(this->m_hWnd, HOTKEY_THREE);  
if(IMAGER_Close() == FALSE)  
    ::MessageBox(NULL, L"ERROR : IMAGER_Close()", NULL, MB_TOPMOST);  
}
```

3.3 LRSCANNER(Long-Range)

Supported PDA:

Brand	Restriction
M3-BK10	Long-range is not supported
M3-MT10	Long-range is not supported
M3-OX10	Long-range is not supported
M3-PS10	Long-range is not supported
M3-ST10	Long-range is not supported
M3-UL10	No restriction

Basic long-range scanner flow chart:



Tip !!

- I. Scanner communicates through serial so that other programs cannot use scanner while scanner is being run. For **Example**, it occurs an error when other program access scanner while LRScanEmul is running.
- II. LRScanner, H/W Decoder Scanner, takes longer than S/W Decoder when opening and closing scanner. It is recommended that program starts with opening LRScanner and ends with closing it.
- III. Its Virtual Key is VK_F14 for Windows Mobile OS and VK_22 for CE 6.0 OS.
- IV. Scan Data can be obtained through Windows Message and it can be checked in LRScanner.h.

```
#define WM_SCAN_DATA WM_APP+350
```

Open Scanner

```
// Scanner Open
if(LRSCAN_Open () == FALSE)
{
    ::MessageBox(NULL, L"Error : IMAGER_Open ()", NULL, MB_TOPMOST);
}
m_DeviceType = LRSCAN_GetDeviceType ();
// Enable All Symbolologies
```

<pre>LRSCAN_SetSymbologyAll(); // Set Scanner Key if((m_DeviceType == DEVICE_M3UL10_CE) { // Only for WinCE MoveWindow(0, 0, 240, 320); m_bWinCE = TRUE; RegisterHotKey(this->m_hWnd, HOTKEY_ONE, MOD_KEYUP, VK_F22); RegisterHotKey(this->m_hWnd, HOTKEY_TWO, MOD_KEYUP, 0); RegisterHotKey(this->m_hWnd, HOTKEY_THREE, MOD_KEYUP, 0); } Else if((m_DeviceType == DEVICE_MM3) { m_bWinCE = FALSE; RegisterHotKey(this->m_hWnd, HOTKEY_ONE, MOD_KEYUP, VK_F14); RegisterHotKey(this->m_hWnd, HOTKEY_TWO, MOD_KEYUP, 0); RegisterHotKey(this->m_hWnd, HOTKEY_THREE, MOD_KEYUP, 0); }</pre>
Scan Read
<pre>void CLRScanTestDlg::OnBnClickedBtnScan() { LRSCAN_Read(); }</pre>
Get ScanData
<pre>BEGIN_MESSAGE_MAP(CLRScanTestDlg, CDialog) //}}AFX_MSG_MAP ON_MESSAGE(WM_SCAN_DATA, OnScanRead) END_MESSAGE_MAP() : LRESULT CLRScanTestDlg::OnScanRead(WPARAM wParam, LPARAM lParam) { if(m_bNewForm == TRUE) return FALSE; TCHAR szBarType[1024] = {0, }; TCHAR szBarData[1024] = {0, }; LRSCAN_GetScanData(szBarType, szBarData); m_strStaticType = szBarType; m_strEditData = szBarData; UpdateData(FALSE); return LRESULT(); }</pre>
Close Scanner
<pre>void CLRScanTestDlg::OnDestroy()</pre>

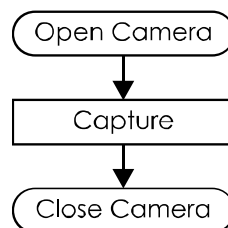
```
{  
    CDialog::OnDestroy();  
    UnregisterHotKey(this->m_hWnd, HOTKEY_ONE);  
    UnregisterHotKey(this->m_hWnd, HOTKEY_TWO);  
    UnregisterHotKey(this->m_hWnd, HOTKEY_THREE);  
    LRSCAN_Close();  
}
```

3.4 CAMERA (CE)

Supported PDA:

Brand	Restriction
M3-BK10-WM	Windows Mobile base brand
M3 BK10 CE	No restriction
M3 MT10	No restriction
M3-OX10-WM	Windows Mobile base brand
M3 OX10 CE	No restriction
M3 PS10	No restriction
M3-ST10-WM	Windows Mobile base brand
M3 ST10 CE	No restriction
M3 UL10	No restriction

Basic camera flow chart:



Tip !!

- I. Its Virtual Key is VK_23.
- II. File name can be editable when capturing pictures. File is stored with file name that users input, otherwise it will be stored by date as default.
- III. CapStatusEvent function is for current status of capturing pictures and name of picture can be obtained through CAM_GetLastSaveFilePath function when capturing is done completely.
- IV. EXIF information can be inserted when enabling option for EXIF information. GPS information can be included in EXIF information and for that, GPS supported device is required. After enabling option for GPS information, GPS information can be confirmed when receiving the grid information from the satellite.
- V. Self Auto-Focus function can be used in devices that AF function is available such as M3 T. It recognizes changes in the preview screen automatically and perform focusing automatically.

Open Camera
<pre>MODEL_TYPE model = CAM_Open(m_hWnd, m_ctrPreview.m_hWnd) ; if(m_Model == MODEL_UNKNOWN) {</pre>

```

::MessageBox(NULL, L"Open error", L"Open", NULL);

return;
}

```

Capture Still

```

TCHAR m_tzSavePath[256] = {0x00};
if(!m_bCapturing)
{
    CAM_Capture(m_tzSavePath);
}

```

Preview

```

BOOL PreviewStart(BOOL bOn)
{
    if(bOn == TRUE)
    {
        CAM_PreviewStart();
    }
    else
    {
        CAM_PreviewStop();
    }
}

```

Insert Exif Information

```

#define WM_SHOW_CAP_STATUS      (WM_USER + 5)
BEGIN_MESSAGE_MAP(CCamTestDlg, CDialog)
    ON_MESSAGE(WM_SHOW_CAP_STATUS, OnShowCaptureStatus)
END_MESSAGE_MAP()
BOOL Init()
{
    HANDLE m_hCapStatusEvent = CreateEvent(NULL, FALSE, 0, _T("CapStatusEvent"));
    if (m_hCapStatusEvent == NULL)
        return FALSE;
}
DWORD ThreadFuncCapStatus(LPVOID pParam)
{
    DWORD CapStatus;
    while(!m_bThreadCapStatusExit) {
        WaitForSingleObject(m_hCapStatusEvent, INFINITE);
        if (m_bThreadCapStatusExit) {
            return 0;
        }
        CapStatus = GetEventData(m_hCapStatusEvent);
        PostMessage(WM_SHOW_CAP_STATUS, CapStatus, 0);
    }
}

```

```

return 0;
}

LRESULT CCamTestDlg::OnShowCaptureStatus(WPARAM wParam, LPARAM lParam)
{
    switch (wParam) {
        case CAP_STATUS_END:
            m_bCapturing= FALSE;
            TCHAR tzFileName[MAX_PATH];
            memset(tzFileName, 0x00, sizeof(TCHAR) * MAX_PATH);
            CAM_GetLastSaveFilePath(tzFileName);
            if(_tcsstr(tzFileName, _T(".jpg"))!=NULL)
                if(m_onEXIF)
                    InsertExifInform();
            break;
        default:
            break;
    }
    return 0L;
}

BOOL InsertExifInform()
{
    ExifInfo info={0,};
    TCHAR OutFileBuf[260]={0,};
    TCHAR tzFileName[260]={0,};
    if(!CAM_GetLastSaveFilePath(tzFileName))
        return FALSE;
    _tcscpy(OutFileBuf, tzFileName);
    TCHAR* ImageName;
    ImageName=_tcstok(OutFileBuf, _T("\\"));
    while (ImageName!=NULL)
    {
        if(_tcsstr(ImageName, _T(".jpg"))!=NULL || _tcsstr(ImageName, _T(".JPG"))!=NULL)
            break;
        ImageName=_tcstok(NULL, _T("\\"));
    }
    _tcscpy(info.TitleName, ImageName);
    _tcscpy(info.Make, _T("M3 Mobile Co.Ltd"));
    switch(m_Model)
    {
        case MODEL_SMART:
            _tcscpy(info.Model, _T("M3 Smart"));
            break;
        case MODEL_POS:
            _tcscpy(info.Model, _T("M3 POS"));
            break;
    }
}

```

```
case MODEL_GREEN:
    _tcscpy(info.Model, _T("M3 Green"));
    break;
case MODEL_T:
    _tcscpy(info.Model, _T("M3 T"));
    break;
}
if(CAM_InsertExifInformation(tzFileName, info))
{
    return TRUE;
}
else
{
    return FALSE;
}
}
```

Close Camra

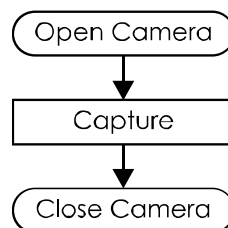
CAM_Close();

3.5 CAMERA (WM)

Supported PDA:

Brand	Restriction
M3 BK10 WM	No restriction
M3 BK10 CE	Windows CE base brand
M3 MT10	Windows CE base brand
M3 OX10 WM	No restriction
M3 OX10 CE	Windows CE base brand
M3 PS10	Windows CE base brand
M3 ST10 WM	No restriction
M3 ST10 CE	Windows CE base brand
M3 UL10	Windows CE base brand

Basic camera flow chart:



Tip !!

- I. Combined version of Camera.dll for Windows Mobile OS.
- II. Its Virtual Key is VK_13.
- III. File name can be editable when capturing pictures. File is stored with file name that users input, otherwise it will be stored by date as default.
- IV. EXIF information can be inserted when enabling option for EXIF information. GPS information can be included in EXIF information and for that, GPS supported device is required. After enabling option for GPS information, GPS information can be confirmed when receiving the grid information from the satellite.
- V. Self Auto-Focus function can be used in devices that AF function is available such as M3 SKY and M3 ORANGE. It recognizes changes in the preview screen automatically and perform focusing automatically.

Open Camera
<pre>m_CameraMode = STILL_MODE; // or VIDEO_MODE m_VideoType = VIDEO_WMV; // or VIDEO_ASF CAM_Open(m_hWnd, m_CameraMode, m_VideoType);</pre>

Capture Still

```
TCHAR tzFile[MAX_PATH] = {0x00};
TCHAR tzOutFile[MAX_PATH] = {0x00};
memset(tzFile, 0x00, sizeof(TCHAR) * MAX_PATH);
memset(tzOutFile, 0x00, sizeof(TCHAR) * MAX_PATH);
wsprintf(tzFile, L"\\Flash Disk\\Cam\\Picture1.jpg");
m_bAutolnit = TRUE;
CAM_Capture(tzFile, tzOutFile, m_bAutolnit);
```

Preview

```
BOOL PreviewStart(BOOL bStart)
{
    if(bStart)
    {
        CAM_PreviewStart();
    }
    else
    {
        CAM_PreviewStop();
    }
}
```

Insert Exif Information

```
BOOL InsertExifInform(TCHAR * tzFileName)
{
    ExifInfo info={0,};
    TCHAR OutFileBuf[260]={0,};
    _tcscpy(OutFileBuf, tzFileName);
    TCHAR* ImageName;
    ImageName=_tcstok(OutFileBuf, _T("\\"));
    while (ImageName!=NULL)
    {
        if(_tcsstr(ImageName, _T(".jpg"))!=NULL || _tcsstr(ImageName, _T(".JPG"))!=NULL)
            break;
        ImageName=_tcstok(NULL, _T("\\"));
    }
    _tcscpy(info.TitleName, ImageName);
    _tcscpy(info.Make, _T("M3 Mobile Co.Ltd"));
    switch(m_ModelType)
    {
        case MODEL_MM3:
            _tcscpy(info.Model, _T("MM3"));
            break;
        case MODEL_SMART:
            _tcscpy(info.Model, _T("M3 Smart"));
            break;
    }
}
```

```
        break;
    case MODEL_SKY:
        _tcscpy(info.Model, _T("M3 Sky"));
        break;
    case MODEL_ORANGE:
        _tcscpy(info.Model, _T("M3 Orange"));
        break;
    case MODEL_UNKONWN:
        _tcscpy(info.Model, _T("M3"));
        break;
    }
    if(CAM_InsertExifInformation(tzFileName, info))
    {
        RETAILMSG(RT_MSG, (_T("CamTest - Insert EXIF Success!!\n")));
        return TRUE;
    }
    else
    {
        return FALSE;
    }
}
```

Close Camra

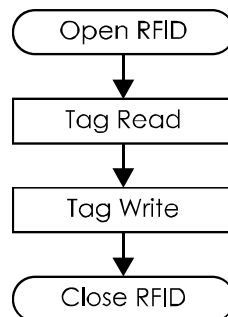
CAM_Close();

3.6 RFID (LF/HF)

Supported PDA:

Brand	Restriction
M3-BK10	Not supported
M3-MT10	Not supported
M3-O10	HF RFID modules is installed. UHF is not supported
M3-PS10	Not supported
M3-ST10	Not supported
M3-UL10	Not supported

Basic RFID flow chart:



Tip !!

- I. Power when using RFID can be reduced by Antenna On/Off.
- II. Read/Write speed can be improved when setting the tag type to read.
- III. RFID_SendContinuousRead and RFID_GetData functions are suitable if using Serial Number only. There is RFID_SelectTag function that has same function.
- IV. The latest Firmware version of RFID module is 1.2.5 and FW upgrade is not supported through module itself.
- V. RFID Close cuts power with RFID_PowerSupply function and closes program.

Open RFID

```
RFID_TYPE m_RfType = RFID_Open();
TCHAR tzRadioType[256] = {0x00};
if(m_RfType == RFID_LF)
{
    wsprintf(tzRadioType, L"Low Frequency");
}
else if(m_RfType == RFID_HF)
{
    wsprintf(tzRadioType, L"High Frequency");
}
```

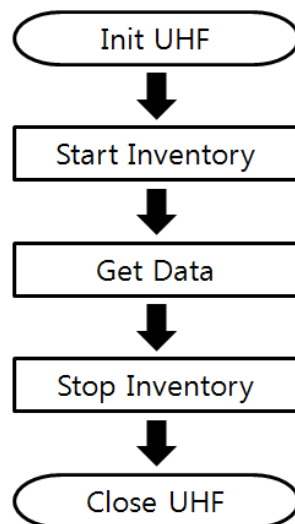
<pre>} else { return; }</pre>
Antenna On/Off
<pre>if(bAntennaOn == TRUE) { RFID_SetAntenna(TRUE); } else { RFID_SetAntenna(FALSE); }</pre>
Tag Select
<pre>TCHAR tzSerial[100] = {0x00}; RFID_SelectTag(tzSerial);</pre>
Data Read
<pre>TCHAR tzSerial[32] = {0x00}; TCHAR tzData[nMaxData] = {0x00}; memset(tzSerial, 0x00, sizeof(TCHAR) * 32); memset(tzData, 0x00, sizeof(TCHAR) * nMaxData); if(RFID_SelectTag(tzSerial)) { RFID_ReadBlock(L"01", tzData); }</pre>
Data Write
<pre>TCHAR tzSerial[32] = {0x00}; TCHAR tzOutData[nMaxData] = {0x00}; memset(tzSerial, 0x00, sizeof(TCHAR) * 32); memset(tzOutData, 0x00, sizeof(TCHAR) * 1024); if(RFID_SelectTag(tzSerial)) { RFID_WriteBlock(L"01", L"00112233", tzOutData); }</pre>

3.7 UHF Gun Reader

Supported PDA:

Brand	Restriction
M3-BK10	Not supported
M3-MT10	Not supported
M3-O10	Need to have UGR Gun
M3-PS10	Not supported
M3-ST10	Not supported
M3-UL10	Not supported

Basic UHF GUN READER flowchart is shown below:



< UHF RFID flow chart >

Tip !!

- i. It can be used with both Windows Mobile and Windows CE platforms.
- ii. Initialize will return fail in the following conditions:
 - PDA detached from gun reader
 - RFID / SCAN switch is at SCAN position
 - Gun reader's battery is detached or empty
 - PDA in gun reader is synced with PC
- iii. Power status can be obtained using delegate ReceivedPower. When turning off, UHF RFID must be closed. When turning on, UHF RFID must be initialized.
- iv. UHF RFID uses the same virtual key as scanner; VK_H14 for WM and VK_F22 for CE.

Init UHF

```
BEGIN_MESSAGE_MAP(CUHF_GunTestDlg, CDialog)
    ON_MESSAGE(WM_MSG_POWER, ReceivedPower)
END_MESSAGE_MAP()

BOOL Init()
{
    RFID_STATUS status;
    CString strstatus;

    status = UHF_Init(m_hWnd);
    if(status != RFID_STATUS_OK)
    {
        strstatus.Format(_T("RFID Init Error-[%x]"),status);
        // MessageBox(strstatus);

        return FALSE;
    }
    return TRUE;
}

LRESULT ReceivedPower(WPARAM wParam, LPARAM lParam)
{
    BOOL* bPowerOn = (BOOL*)wParam;

    if(*bPowerOn)
    {
        if(m_bOpen == FALSE)
        {
            Init();
            m_bOpen = TRUE;
        }
    }
    else
    {
        if(m_bOpen == TRUE)
        {
            Close();
            m_bOpen = FALSE;
        }
    }
}
```

<pre>return 0; }</pre>
Start Inventory
<pre>void Inventory() { UHF_Inventory(m_hWnd); }</pre>
Get Data
<pre>BEGIN_MESSAGE_MAP(CUHF_GunTestDlg, CDialog) ON_MESSAGE(WM_MSG_INVENTORY, ReceivedInventory) END_MESSAGE_MAP() LRESULT ReceivedInventory(WPARAM wParam, LPARAM lParam) { int nTaglength = 0; unsigned char czTagData[128] = {0,}; CString strTagData; nTaglength = UHF_GetData(czTagData); if(m_chkCRC.GetCheck()) { for(int i=0;i<nTaglength;i++) { strTagData.AppendFormat(_T("%02X"), czTagData[i]); } } else { for(int i=0;i<nTaglength-2;i++) { strTagData.AppendFormat(_T("%02X"), czTagData[i]); } } return LRESULT(); }</pre>
Stop Inventory
<pre>void InventoryStop() { UHF_InventoryStop(); }</pre>

Close UHF

```
BOOL Close()
{
    RFID_STATUS status;
    CString strstatus;

    status = UHF_Close();
    if(status != RFID_STATUS_OK)
    {
        strstatus.Format(_T("%x"),status);
        // MessageBox(_T("RFID Close Error"), strstatus);

        return FALSE;
    }

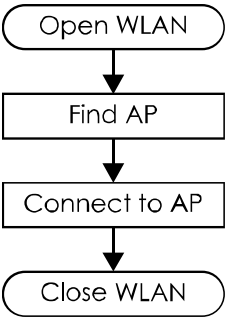
    // MessageBox(L"Close");
    ::SetWindowText(::GetDlgItem(m_hWnd, IDC_STATIC_STATUS), L"Close UHF");
    return TRUE;
}
```

3.8 WLAN

Supported PDA:

Brand	Restriction
M3 BK10	WLAN must be SUMMIT
M3 MT10	WLAN must be SUMMIT
M3 OX10	WLAN must be SUMMIT
M3 PS10	WLAN must be SUMMIT
M3 ST10	WLAN must be SUMMIT
M3 UL10	WLAN must be SUMMIT

Basic WLAN flow chart:



Tip !!

WLANTest.exe functions as same as SCU (Summit Client Utility) and they are synchronized.

Combined version of WLAN.dll can be used in all M3 Summit devices regardless of model type or OS.

Init Wlan
<pre>if(WLAN_Init() == FALSE) { AfxMessageBox(L"Fail to WLAN_Init()"); }</pre>
Searching AP
<pre>WLAN_SSID_LIST* SSIDList=new WLAN_SSID_LIST(); SSIDList->m_SSID=new WLAN_SSID[40]; do { WLAN_GetBssidList(SSIDList); } while(SSIDList->m_NumberOfItems==0);</pre>
Connect Config

<pre>int result=0; // Security result=WLAN_ConnectAPEX (_T("SSID"), _T("12345"), Encryption Type, 2); if(result!=0) AfxMessageBox(_T("Fail to WLAN_ConnectAPEX ()")); // Open result=WLAN_ConnectAPEX (_T("SSID"), _T("\0"), 0, 2); AfxMessageBox(_T("Fail to WLAN_ConnectAPEX()"));</pre>
Delete Config
<pre>int result=WLAN_DeleteConfig(_T("SSID"); if(!result) AfxMessageBox(_T("Activate config can not removed!!"));</pre>
Change Config
<pre>if result=WLAN_ActivateConfig(_T("SSID"); if(!result) AfxMessageBox(_T("Fail to WLAN_ActivateConfig()"));</pre>
Export Configuration
<pre>TCHAR szFilter[]=_T("SDC files(*.sdc) *.sdc All files(*.*) *.* "); CFileDialog SaveDlg(FALSE, _T("sdc"), _T("SummitSettings.sdc"), OFN_HIDEREADONLY, szFilter); if (IDOK==SaveDlg.DoModal()) m_filepath=SaveDlg.GetPathName(); if(WLAN_ExportConfig(m_filepath.GetBuffer())) m_result=_T("Export Success!!"); else m_result=_T("Export Fail!!"); UpdateData(FALSE);</pre>
WLAN Power On
<pre>BOOL result= WLAN_PowerOn(); if(!result) AfxMessageBox(_T("Fail to WLAN_PowerOn()"));</pre>
WLAN Power Off
<pre>BOOL result= WLAN_PowerOff(); if(!result) AfxMessageBox(_T("Fail to WLAN_PowerOff()"));</pre>
Import Configuration
<pre>TCHAR szFilter[]=_T("SDC files(*.sdc) *.sdc All files(*.*) *.* "); CFileDialog LoadDlg(TRUE, _T("sdc"), _T("SummitSettings.sdc"), OFN_HIDEREADONLY, szFilter); if (IDOK==LoadDlg.DoModal()) m_filepath=LoadDlg.GetPathName(); if(WLAN_ImportConfig(m_filepath.GetBuffer()))</pre>

```
        m_result=_T("Import Success!!");  
    else  
        m_result=_T("Import Fail!!");  
    UpdateData(FALSE);
```

Close WLAN

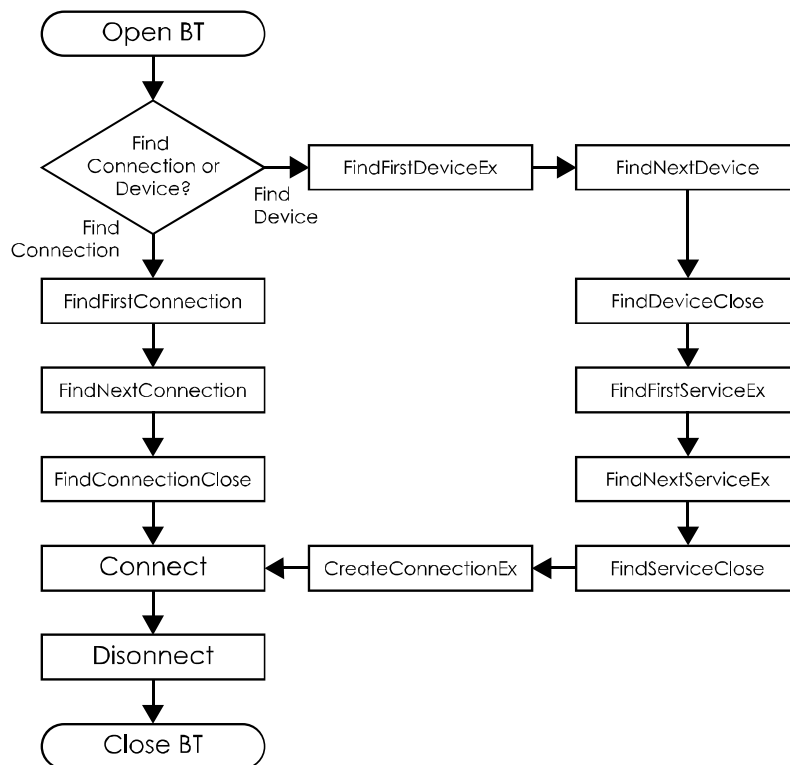
```
BOOL result=WLAN_Close();  
if(!result)  
    AfxMessageBox(_T("Fail to WLAN_Close()"));
```

3.9 BLUETOOTH

Supported PDA:

Brand	Restriction
M3 BK10	BTExplorer version 2.1.1 and Build version 27460 or higher
M3 MT10	BTExplorer version 2.1.1 and Build version 27460 or higher
M3 OX10	BTExplorer version 2.1.1 and Build version 27460 or higher
M3 PS10	BTExplorer version 2.1.1 and Build version 27460 or higher
M3 ST10 WM	BTExplorer version 2.1.1 and Build version 27460 or higher
M3 ST10 CE	Not supported
M3 UL10	BTExplorer version 2.1.1 and Build version 27460 or higher

Basic Bluetooth flow chart:



Tip !!

- I. Bluetooth SDK can be applied to upper level from StonestreetOne BTExplorer Version 2.1.1 and Build Version 27460.
- II. Bluetooth communicates through Serial with COM port 9.
- III. BT can provide following services;
 - AVC Audio/Video Control Transport Protocol Sample Application
 - AVR Audio/Video Remove Control Profile Sample Application
 - BTC Generic Bluetooth COM Profile Sample Application
 - DUN Dial-Up Networking Profile Sample Application
 - FTP OBEX File Transfer Profile Sample Application

GAV Generic Audio/Video Profile Sample Application
HDS Headset Profile Sample Application
OBP OBEX Object Push Profile Sample Application
PAN Personal Area Networking Profile Sample Application
SDP Sample SDP Application using Bluetopia
SPP Sample SPP Application using Bluetopia

Initialization

```
// Bluetooth Open
void CBluetoothTestDlg::OnBnClickedBtnOpen()
{
    TCHAR* result = NULL;
    if(BLUETOOTH_Open())
    {
        m_List.DeleteAllItems();
        m_State.SetWindowText(L"Open Success");
        g_bBluetoothOpen = TRUE;
        m_Open.EnableWindow(FALSE);
        m_DeviceFind.EnableWindow(TRUE);
        m_ServiceFind.EnableWindow(FALSE);
        m_CreateConn.EnableWindow(FALSE);
        m_ConnFind.EnableWindow(TRUE);
        m_Connect.EnableWindow(FALSE);
        m_Disconnect.EnableWindow(FALSE);
        m_DeleteConn.EnableWindow(FALSE);
        m_Close.EnableWindow(TRUE);
        m_AllDelDevice.EnableWindow(TRUE);
        m_LocalInfo.EnableWindow(TRUE);
    }
    else
        m_State.SetWindowText(L"Open Fail");
}
```

Device Find

```
// Bluetooth Device Find
void CBluetoothTestDlg::OnBnClickedBtnDeviceFind()
{
    if(g_bBluetoothOpen == TRUE)
    {
        m_listtype = DeviceFind;
        m_List.DeleteAllItems();
        m_Deviceitem.RemoveAll();
        m_Connectitem.RemoveAll();
        memset(&m_connectionInfo, 0, sizeof(m_connectionInfo));
    }
}
```

```

m_connectionInfo.Size = sizeof(BTP_Connection_Info_Ex_t);
m_connectionInfo.ProfileType = BTP_PROFILE_SPP;
CString strAddr;
CString strName;
BTP_Device_Find deviceFind;
BTP_Device_Info_t deviceInfo;
BTP_Device_Query_Ex_t deviceQuery;
memset(&deviceQuery, 0, sizeof(deviceQuery));
deviceQuery.Size = sizeof(BTP_Device_Query_Ex_t);
deviceQuery.DeviceAttributes = BTP_DEVICE_ALL;
deviceQuery.DeviceType = (m_connectionInfo.ProfileType == BTP_PROFILE_SPP ? bdAll : bdHID);
deviceQuery.InquiryTimeout = 10;
BeginWaitCursor();
hResult = BLUETOOTH_FindFirstDeviceEx(&deviceFind, &deviceInfo, &deviceQuery);
if (BTP_ERROR_SUCCESS == hResult)
{
    EndWaitCursor();
    do
    {
        BTP_Device_Info_t deviceInfo_temp;
        memcpy(&deviceInfo_temp, &deviceInfo, sizeof(BTP_Device_Info_t));
        m_Deviceitem.AddTail(deviceInfo_temp);
        hResult = BLUETOOTH_FindNextDevice(deviceFind, &deviceInfo);
    } while (BTP_ERROR_SUCCESS == hResult);
}
BLUETOOTH_FindDeviceClose(deviceFind);
int Importdevicecount = m_Deviceitem.GetCount();
int i = 0;
for(i = Importdevicecount-1; i >= 0; i--)
{
    deviceInfo = m_Deviceitem.GetAt(m_Deviceitem.FindIndex(i));
    strName.Format(L"%s", CString(deviceInfo.Name));
    m_List.InsertItem(0, strName, 0);
    strAddr.Format(L"%02X:%02X:%02X:%02X:%02X:%02X",
        deviceInfo.BD_ADDR.BD_ADDR5,
        deviceInfo.BD_ADDR.BD_ADDR4,
        deviceInfo.BD_ADDR.BD_ADDR3,
        deviceInfo.BD_ADDR.BD_ADDR2,
        deviceInfo.BD_ADDR.BD_ADDR1,
        deviceInfo.BD_ADDR.BD_ADDR0);
    m_List.SetItem(0, 1, LVIF_TEXT, strAddr, NULL, NULL, NULL, NULL);
}
m_State.SetWindowText(L"Device Find");
g_bAllDelDevice = FALSE;
m_Open.EnableWindow(FALSE);
m_DeviceFind.EnableWindow(FALSE);

```

```

        m_ServiceFind.EnableWindow(TRUE);
        m_CreateConn.EnableWindow(FALSE);
        m_ConnFind.EnableWindow(TRUE);
        m_Connect.EnableWindow(FALSE);
        m_Disconnect.EnableWindow(FALSE);
        m_DeleteConn.EnableWindow(FALSE);
        m_Close.EnableWindow(TRUE);
        m_AllDelDevice.EnableWindow(TRUE);
        m_LocalInfo.EnableWindow(TRUE);
    }
}

```

Service Find

```

// Bluetooth Service Find
void CBluetoothTestDlg::OnBnClickedBtnServiceFind()
{
    if(g_bBluetoothOpen == TRUE && g_bAllDelDevice == FALSE)
    {
        m_listtype = ServiceFind;
        m_List.DeleteAllItems();
        BTP_Service_Find    serviceFind;
        BTP_Service_Info_Ex_t    serviceInfo;
        BTP_Service_Query_t    serviceQuery;
        SDP_UUID_Entry_t    service[1];
        CString                strname;
        serviceInfo.Size = sizeof(BTP_Service_Info_Ex_t);
        memset(&serviceQuery, 0, sizeof(serviceQuery));
        serviceQuery.BD_ADDR = m_connectionInfo.BD_ADDR;
        serviceQuery.NumberServiceUUID = 1;
        serviceQuery.Service = service;
        service[0].SDP_Data_Element_Type = deUUID_16;
        switch(m_connectionInfo.ProfileType)
        {
        case BTP_PROFILE_SPP:
            if (128 == serviceUUID)
            {
                service[0].SDP_Data_Element_Type = deUUID_128;
                ASSIGN_UUID_128((service[0].UUID_Value.UUID_128),
                    0x85, 0x60, 0xCA, 0x18,    // 0x8560CA18
                    0x62, 0x3F,                // 0x623f
                    0x4A, 0x77,                // 0x4a77
                    0x9F, 0x5E, 0x3C, 0x52, 0x66, 0x68, 0x05, 0x1A); // 0x9f, 0x5e, 0x3c, 0x52, 0x66, 0x68, 0x05, 0x1a
            }
            else
            {
                ASSIGN_UUID_16((service[0].UUID_Value.UUID_16), 0x11, 0x01)
            }
        }
    }
}

```



```

    }
    break;
case BTP_PROFILE_HID_HOST:
case BTP_PROFILE_HID_DEVICE:
    ASSIGN_UUID_16((service[0].UUID_Value.UUID_16), 0x11, 0x24)
    break;
}
BeginWaitCursor();
hResult = BLUETOOTH_FindFirstServiceEx(&serviceFind, &serviceInfo, &serviceQuery);
if (BTP_ERROR_SUCCESS == hResult)
{
    EndWaitCursor();
    do
    {
        m_connectionInfo.ProfileType = serviceInfo.ProfileType;
        m_connectionInfo.MajorVersion = serviceInfo.MajorVersion;
        m_connectionInfo.MinorVersion = serviceInfo.MinorVersion;
        switch(m_connectionInfo.ProfileType)
        {
            case BTP_PROFILE_UNKNOWN:
                m_connectionInfo.ProfileInformation.RemoteSerialPortProfileInfo.RFCOMMServerPort =
                serviceInfo.ProfileInformation.RemoteSerialPortProfileInfo.RFCOMMServerPort;

                m_connectionInfo.ProfileInformation.RemoteSerialPortProfileInfo.UseActiveSync =
                serviceInfo.ProfileInformation.RemoteSerialPortProfileInfo.UseActiveSync;

                break;

            case BTP_PROFILE_SPP:
                m_connectionInfo.ProfileInformation.RemoteSerialPortProfileInfo.RFCOMMServerPort =
                serviceInfo.ProfileInformation.RemoteSerialPortProfileInfo.RFCOMMServerPort;

                m_connectionInfo.ProfileInformation.RemoteSerialPortProfileInfo.UseActiveSync =
                serviceInfo.ProfileInformation.RemoteSerialPortProfileInfo.UseActiveSync;

                break;

            case BTP_PROFILE_HID_HOST:
            case BTP_PROFILE_HID_DEVICE:
                m_connectionInfo.ProfileInformation.RemoteHIDProfileInfo.DeviceAutomaticReconnect =
                serviceInfo.ProfileInformation.RemoteHIDProfileInfo.DeviceAutomaticReconnect;

                m_connectionInfo.ProfileInformation.RemoteHIDProfileInfo.DeviceNormallyConnectable=
                serviceInfo.ProfileInformation.RemoteHIDProfileInfo.DeviceNormallyConnectable;

                m_connectionInfo.ProfileInformation.RemoteHIDProfileInfo.DeviceSubclass =
                serviceInfo.ProfileInformation.RemoteHIDProfileInfo.DeviceSubclass;

                m_connectionInfo.ProfileInformation.RemoteHIDProfileInfo.L2CAPControlChannel=
                serviceInfo.ProfileInformation.RemoteHIDProfileInfo.L2CAPControlChannel;

                m_connectionInfo.ProfileInformation.RemoteHIDProfileInfo.L2CAPInterruptChannel =
                serviceInfo.ProfileInformation.RemoteHIDProfileInfo.L2CAPInterruptChannel;

                m_connectionInfo.ProfileInformation.RemoteHIDProfileInfo.VirtualCableSupported =
                serviceInfo.ProfileInformation.RemoteHIDProfileInfo.VirtualCableSupported;

                break;
        }

        strname.Format(L"%s", (CString)serviceInfo.ServiceName);
        strname = serviceInfo.ServiceName;
    }
}

```

```

        m_List.InsertItem(0, strname, 0);

        hResult = BLUETOOTH_FindNextServiceEx(serviceFind, &serviceInfo);
    } while (BTP_ERROR_SUCCESS == hResult);
    if (BTP_ERROR_NO_MORE != hResult)
        m_State.SetWindowText(L"Failed to Find Next Service");
    BLUETOOTH_FindServiceClose(serviceFind);
}
m_State.SetWindowText(L"Service Find");
m_Open.EnableWindow(FALSE);
m_DeviceFind.EnableWindow(TRUE);
m_ServiceFind.EnableWindow(FALSE);
m_CreateConn.EnableWindow(TRUE);
m_ConnFind.EnableWindow(TRUE);
m_Connect.EnableWindow(FALSE);
m_Disconnect.EnableWindow(FALSE);
m_DeleteConn.EnableWindow(FALSE);
m_Close.EnableWindow(TRUE);
m_AllDelDevice.EnableWindow(FALSE);
m_LocalInfo.EnableWindow(TRUE);
}
else if(g_bAllDelDevice == TRUE)
{
    m_State.SetWindowText(L"Please, 'Device Find' is re-click.");
}
}

```

Connection Management

```

// Bluetooth Connection Management - (1) Create Connection
void CBluetoothTestDlg::OnBnClickedBtnCreateConnection()
{
    if(g_bBluetoothOpen == TRUE)
    {
        m_List.DeleteAllItems();
        m_connectionInfo.ConnectionAttributes = BTP_CONNECTION_REMEMBERED | BTP_CONNECTION_ACTIVE;
        m_connectionInfo.LocalCOMPort = 9;
        m_connectionInfo.ProfileType = BTP_PROFILE_SPP;
        hResult = BLUETOOTH_CreateConnectionEx(&m_connectionInfo);
        if (BTP_ERROR_SUCCESS == hResult)
            m_State.SetWindowText(L"Create connection");
        else
            m_State.SetWindowText(L"Create Connection Error");
        m_Open.EnableWindow(FALSE);
        m_DeviceFind.EnableWindow(TRUE);
        m_ServiceFind.EnableWindow(TRUE);
        m_CreateConn.EnableWindow(FALSE);
        m_ConnFind.EnableWindow(TRUE);
    }
}

```

```

        m_Connect.EnableWindow(FALSE);
        m_Disconnect.EnableWindow(FALSE);
        m_DeleteConn.EnableWindow(FALSE);
        m_Close.EnableWindow(TRUE);
        m_AllDelDevice.EnableWindow(FALSE);
        m_LocalInfo.EnableWindow(TRUE);
    }
}

// Bluetooth Connection Management - (2) Delete Connection
void CBluetoothTestDlg::OnBnClickedBtnDeleteConnection()
{
    if(g_bBluetoothOpen == TRUE)
    {
        m_List.DeleteAllItems();
        hResult = BLUETOOTH_DeleteConnection(m_connectionInfo.ConnectionID);
        if (BTP_ERROR_SUCCESS == hResult)
            m_State.SetWindowText(L"Delete Connection");
        else
            m_State.SetWindowText(L"Delete Connection Error");
        m_Open.EnableWindow(FALSE);
        m_DeviceFind.EnableWindow(TRUE);
        m_ServiceFind.EnableWindow(FALSE);
        m_CreateConn.EnableWindow(FALSE);
        m_ConnFind.EnableWindow(TRUE);
        m_Connect.EnableWindow(FALSE);
        m_Disconnect.EnableWindow(FALSE);
        m_DeleteConn.EnableWindow(FALSE);
        m_Close.EnableWindow(TRUE);
        m_AllDelDevice.EnableWindow(TRUE);
        m_LocalInfo.EnableWindow(TRUE);
    }
}

// Bluetooth Connection Management - (3) Connection Find
void CBluetoothTestDlg::OnBnClickedBtnConnectionFind()
{
    if(g_bBluetoothOpen == TRUE)
    {
        m_List.DeleteAllItems();
        m_Deviceitem.RemoveAll();
        m_Connectitem.RemoveAll();
        m_listtype = ConnectionFind;
        CString strID;
        CString strAddr;
        BTP_Connection_Find    connectFind;
        BTP_Connection_Info_t  connectionInfo;
        BTP_Connection_Query_t connectQuery;
    }
}

```

```

BTP_Connection_Info_t connectInfo_temp;
memset(&connectQuery, 0, sizeof(connectQuery));
connectQuery.ConnectionAttributes = BTP_CONNECTION_REMEMBERED;
BeginWaitCursor();
hResult = BLUETOOTH_FindFirstConnection(&connectFind, &connectionInfo, &connectQuery);
if(BTP_ERROR_NO_MORE == hResult)
{
    EndWaitCursor();
    m_State.SetWindowText(L"PDA doesn't have a 'Connection'.");
    return;
}
else if (BTP_ERROR_SUCCESS == hResult)
{
    EndWaitCursor();
    do
    {
        memcpy(&connectInfo_temp,&connectionInfo,sizeof(BTP_Connection_Info_t));
        m_Connectitem.AddTail(connectInfo_temp);
        hResult = BLUETOOTH_FindNextConnection(connectFind, &connectionInfo);
    } while (BTP_ERROR_SUCCESS == hResult);
}
BLUETOOTH_FindConnectionClose(connectFind);
int ImportDeviceCount = m_Connectitem.GetCount();
int i = 0;
for(i = ImportDeviceCount-1;i>=0;i--)
{
    connectionInfo = m_Connectitem.GetAt(m_Connectitem.FindIndex(i));
    strID.Format(L"%d", connectionInfo.ConnectionID);
    m_List.InsertItem(0, strID, 0);
    strAddr.Format(L"%02X:%02X:%02X:%02X:%02X:%02X",
        connectionInfo.BD_ADDR.BD_ADDR5,
        connectionInfo.BD_ADDR.BD_ADDR4,
        connectionInfo.BD_ADDR.BD_ADDR3,
        connectionInfo.BD_ADDR.BD_ADDR2,
        connectionInfo.BD_ADDR.BD_ADDR1,
        connectionInfo.BD_ADDR.BD_ADDR0);
    m_List.SetItem(0, 1, LVIF_TEXT, strAddr, NULL, NULL, NULL, NULL);
}
m_State.SetWindowText(L"Connection Find");
m_Open.EnableWindow(FALSE);
m_DeviceFind.EnableWindow(FALSE);
m_ServiceFind.EnableWindow(FALSE);
m_CreateConn.EnableWindow(FALSE);
m_ConnFind.EnableWindow(FALSE);
m_Connect.EnableWindow(TRUE);
m_Disconnect.EnableWindow(FALSE);

```

```

        m_DeleteConn.EnableWindow(TRUE);
        m_Close.EnableWindow(TRUE);
        m_AllDelDevice.EnableWindow(FALSE);
        m_LocalInfo.EnableWindow(TRUE);
    }
}

```

Connect

```

// Bluetooth Connect
void CBluetoothTestDlg::OnBnClickedBtnConnect()
{
    if(g_bBluetoothOpen == TRUE)
    {
        m_List.DeleteAllItems();
        hResult = BLUETOOTH_Connect(m_connectionInfo.ConnectionID);
        CreateFile((LPCWSTR)"COM9:", GENERIC_READ | GENERIC_WRITE, 0, NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL,
        NULL);
        if(BTP_ERROR_SUCCESS == hResult)
        {
            m_State.SetWindowText(L"Connect");
        }
        else if(BTP_ERROR_INVALID_PARAMETER == hResult)
        {
            m_State.SetWindowText(L"Error Invalid parameter");
        }
        else
        {
            m_State.SetWindowText(L"Connect Error");
        }
        m_Open.EnableWindow(FALSE);
        m_DeviceFind.EnableWindow(TRUE);
        m_ServiceFind.EnableWindow(FALSE);
        m_CreateConn.EnableWindow(FALSE);
        m_ConnFind.EnableWindow(FALSE);
        m_Connect.EnableWindow(FALSE);
        m_Disconnect.EnableWindow(TRUE);
        m_DeleteConn.EnableWindow(FALSE);
        m_Close.EnableWindow(TRUE);
        m_AllDelDevice.EnableWindow(FALSE);
        m_LocalInfo.EnableWindow(TRUE);
    }
}

```

Disconnect

```

// Bluetooth Disconnect

```

```

void CBluetoothTestDlg::OnBnClickedBtnDisconnect()
{
    if(g_bBluetoothOpen == TRUE)
    {
        m_List.DeleteAllItems();
        hResult = BLUETOOTH_Disconnect(m_connectionInfo.ConnectionID);
        if(BTP_ERROR_SUCCESS == hResult)
        {
            m_State.SetWindowText(L"Disconnect");
        }
        else if(BTP_ERROR_INVALID_PARAMETER == hResult)
        {
            m_State.SetWindowText(L"Error Invalid parameter");
        }
        else
        {
            m_State.SetWindowText(L"Disconnect Error");
        }
        m_Open.EnableWindow(FALSE);
        m_DeviceFind.EnableWindow(TRUE);
        m_ServiceFind.EnableWindow(FALSE);
        m_CreateConn.EnableWindow(FALSE);
        m_ConnFind.EnableWindow(TRUE);
        m_Connect.EnableWindow(FALSE);
        m_Disconnect.EnableWindow(FALSE);
        m_DeleteConn.EnableWindow(TRUE);
        m_Close.EnableWindow(TRUE);
        m_AllDelDevice.EnableWindow(FALSE);
        m_LocalInfo.EnableWindow(TRUE);
    }
}

```

Close

//Bluetooth Close

```
void CBluetoothTestDlg::OnBnClickedBtnClose()
```

```

{
    if(g_bBluetoothOpen == TRUE)
    {
        m_List.DeleteAllItems();
        m_Connectitem.RemoveAll();
        BLUETOOTH_Close();
        m_State.SetWindowText(L"Close Success");
        m_Open.EnableWindow(TRUE);
        m_DeviceFind.EnableWindow(FALSE);
        m_ServiceFind.EnableWindow(FALSE);
        m_CreateConn.EnableWindow(FALSE);
    }
}

```

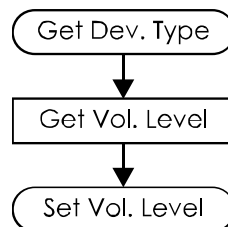
```
m_ConnFind.EnableWindow(FALSE);  
m_Connect.EnableWindow(FALSE);  
m_Disconnect.EnableWindow(FALSE);  
m_DeleteConn.EnableWindow(FALSE);  
m_Close.EnableWindow(FALSE);  
m_AllDelDevice.EnableWindow(FALSE);  
m_LocalInfo.EnableWindow(FALSE);  
}  
g_bBluetoothOpen = FALSE;  
}
```

3.10 SYSTEM SDK

Supported PDA:

Brand	Restriction
M3 BK10	No restriction
M3 MT10	No restriction
M3 OX10	No restriction
M3 PS10	No restriction
M3 ST10	No restriction
M3 UL10	No restriction

Basic system SDK flow chart:



Get Device Type
<pre>DeviceInfo = (DEVICE_INFO)SYSTEM_GetDeviceInfo(); if((DeviceInfo == DEVICE_M3SMARTCE) (DeviceInfo == DEVICE_M3T) (DeviceInfo == DEVICE_Pos)) { // Only for WinCE MoveWindow(0, 0, 240, 320); }</pre>
Get Volume Level
<pre>if(m_DeviceInfo != DEVICE_M3T m_DeviceInfo != DEVICE_Pos) { m_ctlMainVolumeLevel.SetRange(0, 5); m_ctlMainVolumeLevel.SetTicFreq(1); dwVolumeLevel_Cur = SYSTEM_GetVolumeLevel(); switch (dwVolumeLevel_Cur) { case 0: dwVolumeLevel_Cur = 5; break; case 1: dwVolumeLevel_Cur = 4;</pre>


```
        break;
    case 2:
        dwVolumeLevel_Cur = 3;
        break;
    case 3:
        dwVolumeLevel_Cur = 2;
        break;
    case 4:
        dwVolumeLevel_Cur = 1;
        break;
    case 5:
        dwVolumeLevel_Cur = 0;
        break;
    default:
        dwVolumeLevel_Cur = 3;
        break;
}
m_ctlMainVolumeLevel.SetPos(dwVolumeLevel_Cur);
}
```

Set Volume Level

```
if(pScrollBar == (CScrollBar*)&m_ctlMainVolumeLevel)
{
    dwPos = m_ctlMainVolumeLevel.GetPos();

    SYSTEM_SetVolumeLevel(dwPos);
}
```

4 References

This chapter provides references of the modules included in M3 products. APIs are described using C/C++. Applications are written in VS2005.

Below table describes required files and supported M3 products.

Module	Required Files	Supported brand
SCANNER	Scanner.h Scanner.lib	All brands with 1D scanner
IMAGER	Imager.h Imager.lib	M3 MT, M3 ST10 WM, M3 OX10, M3 UL10, M3 BK10 * Please see IMAGER tutorial for restrictions
CAMERA CE	Cam.h Cam.lib	All M3 CE units * Please see CAMERA (CE) tutorial for more details
CAMERA WM	Cam.h Cam.lib	All M3 WM units * Please see CAMERA (WM) tutorial for more details
RFID (LF / HF)	RFID.h RFID.lib	M3 OX10 (HF)
UHF Gun Reader	RFID_UHF.h RFID_UHF.lib	M3 OX10
WLAN	WLAN.h WLAN.lib	All brands with SUMMIT WLAN
BLUETOOTH	BLUETOOTH.h BLUETOOTH.lib	All brands with BT * Please see BLUETOOTH tutorial for more details
SYSTEM	System.h M3System.lib	All brands *Please see SYSTEM SDK tutorial for more details

4.1 SCANNER (1D)

Status Return value

Please refer to the below table for the status value definition.

Status Value Definition	Code	Meaning
TRUE	1	Success
FALSE	0	General Error

Define
#define WM_SCAN_DATA WM_APP + 350
Enum
<pre>typedef enum { DEVICE_M3SKY = 0, DEVICE_M3SKYSAM, DEVICE_MM3, DEVICE_M3ORANGE, DEVICE_M3SMART_CE, DEVICE_M3SMART_WM, DEVICE_M3GREEN, DEVICE_M3T, DEVICE_M3POS, DEVICE_M3ORANGEPLUS, DEVICE_M3ORANGEPLUS_CE } SCAN_DEVICE_TYPE; typedef enum { OPTICON = 0, SYMBOL, INTERMEC, SYMBOL_HW, CINO, ZEBRA_HW_SE965, HONEYWELL_N4313, UNKNOWN, NOTYET } SCAN_ENGINE_TYPE; typedef enum { SOUND_DEFAULT = 0, SOUND_BEEP, SOUND_NONE } SCAN_SOUND; typedef enum { CODE39_STARNDARD = 0,</pre>

```
CODE39_CODE32,  
CODE39_PZN,  
CODE39_TRIOPTIC,  
CODE39_FULLASCII  
} SCAN_CODE39_FORMAT;
```

Structure

```
typedef struct _DECODER{  
    BYTE bUPCA;  
    BYTE bUPCE;  
    BYTE bEAN13;  
    BYTE bEAN8;  
    BYTE bCODE39;  
    BYTE bCODE128;  
    BYTE bCODE93;  
    BYTE bCODE11;  
    BYTE bCODE25;  
    BYTE bCODABAR;  
    BYTE bKOREAPOST;  
    BYTE bMSI;  
    BYTE bGS1;  
    BYTE bTELEPEN;  
}DECODER, *PDECODER;  
  
typedef struct _DECODER_PARAMS{  
    BYTE bWindeScan;  
    BYTE bHighFilter;  
    BYTE bContinueMode;  
    BYTE bXmitAimID;  
    BYTE bVibrate;  
    int  nSound;  
    int  nTimeOut;  
    int  nSecurityLevel;  
}DECODER_PARAMS, *PDECODER_PARAMS;  
  
typedef struct _UPCA_PARAMS{  
    BYTE  bEnable;  
    BYTE  bXNum;  
    BYTE  bXCD;  
    BYTE  bUPCA_AS_EAN13;  
    BYTE  bAddOn;  
}UPCA_PARAMS, *PUPCA_PARAMS;  
  
typedef struct _UPCE_PARAMS{  
    BYTE  bEnable;  
    BYTE  bXNum;  
    BYTE  bXCD;  
    int  nConvert;  
}UPCE_PARAMS, *PUPCE_PARAMS;
```

```
typedef struct _EAN13_PARAMS{
    BYTE    bEnable;
    BYTE    bBOOKLAND;
    BYTE    bXCD;
    BYTE    bAddOn;
}EAN13_PARAMS, *PEAN13_PARAMS;
typedef struct _EAN8_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
    BYTE    bEAN8_AS_EAN13;
}EAN8_PARAMS, *PEAN8_PARAMS;
typedef struct _CODE39_PARAMS{
    BYTE    bEnable;
    int     nFormat;
    BYTE    bCDV;
    BYTE    bXCD;
    int     nMinLen;
    int     nMaxLen;
}CODE39_PARAMS, *PCODE39_PARAMS;
typedef struct _CODE128_PARAMS{
    BYTE    bEnable;
    BYTE    bUCCEAN128;
    TCHAR szFNC1_ASCII[256];
    int     nMinLen;
    int     nMaxLen;
}CODE128_PARAMS, *PCODE128_PARAMS;
typedef struct _CODE93_PARAMS{
    BYTE    bEnable;
    int     nMinLen;
    int     nMaxLen;
}CODE93_PARAMS, *PCODE93_PARAMS;
typedef struct _KOREAPOST_PARAMS{
    BYTE    bEnable;
}KOREAPOST_PARAMS, *PKOREAPOST_PARAMS;
typedef struct _CODE11_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
    int     nCDV;
    int     nMinLen;
    int     nMaxLen;
}CODE11_PARAMS, *PCODE11_PARAMS;
typedef struct _CODE25_PARAMS{
    BYTE    bI2OF5;
    BYTE    bS2OF5;
    BYTE    bITF14;
    BYTE    bMATRIX2OF5;
```

```
    BYTE    bCHINAPOST;
    BYTE    bINDUSTRY;
    BYTE    bIATA;
    BYTE    bCDV;
    BYTE    bXCD;
    int     nMinLen;
    int     nMaxLen;
}CODE25_PARAMS, *PCODE25_PARAMS;

typedef struct _CODABAR_PARAMS{
    BYTE    bEnable;
    BYTE    bXSS;
    int     nMinLen;
    int     nMaxLen;
}CODABAR_PARAMS, *PCODABAR_PARAMS;

typedef struct _MSI_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
    int     nCDV;
    int     nMinLen;
    int     nMaxLen;
}MSI_PARAMS, *PMSI_PARAMS;

typedef struct _GS1_PARAMS{
    BYTE    bGS1;
    BYTE    bGS1LIM;
    BYTE    bGS1EXP;
}GS1_PARAMS, *PGS1_PARAMS;

typedef struct _TELEPEN_PARAMS{
    BYTE    bEnable;
    BYTE    bNumeric;
}TELEPEN_PARAMS, *PTELEPEN_PARAMS;
```

Functions for 1D Scanner

Name	Description
<u>SCAN_Close</u>	Closes an open scanner
<u>SCAN_GetAdaptiveScanning</u>	Get the information of adaptive Scanning in SE 965 engine
<u>SCAN_GetCODABAR</u>	Gets the option of CODABAR Barcode
<u>SCAN_GetCODE11</u>	Gets the option of CODE11 Barcode
<u>SCAN_GetCODE128</u>	Gets the option of CODE128 Barcode
<u>SCAN_GetCODE25</u>	Gets the option of CODE25 Barcode
<u>SCAN_GetCODE39</u>	Gets the option of CODE39 Barcode
<u>SCAN_GetCODE93</u>	Gets the option of CODE93 Barcode
<u>SCAN_GetDeviceType</u>	Gets the type of device
<u>SCAN_GetEAN13</u>	Gets the option of EAN-13 Barcode
<u>SCAN_GetEAN8</u>	Gets the option of EAN-8 Barcode
<u>SCAN_GetEngineType</u>	Gets the type of scanner engine
<u>SCAN_GetGS1</u>	Gets the option of GS1 Barcode
<u>SCAN_GetKOREAPOST</u>	Gets the option of KOREAPOST Barcode
<u>SCAN_GetMSI</u>	Gets the option of MSI Barcode
<u>SCAN_GetOption</u>	Gets the option of Scanner
<u>SCAN_GetScanData</u>	Gets the ScanData which is read by scanner
<u>SCAN_GetSymbology</u>	Gets Enable/Disable of Symbologies
<u>SCAN_GetTELEPEN</u>	Gets the option of TELEPEN Barcode
<u>SCAN_GetUPCA</u>	Gets the option of UPC-A Barcode
<u>SCAN_GetUPCE</u>	Gets the option of UPC-E Barcode
<u>SCAN_GetVersionInfo</u>	Gets the information of scanner engine and dll version
<u>SCAN_Open</u>	Opens a scanner
<u>SCAN_Read</u>	Starts the beaming of scanner
<u>SCAN_ReadCancel</u>	Stops the beaming of scanner
<u>SCAN_SetAdaptiveScanning</u>	Set the adaptive scanning function in SE 965 engine
<u>SCAN_SetCODABAR</u>	Sets the option of CODABAR Barcode
<u>SCAN_SetCODE11</u>	Sets the option of CODE11 Barcode
<u>SCAN_SetCODE128</u>	Sets the option of CODE128 Barcode
<u>SCAN_SetCODE25</u>	Sets the option of CODE25 Barcode

<u>SCAN_SetCODE39</u>	Sets the option of CODE39 Barcode
<u>SCAN_SetCODE93</u>	Sets the option of CODE93 Barcode
<u>SCAN_SetEAN13</u>	Sets the option of EAN-13 Barcode
<u>SCAN_SetEAN8</u>	Sets the option of EAN-8 Barcode
<u>SCAN_SetGS1</u>	Sets the option of GS1 Barcode
<u>SCAN_SethWnd</u>	Sets the windows handle of message
<u>SCAN_SetKOREAPOST</u>	Sets the option of KOREAPOST Barcode
<u>SCAN_SetMSI</u>	Sets the option of MSI Barcode
<u>SCAN_SetOption</u>	Sets the option of Scanner
<u>SCAN_SetSymbology</u>	Sets the Enable/Disable of Symbologies
<u>SCAN_SetSymbologyAll</u>	Enables all of Symbologies
<u>SCAN_SetSymbologyDefault</u>	Initializes all option of scanner
<u>SCAN_SetTELEPEN</u>	Sets the option of TELEPEN Barcode
<u>SCAN_SetUPCA</u>	Sets the option of UPC-A Barcode
<u>SCAN_SetUPCE</u>	Sets the option of UPC-E Barcode

4.1.1 SCAN_Close

Description

Closes an open scanner.

Syntax

```
SCANNER_API BOOL SCAN_Close();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See also

SCAN_Open

For .Net

Namespace : `ScannerNet.Scanner`

Function : `bool Close()`

Example

```
if(SCAN_Close() == FALSE)
{
    ::MessageBox(NULL, L"Error : SCAN_Close()", NULL, MB_TOPMOST);
}
```

4.1.2 SCAN_GetAdaptiveScanning

Description

Gets the information of adaptive Scanning in SE 965 engine.

Syntax

```
SCANNER_API BOOL SCAN_GetAdaptiveScanning();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetAdaptiveScanning

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetAdaptiveScanning ()

Example

```
BOOL bResult = SCAN_GetAdaptiveScanning();
```

4.1.3 SCAN_GetCODABAR

Description

Gets the option of CODABAR Barcode.

Syntax

```
SCANNER_API BOOL SCAN_GetCODABAR(PCODABAR_PARAMS pCodabar);
```

Parameters

pCodabar

Pointer to a CODABAR_PARAMS structure to be filled in with the CODABAR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetCODABAR

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetCODABAR(out CODABAR_PARAMS pCodabar)

Example

```
PCODABAR_PARAMS  pCodabar = new CODABAR_PARAMS();  
if(SCAN_GetCODABAR(pCodabar) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_GetCODABAR()", NULL, MB_TOPMOST);  
m_bEnable = pCodabar->bEnable;  
m_bXSS = pCodabar->bXSS;  
m_nMinLen = pCodabar->nMinLen;  
m_nMaxLen = pCodabar->nMaxLen;  
delete pCodabar;
```

4.1.4 SCAN_GetCODE11

Description

Gets the option of CODE11 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_GetCODE11(PCODE11_PARAMS pCode11);
```

Parameters

pCode11

Pointer to a CODE11_PARAMS structure to be filled in with the CODE11 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetCODE11

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetCODE11(out CODE11_PARAMS pCode11)

Example

```
PCODE11_PARAMS pCode11 = new CODE11_PARAMS();
if(SCAN_GetCODE11(pCode11) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_GetCODE11()", NULL, MB_TOPMOST);
m_bEnable = pCode11->bEnable;
m_bXCD = pCode11->bXCD;
m_nCDV = pCode11->nCDV;
m_nMinLen = pCode11->nMinLen;
m_nMaxLen = pCode11->nMaxLen;
delete pCode11;
```

4.1.5 SCAN_GetCODE128

Description

Gets the option of CODE128 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_GetCODE128(PCODE128_PARAMS pCode128);
```

Parameters

pCode128

Pointer to a CODE128_PARAMS structure to be filled in with the CODE128 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetCODE128

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetCODE128(out CODE128_PARAMS pCode128)

Example

```
PCODE128_PARAMS pCode128 = new CODE128_PARAMS();
if(SCAN_GetCODE128(pCode128) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_GetCODE128()", NULL, MB_TOPMOST);
m_bEnable = pCode128->bEnable;
m_bUccean128 = pCode128->bUCCEAN128;
m_nMinLen = pCode128->nMinLen;
m_nMaxLen = pCode128->nMaxLen;
if((pScanTestDlg->m_EngineType == OPTICON) || (pScanTestDlg->m_EngineType == SYMBOL) ||
(pScanTestDlg->m_EngineType == INTERMEC))
    m_strFNCASCII.Format(L"%s", pCode128->szFNC1_ASCII);
delete pCode128;
```

4.1.6 SCAN_GetCODE25

Description

Gets the option of CODE25 Barcode.

Syntax

SCANNER_API BOOL SCAN_GetCODE25(PCODE25_PARAMS pCode25);

Parameters

pCode25

Pointer to a CODE25_PARAMS structure to be filled in with the CODE25 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetCODE25

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetCODE25(out CODE25_PARAMS pCode25)

Example

```
PCODE25_PARAMS pCode25 = new CODE25_PARAMS();
if(SCAN_GetCODE25(pCode25) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_GetCODE25()", NULL, MB_TOPMOST);
m_bl2of5 = pCode25->bl2OF5;
m_bS2of5 = pCode25->bS2OF5;
m_bITF14 = pCode25->blTF14;
m_bMatrix2of5 = pCode25->bMATRIX2OF5;
m_bChinaPost = pCode25->bCHINAPOST;
m_bIndustry = pCode25->bINDUSTRY;
m_blata = pCode25->blATA;
m_bCDV = pCode25->bCDV;
m_bXCD = pCode25->bXCD;
m_nMinLen = pCode25->nMinLen;
m_nMaxLen = pCode25->nMaxLen;
delete pCode25;
```

4.1.7 SCAN_GetCODE39

Description

Gets the option of CODE39 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_GetCODE39(PCODE39_PARAMS pCode39);
```

Parameters

pCode39

Pointer to a CODE39_PARAMS structure to be filled in with the CODE39 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetCODE39

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetCODE39(out CODE39_PARAMS pCode39)

Example

```
PCODE39_PARAMS pCode39 = new CODE39_PARAMS();
if(SCAN_GetCODE39(pCode39) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_GetCODE39()", NULL, MB_TOPMOST);
m_bEnable = pCode39->bEnable;
m_nFormat = pCode39->nFormat;
m_bCDV = pCode39->bCDV;
m_bXCD = pCode39->bXCD;
m_nMinLen = pCode39->nMinLen;
m_nMaxLen = pCode39->nMaxLen;
delete pCode39;
```

4.1.8 SCAN_GetCODE93

Description

Gets the option of CODE93 Barcode.

Syntax

SCANNER_API BOOL SCAN_GetCODE93(PCODE93_PARAMS pCode93);

Parameters

pCode93

Pointer to a CODE93_PARAMS structure to be filled in with the CODE93 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetCODE39

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetCODE93(out CODE93_PARAMS pCode93)

Example

```
PCODE93_PARAMS pCode93 = new CODE93_PARAMS();  
if(SCAN_GetCODE93(pCode93) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_GetCODE93()", NULL, MB_TOPMOST);  
m_bEnable = pCode93->bEnable;  
m_nMinLen = pCode93->nMinLen;  
m_nMaxLen = pCode93->nMaxLen;  
delete pCode93;
```

4.1.9 SCAN_GetDeviceType

Description

Gets the type of device.

Syntax

```
SCANNER_API SCAN_DEVICE_TYPE SCAN_GetDeviceType();
```

Parameters

None

Return Value

The return value is SCAN_DEVICE_TYPE.

Remarks

None

See Also

SCAN_GetEngineType

For .Net

Namespace : ScannerNet.Scanner

Function : SCAN_DEVICE_TYPE GetDeviceType();

Example

None

4.1.10 SCAN_GetEAN13

Description

Gets the option of EAN-13 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_GetEAN13(PEAN13_PARAMS pEan13);
```

Parameters

pEan13

Pointer to a EAN13_PARAMS structure to be filled in with the EAN-13 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetEAN13

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetEAN13(out EAN13_PARAMS pEan13)

Example

```
PEAN13_PARAMS pEan13 = new EAN13_PARAMS();  
if(SCAN_GetEAN13(pEan13) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_GetEAN13()", NULL, MB_TOPMOST);  
m_bEnable = pEan13->bEnable;  
m_bBookland = pEan13->bBOOKLAND;  
m_bXCD = pEan13->bXCD;  
m_bAddOn = pEan13->bAddOn;  
delete pEan13;
```

4.1.11 SCAN_GetEAN8

Description

Gets the option of EAN-8 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_GetEAN8(PEAN8_PARAMS pEan8);
```

Parameters

pEan8

Pointer to an EAN8_PARAMS structure to be filled in with the EAN-8 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetEAN8

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetEAN8(out EAN8_PARAMS pEan8)

Example

```
PEAN8_PARAMS pEan8 = new EAN8_PARAMS();  
if(SCAN_GetEAN8(pEan8) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_GetEAN8()", NULL, MB_TOPMOST);  
m_bEnable = pEan8->bEnable;  
m_bXCD = pEan8->bXCD;  
m_bEan8AsEan13 = pEan8->bEAN8_AS_EAN13;  
delete pEan8;
```

4.1.12 SCAN_GetEngineType

Description

Gets the type of scanner engine.

Syntax

```
SCANNER_API SCAN_ENGINE_TYPE SCAN_GetEngineType();
```

Parameters

None

Return Value

The return value is SCAN_ENGINE_TYPE.

Remarks

None

See Also

SCAN_GetDeviceType

For .Net

Namespace : ScannerNet.Scanner

Function : SCAN_ENGINE_TYPE GetEngineType();

Example

None

4.1.13 SCAN_GetGS1

Description

Gets the option of GS1 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_GetGS1(PGS1_PARAMS pGs1);
```

Parameters

pGs1

Pointer to a GS1_PARAMS structure to be filled in with the GS1 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetGS1

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetGS1(out GS1_PARAMS pGs1)

Example

```
PGS1_PARAMS    pGs1 = new GS1_PARAMS();  
if(SCAN_GetGS1(pGs1) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_GetGS1()", NULL, MB_TOPMOST);  
m_bEnable = pGs1->bGS1;  
m_bGs1Lim = pGs1->bGS1LIM;  
m_bGs1Exp = pGs1->bGS1EXP;  
delete pGs1;
```

4.1.14 SCAN_GetKOREAPOST

Description

Gets the option of KOREAPOST Barcode.

Syntax

```
SCANNER_API BOOL SCAN_GetKOREAPOST(PKOREAPOST_PARAMS pKoreaPost);
```

Parameters

pKoreaPost

Pointer to a KOREAPOST_PARAMS structure to be filled in with the KOREAPOST common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetKOREAPOST

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetKOREAPOST(out KOREAPOST_PARAMS pKoreaPost)

Example

```
PKOREAPOST_PARAMS pKoreaPost = new KOREAPOST_PARAMS();  
if(SCAN_GetKOREAPOST(pKoreaPost) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_GetKOREAPOST()", NULL, MB_TOPMOST);  
m_bEnable = pKoreaPost->bEnable;  
delete pKoreaPost;
```

4.1.15 SCAN_GetMSI

Description

Gets the option of MSI Barcode.

Syntax

```
SCANNER_API BOOL SCAN_GetMSI(PMSI_PARAMS pMsi);
```

Parameters

pMsi

Pointer to a MSI_PARAMS structure to be filled in with the MSI common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetMSI

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetMSI(out MSI_PARAMS pMsi)

Example

```

PMSI_PARAMS pMsi = new MSI_PARAMS();
if(SCAN_GetMSI(pMsi) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_GetMSI()", NULL, MB_TOPMOST);
m_bEnable = pMsi->bEnable;
m_bXCD = pMsi->bXCD;
m_nCDV = pMsi->nCDV;
m_nMinLen = pMsi->nMinLen;
m_nMaxLen = pMsi->nMaxLen;
delete pMsi;

```

4.1.16 SCAN_GetOption

Description

Gets the option of Scanner.

Syntax

```
SCANNER_API BOOL SCAN_GetOption(PDECODER_PARAMS pOption);
```

Parameters

pOption

Pointer to a DECODER_PARAMS structure to be filled in with the scanner parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetOption

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetOption(out DECODER_PARAMS pOption)

Example

```

PDECODER_PARAMS pOption = new DECODER_PARAMS();
SCAN_GetOption(pOption);
m_nSyncMode = pScanTestDlg->m_bSyncMode;
m_nSound = pOption->nSound;
m_bVibrate = pOption->bVibrate;
m_bAimID = pOption->bXmitAimID;

```

```
m_bContinueMode = pOption->bContinueMode;
m_bWideScan = pOption->bWideScan;
m_bHighFilter = pOption->bHighFilter;
m_ctrlComboTimeOut.SetCurSel(pOption->nTimeOut - 1);
m_ctrlComboSecurityLevel.SetCurSel(pOption->nSecurityLevel - 1);
delete pOption;
```

4.1.17 SCAN_GetScanData

Description

Gets the ScanData which is read by scanner.

Syntax

```
SCANNER_API BOOL SCAN_GetScanData(TCHAR *pszBarType, TCHAR *pszBarData);
```

Parameters

pszBarType

Pointer to barcode type.

pszBarData

Pointer to barcode data

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetScanData(StringBuilder szBarType, StringBuilder szBarData)

Example

```
TCHAR  szBarType[1024] = {0, };
TCHAR  szBarData[1024] = {0, };
SCAN_GetScanData(szBarType, szBarData);
```

4.1.18 SCAN_GetSymbology

Description

Gets Enable/Disable of Symbologies.

Syntax

```
SCANNER_API BOOL SCAN_GetSymbology(PDECODER pSymbology);
```

Parameters

pSymbology

Pointer to a DECODER structure to be filled in with the symbologies enable or disable.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetSymbology

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetSymbology(out DECODER pSymbology)

Example

```
PDECODER pDecoder = new DECODER();
SCAN_GetSymbology(pDecoder);
m_bUpca = pDecoder->bUPCA;
m_bUpce = pDecoder->bUPCE;
m_bEan13 = pDecoder->bEAN13;
m_bEan8 = pDecoder->bEAN8;
m_bCode39 = pDecoder->bCODE39;
m_bCode128 = pDecoder->bCODE128;
m_bCode93 = pDecoder->bCODE93;
m_bCode11 = pDecoder->bCODE11;
m_bCode25 = pDecoder->bCODE25;
m_bCodabar = pDecoder->bCODABAR;
m_bKoreaPost = pDecoder->bKOREAPOST;
m_bMsi = pDecoder->bMSI;
m_bGs1 = pDecoder->bGS1;
m_bTelepen = pDecoder->bTELEPEN;
delete pDecoder;
```

4.1.19 SCAN_GetTELEPEN

Description

Gets the option of TELEPEN Barcode.

Syntax

```
SCANNER_API BOOL SCAN_GetTELEPEN(PTELEPEN_PARAMS pTelepen);
```

Parameters

pTelepen

Pointer to a TELEPEN_PARAMS structure to be filled in with the TELEPEN common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetTELEPEN

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetTELEPEN(out TELEPEN_PARAMS pTelepen)

Example

```
PTELEPEN_PARAMS pTelepen = new TELEPEN_PARAMS();
if(SCAN_GetTELEPEN(pTelepen) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_GetTELEPEN()", NULL, MB_TOPMOST);
m_bEnable = pTelepen->bEnable;
m_bNumeric = pTelepen->bNumeric;
delete pTelepen;
```

4.1.20 SCAN_GetUPCA

Description

Gets the option of UPC-A Barcode.

Syntax

```
SCANNER_API BOOL SCAN_GetUPCA(PUPCA_PARAMS pUpca);
```

Parameters

pUpca

Pointer to a UPCA_PARAMS structure to be filled in with the UPC-A common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetUPCA

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetUPCA(out UPCA_PARAMS pUpca)

Example

```
PUPCA_PARAMS pUpca = new UPCA_PARAMS();
if(SCAN_GetUPCA(pUpca) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_GetUPCA()", NULL, MB_TOPMOST);
m_bEnable = pUpca->bEnable;
m_bXNum = pUpca->bXNum;
m_bXCD = pUpca->bXCD;
m_bUpcaAsEan13 = pUpca->bUPCA_AS_EAN13;
m_bAddOn = pUpca->bAddOn;
delete pUpca;
```

4.1.21 SCAN_GetUPCE

Description

Gets the option of UPC-E Barcode.

Syntax

SCANNER_API BOOL SCAN_GetUPCE(PUPCE_PARAMS pUpce);

Parameters

pUpce

Pointer to a UPCE_PARAMS structure to be filled in with the UPC-E common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetUPCE

For .Net

Namespace : ScannerNet.Scanner

Function : bool GetUPCE(out UPCE_PARAMS pUpce)

Example

```
PUPCE_PARAMS pUpce = new UPCE_PARAMS();  
if(SCAN_GetUPCE(pUpce) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_GetUPCE()", NULL, MB_TOPMOST);  
m_bEnable = pUpce->bEnable;  
m_bXNum = pUpce->bXNum;  
m_bXCD = pUpce->bXCD;  
m_nConvert = pUpce->nConvert;  
delete pUpce;
```

4.1.22 SCAN_GetVersionInfo

Description

Gets the information of scanner engine and dll version.

Syntax

```
SCANNER_API BOOL SCAN_GetVersionInfo(TCHAR* pszVersion);
```

Parameters

pszVersion

Pointer to a TCHAR to be filled in with the version info.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : ScannerNet.Scanner

Function : string GetVersionInfo()

Example

```
TCHAR szVersionInfo[1024] = {0, };  
SCAN_GetVersionInfo(szVersionInfo);
```

4.1.23 SCAN_Open

Description

Opens a scanner.

Syntax

```
SCANNER_API BOOL SCAN_Open();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_Close

For .Net

Namespace : ScannerNet.Scanner

Function : bool Open()

Example

```
if(SCAN_Open() == FALSE)
{
    ::MessageBox(NULL, L"Error : SCAN_Open()", NULL, MB_TOPMOST);
}
```

4.1.24 SCAN_Read

Description

Starts the beaming of scanner.

Syntax

```
SCANNER_API BOOL SCAN_Read();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_ReadCancel

For .Net

Namespace : ScannerNet.Scanner

Function : bool Read()

Example

None

4.1.25 SCAN_ReadCancel

Description

Stops the beaming of scanner.

Syntax

SCANNER_API BOOL SCAN_ReadCancel();

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_Read

For .Net

Namespace : ScannerNet.Scanner

Function : bool ReadCancel()

Example

None

4.1.26 SCAN_SetAdaptiveScanning

Description

Set the adaptive scanning function in SE 965 engine.

Syntax

SCANNER_API BOOL SCAN_SetAdaptiveScanning (bool bEnable);

Parameters

bEnable

Whether to enable adaptive scanning function or not.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetAdaptiveScanning

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetAdaptiveScanning (bool bEnable)

Example

```
SCAN_SetAdaptiveScanning(true);
```

4.1.27 SCAN_SetCODABAR

Description

Sets the option of CODABAR Barcode.

Syntax

```
SCANNER_API BOOL SCAN_SetCODABAR(PCODABAR_PARAMS pCodabar);
```

Parameters

pCodabar

Pointer to a CODABAR_PARAMS structure holding the CODABAR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetCODABAR

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetCODABAR(ref CODABAR_PARAMS pCodabar)

Example

```
PCODABAR_PARAMS pCodabar = new CODABAR_PARAMS();
```

```
pCodabar->bEnable = m_bEnable;
```

```
pCodabar->bXSS = m_bXSS;
```

```
pCodabar->nMinLen = m_nMinLen;
```

```
pCodabar->nMaxLen = m_nMaxLen;
if(SCAN_SetCODABAR(pCodabar) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetCODABAR()", NULL, MB_TOPMOST);
delete pCodabar;
```

4.1.28 SCAN_SetCODE11

Description

Sets the option of CODE11 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_SetCODE11(PCODE11_PARAMS pCode11);
```

Parameters

pCode11

Pointer to a CODE11_PARAMS structure holding the CODE11 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetCODE11

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetCODE11(ref CODE11_PARAMS pCode11)

Example

```
PCODE11_PARAMS pCode11 = new CODE11_PARAMS();
pCode11->bEnable = m_bEnable;
pCode11->bXCD = m_bXCD;
pCode11->nCDV = m_nCDV;
pCode11->nMinLen = m_nMinLen;
pCode11->nMaxLen = m_nMaxLen;
if(SCAN_SetCODE11(pCode11) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetCODE11()", NULL, MB_TOPMOST);
delete pCode11;
```

4.1.29 SCAN_SetCODE128

Description

Sets the option of CODE128 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_SetCODE128(PCODE128_PARAMS pCode128);
```

Parameters

pCode128

Pointer to a CODE128_PARAMS structure holding the CODE128 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetCODE128

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetCODE128(ref CODE128_PARAMS pCode128)

Example

```
PCODE128_PARAMS pCode128 = new CODE128_PARAMS();
pCode128->bEnable = m_bEnable;
pCode128->bUCCEAN128 = m_bUccean128;
pCode128->nMinLen = m_nMinLen;
pCode128->nMaxLen = m_nMaxLen;
wsprintf(pCode128->szFNC1_ASCII, L"%s", m_strFNCASCII);
if(SCAN_SetCODE128(pCode128) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetCODE128()", NULL, MB_TOPMOST);
delete pCode128;
```

4.1.30 SCAN_SetCODE25

Description

Sets the option of CODE25 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_SetCODE25(PCODE25_PARAMS pCode25);
```

Parameters

pCode25

Pointer to a CODE25_PARAMS structure holding the CODE25 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetCODE25

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetCODE25(ref CODE25_PARAMS pCode25)

Example

```
PCODE25_PARAMS pCode25 = new CODE25_PARAMS();  
pCode25->bl2OF5 = m_bl2of5;  
pCode25->bS2OF5 = m_bS2of5;  
pCode25->blTF14 = m_blTF14;  
pCode25->bMATRIX2OF5 = m_bMatrix2of5;  
pCode25->bCHINAPOST = m_bChinaPost;  
pCode25->bINDUSTRY = m_bIndustry;  
pCode25->blATA = m_blata;  
pCode25->bCDV = m_bCDV;  
pCode25->bXCD = m_bXCD;  
pCode25->nMinLen = m_nMinLen;  
pCode25->nMaxLen = m_nMaxLen;  
if(SCAN_SetCODE25(pCode25) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_SetCODE25()", NULL, MB_TOPMOST);  
delete pCode25;
```

4.1.31 SCAN_SetCODE39

Description

Sets the option of CODE39 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_SetCODE39(PCODE39_PARAMS pCode39);
```

Parameters

pCode39

Pointer to a CODE39_PARAMS structure holding the CODE39 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetCODE39

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetCODE39(ref CODE39_PARAMS pCode39)

Example

```
PCODE39_PARAMS pCode39 = new CODE39_PARAMS();  
if(SCAN_GetCODE39(pCode39) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_GetCODE39()", NULL, MB_TOPMOST);  
m_bEnable = pCode39->bEnable;  
m_nFormat = pCode39->nFormat;  
m_bCDV = pCode39->bCDV;  
m_bXCD = pCode39->bXCD;  
m_nMinLen = pCode39->nMinLen;  
m_nMaxLen = pCode39->nMaxLen;  
delete pCode39;
```

4.1.32 SCAN_SetCODE93

Description

Sets the option of CODE93 Barcode.

Syntax

SCANNER_API BOOL SCAN_SetCODE93(PCODE93_PARAMS pCode93);

Parameters

pCode93

Pointer to a CODE93_PARAMS structure holding the CODE93 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetCODE39

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetCODE93(ref CODE93_PARAMS pCode93)

Example

```
PCODE93_PARAMS pCode93 = new CODE93_PARAMS();
pCode93->bEnable = m_bEnable;
pCode93->nMinLen = m_nMinLen;
pCode93->nMaxLen = m_nMaxLen;
if(SCAN_SetCODE93(pCode93) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetCODE93()", NULL, MB_TOPMOST);
delete pCode93;
```

4.1.33 SCAN_SetEAN13

Description

Sets the option of EAN-13 Barcode.

Syntax

SCANNER_API BOOL SCAN_SetEAN13(PEAN13_PARAMS pEan13);

Parameters

pEan13

Pointer to an EAN13_PARAMS structure holding the EAN-13 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetEAN13

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetEAN13(ref EAN13_PARAMS pEan13)

Example

```
PEAN13_PARAMS pEan13 = new EAN13_PARAMS();
```

```

pEan13->bEnable = m_bEnable;
pEan13->bBOOKLAND = m_bBookland;
pEan13->bXCD = m_bXCD;
pEan13->bAddOn = m_bAddOn;
if(SCAN_SetEAN13(pEan13) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetEAN13()", NULL, MB_TOPMOST);
delete pEan13;

```

4.1.34 SCAN_SetEAN8

Description

Sets the option of EAN-8 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_SetEAN8(PEAN8_PARAMS pEan8);
```

Parameters

pEan8

Pointer to an EAN8_PARAMS structure holding the EAN9 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetEAN8

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetEAN8(ref EAN8_PARAMS pEan8)

Example

```

PEAN8_PARAMS pEan8 = new EAN8_PARAMS();
pEan8->bEnable = m_bEnable;
pEan8->bXCD = m_bXCD;
pEan8->bEAN8_AS_EAN13 = m_bEan8AsEan13;
if(SCAN_SetEAN8(pEan8) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetEAN8()", NULL, MB_TOPMOST);
delete pEan8;

```

4.1.35 SCAN_SetGS1

Description

Sets the option of GS1 Barcode.

Syntax

```
SCANNER_API BOOL SCAN_SetGS1(PGS1_PARAMS pGs1);
```

Parameters

pGs1

Pointer to a GS1_PARAMS structure holding the GS1 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetGS1

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetGS1(ref GS1_PARAMS pGs1)

Example

```
PGS1_PARAMS    pGs1 = new GS1_PARAMS();
pGs1->bGS1 = m_bEnable;
pGs1->bGS1LIM = m_bGs1Lim;
pGs1->bGS1EXP = m_bGs1Exp;
if(SCAN_SetGS1(pGs1) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetGS1()", NULL, MB_TOPMOST);
delete pGs1;
```

4.1.36 SCAN_SethWnd

Description

Sets the windows handle of message

Syntax

```
SCANNER_API BOOL SCAN_SethWnd(HWND hMainWnd)
```

Parameters

hMainWnd

A handle to the window whose window procedure will receive the message

Return Value

TRUE indicates success. FALSE indicates failure.

Remarks

None

Example

```
SCAN_SetHwnd(this->GetSafeHwnd());
```

4.1.37 SCAN_SetKOREAPOST

Description

Sets the option of KOREAPOST Barcode.

Syntax

```
SCANNER_API BOOL SCAN_SetKOREAPOST(PKOREAPOST_PARAMS pKoreaPost);
```

Parameters

pKoreaPost

Pointer to a KOREAPOST_PARAMS structure holding the KOREAPOST common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetKOREAPOST

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetKOREAPOST(ref KOREAPOST_PARAMS pKoreaPost)

Example

```
PKOREAPOST_PARAMS pKoreaPost = new KOREAPOST_PARAMS();  
pKoreaPost->bEnable = m_bEnable;  
if(SCAN_SetKOREAPOST(pKoreaPost) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_SetKOREAPOST()", NULL, MB_TOPMOST);  
delete pKoreaPost;
```

4.1.38 SCAN_SetMSI

Description

Sets the option of MSI Barcode.

Syntax

```
SCANNER_API BOOL SCAN_SetMSI(PMSI_PARAMS pMsi);
```

Parameters

pMsi

Pointer to a MSI_PARAMS structure holding the MSI common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetMSI

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetMSI(ref MSI_PARAMS pMsi)

Example

```
PMSI_PARAMS pMsi = new MSI_PARAMS();  
pMsi->bEnable = m_bEnable;  
pMsi->bXCD = m_bXCD;  
pMsi->nCDV = m_nCDV;  
pMsi->nMinLen = m_nMinLen;  
pMsi->nMaxLen = m_nMaxLen;  
if(SCAN_SetMSI(pMsi) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_SetMSI()", NULL, MB_TOPMOST);  
delete pMsi;
```

4.1.39 SCAN_SetOption

Description

Sets the option of Scanner.

Syntax

```
SCANNER_API BOOL SCAN_SetOption(PDECODER_PARAMS pOption);
```

Parameters

pOption

Pointer to a DECODER_PARAMS structure holding scanner parameters.\

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetOption

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetOption(ref DECODER_PARAMS pOption)

Example

```
PDECODER_PARAMS pOption = new DECODER_PARAMS();
pOption->nSound = m_nSound;
pOption->bVibrate = m_bVibrate;
pOption->bXmitAimID = m_bAimID;
pOption->bContinueMode = m_bContinueMode;
pOption->bWindeScan = m_bWideScan;
pOption->bHighFilter = m_bHighFilter;
pOption->nTimeOut = m_ctrlComboTimeOut.GetCurSel() + 1;
pOption->nSecurityLevel = m_ctrlComboSecurityLevel.GetCurSel() + 1;
SCAN_SetOption(pOption);
delete pOption;
```

4.1.40 SCAN_SetSymbology

Description

Sets Enable/Disable of Symbologies.

Syntax

```
SCANNER_API BOOL SCAN_SetSymbology(PDECODER pSymbology);
```

Parameters

pSymbology

Pointer to a DECODER structure holding the symbologies enable or disable.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetSymbology

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetSymbology(ref DECODER pSymbology)

Example

```
PDECODER pDecoder = new DECODER();  
pDecoder->bUPCA = m_bUpca;  
pDecoder->bUPCE = m_bUpce;  
pDecoder->bEAN13 = m_bEan13;  
pDecoder->bEAN8 = m_bEan8;  
pDecoder->bCODE39 = m_bCode39;  
pDecoder->bCODE128 = m_bCode128;  
pDecoder->bCODE93 = m_bCode93;  
pDecoder->bCODE11 = m_bCode11;  
pDecoder->bCODE25 = m_bCode25;  
pDecoder->bCODABAR = m_bCodabar;  
pDecoder->bKOREAPOST = m_bKoreaPost;  
pDecoder->bMSI = m_bMsi;  
pDecoder->bGS1 = m_bGs1;  
pDecoder->bTELEPEN = m_bTelepen;  
SCAN_SetSymbology(pDecoder);  
delete pDecoder;
```

4.1.41 SCAN_SetSymbologyAll

Description

Enables all of Symbologies.

Syntax

```
SCANNER_API BOOL SCAN_SetSymbologyAll();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetSymbologyDefault

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetSymbologyAll();

Example

None

4.1.42 SCAN_SetSymbologyDefault

Description

Initializes all option of scanner.

Syntax

```
SCANNER_API BOOL SCAN_SetSymbologyDefault();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_SetSymbologyAll

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetSymbologyDefault()

Example

None

4.1.43 SCAN_SetTELEPEN

Description

Sets the option of TELEPEN Barcode.

Syntax

```
SCANNER_API BOOL SCAN_SetTELEPEN(PTELEPEN_PARAMS pTelepen);
```

Parameters

pTelepen

Pointer to a TELEPEN_PARAMS structure holding the TELEPEN common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetTELEPEN

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetTELEPEN(ref TELEPEN_PARAMS pTelepen)

Example

```
PTELEPEN_PARAMS pTelepen = new TELEPEN_PARAMS();
pTelepen->bEnable = m_bEnable;
pTelepen->bNumeric = m_bNumeric;
if(SCAN_SetTELEPEN(pTelepen) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetTELEPEN()", NULL, MB_TOPMOST);
delete pTelepen;
```

4.1.44 SCAN_SetUPCA

Description

Sets the option of UPC-A Barcode.

Syntax

SCANNER_API BOOL SCAN_SetUPCA(PUPCA_PARAMS pUpca);

Parameters

pUpca

Pointer to a UPCA_PARAMS structure holding the UPC-A common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetUPCA

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetUPCA(ref UPCA_PARAMS pUpca)

Example

```
PUPCA_PARAMS pUpca = new UPCA_PARAMS();  
pUpca->bEnable = m_bEnable;  
pUpca->bXNum = m_bXNum;  
pUpca->bXCD = m_bXCD;  
pUpca->bUPCA_AS_EAN13 = m_bUpcaAsEan13;  
pUpca->bAddOn = m_bAddOn;  
if(SCAN_SetUPCA(pUpca) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_SetUPCA()", NULL, MB_TOPMOST);  
delete pUpca;
```

4.1.45 SCAN_SetUPCE

Description

Sets the option of UPC-E Barcode.

Syntax

```
SCANNER_API BOOL SCAN_SetUPCE(PUPCE_PARAMS pUpce);
```

Parameters

pUpce

Pointer to a UPCE_PARAMS structure holding the UPC-E common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SCAN_GetUPCE

For .Net

Namespace : ScannerNet.Scanner

Function : bool SetUPCE(ref UPCE_PARAMS pUpce)

Example

```
PUPCE_PARAMS pUpce = new UPCE_PARAMS();  
pUpce->bEnable = m_bEnable;  
pUpce->bXNum = m_bXNum;  
pUpce->bXCD = m_bXCD;
```

```
pUpce->nConvert = m_nConvert;  
if(SCAN_SetUPCE(pUpce) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_SetUPCE()", NULL, MB_TOPMOST);  
delete pUpce;
```

4.2 IMAGER (2D)

Status Return value

Please refer to the below table for the status value definition.

Status Value Definition	Code	Meaning
TRUE	1	Success
FALSE	0	General Error

Define
<pre>#define WM_SCAN_DATA WM_APP + 350 #define WM_IQ_DATA WM_APP + 380</pre>
Enum
<pre>typedef enum { DEVICE_M3SKY = 0, DEVICE_M3SKYSAM, DEVICE_MM3, DEVICE_M3ORANGE, DEVICE_M3SMART_CE, DEVICE_M3SMART_WM, DEVICE_M3GREEN, DEVICE_M3T, DEVICE_M3POS, DEVICE_M3ORANGEPLUS, DEVICE_M3ORANGEPLUS_CE } SCAN_DEVICE_TYPE; typedef enum { SOUND_DEFAULT = 0, SOUND_BEEP, SOUND_NONE } SCAN_SOUND; typedef enum { CODE39_STANDARD = 0, CODE39_CODE32, CODE39_TRIOPTIC }CODE39_FORMAT; typedef enum { MODE_OCR_DISABLED = 0, MODE_OCR_A, MODE_OCR_B, MODE_OCR_MONEY,</pre>

```

    MODE_OCR_MICR_UNSUPPORTED,
}OCR_MODE;

typedef enum {
    DECODE_STANDARD = 0,
    DECODE_QUICK_OMNI,
    DECODE_LINEAR_PRIORITY
} DECODEMODE;

typedef enum {
    CAM_640X480 = 0,
    CAM_320X240,
    CAM_160X120
} CAM_RESOLUTION;

typedef enum {
    CAM_JPG = 0,
    CAM_BMP,
} CAM_FORMAT;

typedef enum {
    SAVE_DATE = 0,
    SAVE_CUSTOM_WITH_NUM,
    SAVE_CUSTOM
} SAVE_MODE;

typedef enum {
    IQ_AZTEC = 0,
    IQ_CODE39
} IQ_TYPE;

```

Structure

```

typedef struct _DECODER{
    BYTE bAZTEC;
    BYTE bCODABAR;
    BYTE bCODE11;
    BYTE bCODE128;
    BYTE bCODE39;
    BYTE bCODE49;
    BYTE bCODE93;
    BYTE bCOMPOSITE;
    BYTE bDATAMATRIX;
    BYTE bEAN8;
    BYTE bEAN13;
    BYTE bINT25;
    BYTE bMAXICODE;
    BYTE bMICROPDF;
    BYTE bOCR;
    BYTE bPDF417;
    BYTE bPOSTNET;
    BYTE bQR;
}

```

```

BYTE bRSS;
BYTE bUPCA;
BYTE bUPCE;
BYTE bISBT;
BYTE bBPO;
BYTE bCANPOST;
BYTE bAUSPOST;
BYTE bIATA25;
BYTE bCODABLOCK;
BYTE bJAPOST;
BYTE bPLANET;
BYTE bDUTCHPOST;
BYTE bMSI;
BYTE bTLCODE39;
BYTE bSTRT25;
BYTE bMATRIX25;
BYTE bPLESSEY;
BYTE bCHINAPOST;
BYTE bKOREAPOST;
BYTE bTELEPEN;
BYTE bCODE16K;
BYTE bPOSICODE;
BYTE bCOUPONCODE;
BYTE bUSPS4CB;
BYTE bIDTAG;
BYTE bLABEL;
BYTE bGS1_128;
BYTE bHANXIN;
BYTE bGRIDMATRIX;
}DECODER, *PDECODER;
typedef struct _DECODER_PARAMS{
    BYTE bCentering;
    BYTE bContinueMode;
    BYTE bXmitAimID;
    BYTE bHexMode;
    BYTE bVibrate;
    int  nSound;
    int  nTimeOut;
    int  nLightMode;
}DECODER_PARAMS, *PDECODER_PARAMS;
typedef struct _DECOPTION_PARAMS{
    int  nDecodeMode;
    int  nLinearRange;
    int  nPrintWeight;
    int  nMaxDecode;
    int  nMaxSearch;

```

```

    BOOL bVideoReverse;
}DECOPTION_PARAMS, *PDECOPTION_PARAMS;

typedef struct _CAM_PARAMS{
    int      nSaveMode;
    int      nSaveFormat;
    int      nJpegQuality;
    int      nResolution;
    TCHAR    szSaveFolder[256];
    TCHAR    szFileName[256];
} CAM_PARAMS, *PCAM_PARAMS;

typedef struct _IQ_PARAMS{
    int      nSaveMode;
    int      nIQType;
    TCHAR    szSaveFolder[256];
    TCHAR    szFileName[256];
} IQ_PARAMS, *PIQ_PARAMS;

typedef struct _AZTEC_PARAMS{
    BYTE     bEnable;
    int      nMinLen;
    int      nMaxLen;
}AZTEC_PARAMS, *PAZTEC_PARAMS;

typedef struct _CODABAR_PARAMS{
    BYTE     bEnable;
    BYTE     bCDV;
    BYTE     bXCD;
    BYTE     bXSS;
    int      nMinLen;
    int      nMaxLen;
}CODABAR_PARAMS, *PCODABAR_PARAMS;

typedef struct _CODE11_PARAMS{
    BYTE     bEnable;
    BYTE     bCDV;
    int      nMinLen;
    int      nMaxLen;
}CODE11_PARAMS, *PCODE11_PARAMS;

typedef struct _CODE128_PARAMS{
    BYTE     bEnable;
    int      nMinLen;
    int      nMaxLen;
}CODE128_PARAMS, *PCODE128_PARAMS;

typedef struct _CODE39_PARAMS{
    BYTE     bEnable;
    Int      nFormat;
    BYTE     bCDV;
    BYTE     bXCD;
    BYTE     bFullASCII;

```

```

    int    nMinLen;
    int    nMaxLen;
}CODE39_PARAMS, *PCODE39_PARAMS;

typedef struct _CODE49_PARAMS{
    BYTE    bEnable;
    int     nMinLen;
    int     nMaxLen;
}CODE49_PARAMS, *PCODE49_PARAMS;

typedef struct _CODE93_PARAMS{
    BYTE    bEnable;
    int     nMinLen;
    int     nMaxLen;
}CODE93_PARAMS, *PCODE93_PARAMS;

typedef struct _COMPOSITE_PARAMS{
    BYTE    bEnable;
    BYTE    bComposite_Upc;
    int     nMinLen;
    int     nMaxLen;
}COMPOSITE_PARAMS, *PCOMPOSITE_PARAMS;

typedef struct _DATAMATRIX_PARAMS{
    BYTE    bEnable;
    int     nMinLen;
    int     nMaxLen;
}DATAMATRIX_PARAMS, *PDATAMATRIX_PARAMS;

typedef struct _EAN8_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
    BYTE    bAddOn;
}EAN8_PARAMS, *PEAN8_PARAMS;

typedef struct _EAN13_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
    BYTE    bAddOn;
}EAN13_PARAMS, *PEAN13_PARAMS;

typedef struct _INT25_PARAMS{
    BYTE    bEnable;
    BYTE    bCDV;
    BYTE    bXCD;
    int     nMinLen;
    int     nMaxLen;
}INT25_PARAMS, *PINT25_PARAMS;

typedef struct _MAXICODE_PARAMS{
    BYTE    bEnable;
    int     nMinLen;
    int     nMaxLen;
}MAXICODE_PARAMS, *PMAXICODE_PARAMS;

```



```

typedef struct _MICROPDF_PARAMS{
    BYTE    bEnable;
    int     nMinLen;
    int     nMaxLen;
}MICROPDF_PARAMS, *PMICROPDF_PARAMS;

typedef struct _OCR_PARAMS{
    BYTE     bEnable;
    OCR_MODE nMode;
    TCHAR    szTemplate[256];
    TCHAR    szGroupG[256];
    TCHAR    szGroupH[256];
    TCHAR    szCheckChar[64];
}OCR_PARAMS, *POCR_PARAMS;

typedef struct _PDF417_PARAMS{
    BYTE    bEnable;
    int     nMinLen;
    int     nMaxLen;
}PDF417_PARAMS, *PPDF417_PARAMS;

typedef struct _POSTNET_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
}POSTNET_PARAMS, *PPOSTNET_PARAMS;

typedef struct _QR_PARAMS{
    BYTE    bEnable;
    int     nMinLen;
    int     nMaxLen;
}QR_PARAMS, *PQR_PARAMS;

typedef struct _RSS_PARAMS{
    BYTE    bEnable;
    BYTE    bRssLim;
    BYTE    bRssExp;
    int     nMinLen;
    int     nMaxLen;
}RSS_PARAMS, *PRSS_PARAMS;

typedef struct _UPCA_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
    BYTE    bXNum;
    BYTE    bAddOn;
}UPCA_PARAMS, *PUPCA_PARAMS;

typedef struct _UPCE_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
    BYTE    bXNum;
    BYTE    bAddOn;
}UPCE_PARAMS, *PUPCE_PARAMS;

```

```
typedef struct _IATA25_PARAMS{
    BYTE    bEnable;
    int      nMinLen;
    int      nMaxLen;
}IATA25_PARAMS, *PIATA25_PARAMS;
typedef struct _CODABLOCK_PARAMS{
    BYTE    bEnable;
    int      nMinLen;
    int      nMaxLen;
}CODABLOCK_PARAMS, *PCODABLOCK_PARAMS;
typedef struct _PLANET_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
}PLANET_PARAMS, *PPLANET_PARAMS;
typedef struct _MSI_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
    int      nMinLen;
    int      nMaxLen;
}MSI_PARAMS, *PMSI_PARAMS;
typedef struct _STRT25_PARAMS{
    BYTE    bEnable;
    int      nMinLen;
    int      nMaxLen;
}STRT25_PARAMS, *PSTRT25_PARAMS;
typedef struct _MATRIX25_PARAMS{
    BYTE    bEnable;
    int      nMinLen;
    int      nMaxLen;
}MATRIX25_PARAMS, *PMATRIX25_PARAMS;
typedef struct _PLESSEY_PARAMS{
    BYTE    bEnable;
    int      nMinLen;
    int      nMaxLen;
}PLESSEY_PARAMS, *PPLESSEY_PARAMS;
typedef struct _CHINAPOST_PARAMS{
    BYTE    bEnable;
    int      nMinLen;
    int      nMaxLen;
}CHINAPOST_PARAMS, *PCHINAPOST_PARAMS;
typedef struct _KOREAPOST_PARAMS{
    BYTE    bEnable;
    int      nMinLen;
    int      nMaxLen;
}KOREAPOST_PARAMS, *PKOREAPOST_PARAMS;
typedef struct _TELEPEN_PARAMS{
```

```
    BYTE    bEnable;  
    BYTE    bNumeric;  
    int     nMinLen;  
    int     nMaxLen;  
}TELEPEN_PARAMS, *PTELEPEN_PARAMS;  
  
typedef struct _CODE16K_PARAMS{  
    BYTE    bEnable;  
    int     nMinLen;  
    int     nMaxLen;  
}CODE16K_PARAMS, *PCODE16K_PARAMS;  
  
typedef struct _POSICODE_PARAMS{  
    BYTE    bEnable;  
    BYTE    bPosi_Lim1;  
    BYTE    bPosi_Lim2;  
    int     nMinLen;  
    int     nMaxLen;  
}POSICODE_PARAMS, *PPOSICODE_PARAMS;
```

Functions for 2D Imager

Name	Description
<u>IMAGER_CAMCapture</u>	Captures the preview of imaging
<u>IMAGER_CAMGetOption</u>	Gets the option of imaging
<u>IMAGER_CAMInit</u>	Initializes imaging device
<u>IMAGER_CAMPreviewStart</u>	Starts the preview of imaging
<u>IMAGER_CAMPreviewStop</u>	Stops the preview of imaging
<u>IMAGER_CAMSetOption</u>	Sets the option of imaging
<u>IMAGER_CAMUnInit</u>	Uninitializes imaging device
<u>IMAGER_Close</u>	Closes an open imager
<u>IMAGER_GetAZTEC</u>	Gets the option of AZTEC Barcode
<u>IMAGER_GetCenteringWindow</u>	Gets Enable/Disable of centering window function
<u>IMAGER_GetCHINAPOST</u>	Gets the option of CHINAPOST Barcode
<u>IMAGER_GetCODABAR</u>	Gets the option of CODABAR Barcode
<u>IMAGER_GetCODABLOCK</u>	Gets the option of CODABLOCK Barcode
<u>IMAGER_GetCODE11</u>	Gets the option of CODE11 Barcode
<u>IMAGER_GetCODE128</u>	Gets the option of CODE128 Barcode
<u>IMAGER_GetCODE16K</u>	Gets the option of CODE16K Barcode
<u>IMAGER_GetCODE39</u>	Gets the option of CODE39 Barcode
<u>IMAGER_GetCODE49</u>	Gets the option of CODE49 Barcode
<u>IMAGER_GetCODE93</u>	Gets the option of CODE93 Barcode
<u>IMAGER_GetCOMPOSITE</u>	Gets the option of COMPOSITE Barcode
<u>IMAGER_GetDATAMATRIX</u>	Gets the option of DATAMATRIX Barcode
<u>IMAGER_GetDecOption</u>	Gets the option of Decoder
<u>IMAGER_GetDeviceType</u>	Gets the type of device
<u>IMAGER_GetEAN13</u>	Gets the option of EAN-13 Barcode
<u>IMAGER_GetEAN8</u>	Gets the option of EAN-8 Barcode
<u>IMAGER_GetIATA25</u>	Gets the option of IATA25 Barcode
<u>IMAGER_GetINT25</u>	Gets the option of INT25 Barcode
<u>IMAGER_GetKOREAPOST</u>	Gets the option of KOREAPOST Barcode
<u>IMAGER_GetMATRIX25</u>	Gets the option of MATRIX25 Barcode
<u>IMAGER_GetMAXICODE</u>	Gets the option of MAXICODE Barcode

<u>IMAGER_GetMICROPDF</u>	Gets the option of MICROPDF Barcode
<u>IMAGER_GetMSI</u>	Gets the option of MSI Barcode
<u>IMAGER_GetOCR</u>	Gets the option of OCR Barcode
<u>IMAGER_GetOption</u>	Gets the option of imager
<u>IMAGER_GetPDF417</u>	Gets the option of PDF417 Barcode
<u>IMAGER_GetPLANET</u>	Gets the option of PLANET Barcode
<u>IMAGER_GetPLESSEY</u>	Gets the option of PLESSEY Barcode
<u>IMAGER_GetPOSICODE</u>	Gets the option of POSICODE Barcode
<u>IMAGER_GetPOSTNET</u>	Gets the option of POSTNET Barcode
<u>IMAGER_GetQR</u>	Gets the option of QR Barcode
<u>IMAGER_GetRSS</u>	Gets the option of RSS Barcode
<u>IMAGER_GetScanData</u>	Gets the ScanData which is read by imager
<u>IMAGER_GetSTRT25</u>	Gets the option of STRT25 Barcode
<u>IMAGER_GetSymbology</u>	Gets Enable/Disable of Symbologies
<u>IMAGER_GetTELEPEN</u>	Gets the option of TELEPEN Barcode
<u>IMAGER_GetUPCA</u>	Gets the option of UPC-A Barcode
<u>IMAGER_GetUPCE</u>	Gets the option of UPC-E Barcode
<u>IMAGER_GetVersionInfo</u>	Gets the information of imager driver and dll version
<u>IMAGER_IQGetBarcodeData</u>	Gets the ScanData which is read by imager
<u>IMAGER_IQGetOption</u>	Gets the option of IQ Imaging
<u>IMAGER_IQImagingStart</u>	Starts IQ Imaging
<u>IMAGER_IQImagingStop</u>	Stops IQ Imaging
<u>IMAGER_IQInit</u>	Initializes IQ imaging
<u>IMAGER_IQSetOption</u>	Sets the option of IQ Imaging
<u>IMAGER_IQUnInit</u>	Uninitializes IQ imaging
<u>IMAGER_Open</u>	Opens a imager
<u>IMAGER_Read</u>	Starts the beaming of imager
<u>IMAGER_ReadCancel</u>	Stops the beaming of imager
<u>IMAGER_SetAZTEC</u>	Sets the option of AZTEC Barcode
<u>IMAGER_SetCenteringWindow</u>	Enables centering window function
<u>IMAGER_SetCHINAPOST</u>	Sets the option of CHINAPOST Barcode
<u>IMAGER_SetCODABAR</u>	Sets the option of CODABAR Barcode

<u>IMAGER_SetCODABLOCK</u>	Sets the option of CODABLOCK Barcode
<u>IMAGER_SetCODE11</u>	Sets the option of CODE11 Barcode
<u>IMAGER_SetCODE128</u>	Sets the option of CODE128 Barcode
<u>IMAGER_SetCODE16K</u>	Sets the option of CODE16K Barcode
<u>IMAGER_SetCODE39</u>	Sets the option of CODE39 Barcode
<u>IMAGER_SetCODE49</u>	Sets the option of CODE49 Barcode
<u>IMAGER_SetCODE93</u>	Sets the option of CODE93 Barcode
<u>IMAGER_SetCOMPOSITE</u>	Sets the option of COMPOSITE Barcode
<u>IMAGER_SetDATAMATRIX</u>	Sets the option of DATAMATRIX Barcode
<u>IMAGER_SetDecOption</u>	Sets the option of Decoder
<u>IMAGER_SetEAN13</u>	Sets the option of EAN-13 Barcode
<u>IMAGER_SetEAN8</u>	Sets the option of EAN-8 Barcode
<u>IMAGER_SetIATA25</u>	Sets the option of IATA25 Barcode
<u>IMAGER_SetINT25</u>	Sets the option of INT25 Barcode
<u>IMAGER_SetKOREAPOST</u>	Sets the option of KOREAPOST Barcode
<u>IMAGER_SetMATRIX25</u>	Sets the option of MATRIX25 Barcode
<u>IMAGER_SetMAXICODE</u>	Sets the option of MAXICODE Barcode
<u>IMAGER_SetMICROPDF</u>	Sets the option of MICROPDF Barcode
<u>IMAGER_SetMSI</u>	Sets the option of MSI Barcode
<u>IMAGER_SetOCR</u>	Sets the option of OCR Barcode
<u>IMAGER_SetOption</u>	Sets the option of imager
<u>IMAGER_SetPDF417</u>	Sets the option of PDF417 Barcode
<u>IMAGER_SetPLANET</u>	Sets the option of PLANET Barcode
<u>IMAGER_SetPLESSEY</u>	Sets the option of PLESSEY Barcode
<u>IMAGER_SetPOSICODE</u>	Sets the option of POSICODE Barcode
<u>IMAGER_SetPOSTNET</u>	Sets the option of POSTNET Barcode
<u>IMAGER_SetQR</u>	Sets the option of QR Barcode
<u>IMAGER_SetRSS</u>	Sets the option of RSS Barcode
<u>IMAGER_SetSTRT25</u>	Sets the option of STRT25 Barcode
<u>IMAGER_SetSymbology</u>	Sets the Enable/Disable of Symbologies
<u>IMAGER_SetSymbologyAll</u>	Enables all of Symbologies
<u>IMAGER_SetSymbologyDefault</u>	Initializes all option of scanner

<u>IMAGER_SetTELEPEN</u>	Sets the option of TELEPEN Barcode
<u>IMAGER_SetUPCA</u>	Sets the option of UPC-A Barcode
<u>IMAGER_SetUPCE</u>	Sets the option of UPC-E Barcode

4.2.1 IMAGER_CAMCapture

Description

Captures the preview of imaging.

Syntax

```
IMAGER_API BOOL IMAGER_CAMCapture();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_CAMPreviewStart, IMAGER_CAMPreviewStop

For .Net

Namespace : ImagerNet.Imager

Function : bool CAMCapture()

Example

None

4.2.2 IMAGER_CAMGetOption

Description

Gets the option of imaging.

Syntax

```
IMAGER_API BOOL IMAGER_CAMGetOption(PCAM_PARAMS pCamOption);
```

Parameters

pCamOption

Pointer to a CAM_PARAMS structure to be filled in with the camera parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_CAMSetOption

For .Net

Namespace : ImagerNet.Imager

Function : bool CAMGetOption(out CAM_PARAMS pCamOption)

Example

```
PCAM_PARAMS pCamOption = new CAM_PARAMS();
if(IMAGER_CAMGetOption(pCamOption) == FALSE)
{
    ::MessageBox(NULL, L"Error : IMAGER_CAMGetOption()", NULL, MB_TOPMOST);
    return FALSE;
}
m_nSaveFormat = pCamOption->nSaveFormat;
m_nJpegQuality = pCamOption->nJpegQuality;
m_strSaveFolder = pCamOption->szSaveFolder;
m_strFileName = pCamOption->szFileName;
m_ctrlComboSaveMode.SetCurSel(pCamOption->nSaveMode);
m_ctrlComboResolution.SetCurSel(pCamOption->nResolution);
delete pCamOption;
```

4.2.3 IMAGER_CAMInit

Description

Initializes imaging device.

Syntax

```
IMAGER_API BOOL IMAGER_CAMInit(HWND hPictureWnd);
```

Parameters

hPictureWnd

Window that will show preview.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_CAMUnInit

For .Net

Namespace : ImagerNet.Imager

Function : bool CAMInit(IntPtr hPictureWnd)

Example

None

4.2.4 IMAGER_CAMPreviewStart

Description

Starts the preview of imaging.

Syntax

```
IMAGER_CAMPreviewStart()
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_CAMPreviewStop

For .Net

Namespace : ImagerNet.Imager

Function : bool CAMPreviewStop()

Example

None

4.2.5 IMAGER_CAMPreviewStop

Description

Stops the preview of imaging.

Syntax

```
IMAGER_API BOOL IMAGER_CAMPreviewStop();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_CAMPreviewStart

For .Net

Namespace : ImagerNet.Imager

Function : bool CAMPreviewStart()

Example

None

4.2.6 IMAGER_CAMSetOption

Description

Sets the option of imaging.

Syntax

IMAGER_API BOOL IMAGER_CAMSetOption(PCAM_PARAMS pCamOption);

Parameters

pCamOption

Pointer to a CAM_PARAMS structure holding the camera parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_CAMGetOption

For .Net

Namespace : ImagerNet.Imager

Function : bool CAMSetOption(ref CAM_PARAMS pCamOption)

Example

```
PCAM_PARAMS pCamOption = new CAM_PARAMS();
pCamOption->nSaveFormat = m_nSaveFormat;
pCamOption->nJpegQuality = m_nJpegQuality;
pCamOption->nSaveMode = m_ctrlComboSaveMode.GetCurSel();
pCamOption->nResolution = m_ctrlComboResolution.GetCurSel();
wsprintf(pCamOption->szSaveFolder, L"%s", m_strSaveFolder);
wsprintf(pCamOption->szFileName, L"%s", m_strFileName);
if(IMAGER_CAMSetOption(pCamOption) == FALSE)
{
```

```
        ::MessageBox(NULL, L"Error : IMAGER_CAMSetOption()", NULL, MB_TOPMOST);  
        return FALSE;  
    }  
    delete pCamOption;
```

4.2.7 IMAGER_CAMUnInit

Description

Uninitializes imaging device.

Syntax

```
IMAGER_API BOOL IMAGER_CAMUnInit();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_CAMInit

For .Net

Namespace : ImagerNet.Imager

Function : bool CAMUnInit()

Example

None

4.2.8 IMAGER_Close

Description

Closes an open imager.

Syntax

```
IMAGER_API BOOL IMAGER_Close();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_Open

For .Net

Namespace : ImagerNet.Imager

Function : bool Close()

Example

```
if(IMAGER_Close() == FALSE)
{
    ::MessageBox(NULL, L"Error : IMAGER_Close()", NULL, MB_TOPMOST);
}
```

4.2.9 IMAGER_GetAZTEC

Description

Gets the option of AZTEC Barcode.

Syntax

IMAGER_API BOOL IMAGER_GetAZTEC(PAZTEC_PARAMS pAztec);

Parameters

pAztec

Pointer to a AZTEC_PARAMS structure to be filled in with the AZTEC common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetAZTEC

For .Net

Namespace : ImagerNet.Imager

Function : bool GetAZTEC(out AZTEC_PARAMS pAztec)

Example

```
PAZTEC_PARAMS    pAztec = new AZTEC_PARAMS();
if(IMAGER_GetAZTEC(pAztec) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetAZTEC()", NULL, MB_TOPMOST);
m_ bEnable = pAztec->bEnable;
```

```
m_nMinLen = pAztec->nMinLen;  
m_nMaxLen = pAztec->nMaxLen;  
delete pAztec;
```

4.2.10 IMAGER_GetCenteringWindow

Description

Gets Enable/Disable of centering window function.

Syntax

```
IMAGER_API BOOL IMAGER_GetCenteringWindow(RECT* pRect);
```

Parameters

pRect

Defines the region of the image.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetCenteringWindow

For .Net

Namespace : ImagerNet.Imager

Function : bool GetCenteringWindow(out RECT_PARAM pRect)

Example

```
RECT rect;  
if(!IMAGER_GetCenteringWindow(&rect) == FALSE)  
{  
    ::MessageBox(NULL, L"ERROR : IMAGER_GetCenteringWindow()", NULL, MB_TOPMOST);  
    return FALSE;  
}  
  
m_nEditTop = rect.top;  
m_nEidtBottom = rect.bottom;  
m_nEditLeft = rect.left;  
m_nEditRight = rect.right;
```

4.2.11 IMAGER_GetCHINAPOST

Description

Gets the option of CHINAPOST Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetCHINAPOST(PCHINAPOST_PARAMS pChinaPost);
```

Parameters

pChinaPost

Pointer to a CHINAPOST_PARAMS structure to be filled in with the CHINAPOST common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetCHINAPOST

For .Net

Namespace : ImagerNet.Imager

Function : bool GetCHINAPOST(out CHINAPOST_PARAMS pChinaPost)

Example

```
PCHINAPOST_PARAMS pChinapost = new CHINAPOST_PARAMS();  
if(IMAGER_GetCHINAPOST(pChinapost) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetCHINAPOST()", NULL, MB_TOPMOST);  
m_bEnable = pChinapost->bEnable;  
m_nMinLen = pChinapost->nMinLen;  
m_nMaxLen = pChinapost->nMaxLen;  
delete pChinapost;
```

4.2.12 IMAGER_GetCODABAR

Description

Gets the option of CODABAR Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetCODABAR(PCODABAR_PARAMS pCodabar);
```

Parameters

pCodabar

Pointer to a CODABAR_PARAMS structure to be filled in with the CODABAR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetCODABAR

For .Net

Namespace : ImagerNet.Imager

Function : bool GetCODABAR(out CODABAR_PARAMS pCodabar)

Example

```
PCODABAR_PARAMS  pCodabar = new CODABAR_PARAMS();
if(IMAGER_GetCODABAR(pCodabar) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetCODABAR()", NULL, MB_TOPMOST);
m_bEnable = pCodabar->bEnable;
m_bCDV = pCodabar->bCDV;
m_bXCD = pCodabar->bXCD;
m_bXSS = pCodabar->bXSS;
m_nMinLen = pCodabar->nMinLen;
m_nMaxLen = pCodabar->nMaxLen;
delete pCodabar;
```

4.2.13 IMAGER_GetCODABLOCK

Description

Gets the option of CODABLOCK Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetCODABLOCK(PCODABLOCK_PARAMS pCodablock);
```

Parameters

pCodablock

Pointer to a CODABLOCK_PARAMS structure to be filled in with the CODABLOCK common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetCODABLOCK

For .Net

Namespace : ImagerNet.Imager

Function : bool GetCODABLOCK(out CODABLOCK_PARAMS pCodablock);

Example

```
PCODABLOCK_PARAMS pCodablock = new CODABLOCK_PARAMS();
if(!IMAGER_GetCODABLOCK(pCodablock) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetCODABLOCK()", NULL, MB_TOPMOST);
m_bEnable = pCodablock->bEnable;
m_nMinLen = pCodablock->nMinLen;
m_nMaxLen = pCodablock->nMaxLen;
delete pCodablock;
```

4.2.14 IMAGER_GetCODE11

Description

Gets the option of CODE11 Barcode.

Syntax

IMAGER_API BOOL IMAGER_GetCODE11(PCODE11_PARAMS pCode11);

Parameters

pCode11

Pointer to a CODE11_PARAMS structure to be filled in with the CODE11 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetCODE11

For .Net

Namespace : ImagerNet.Imager

Function : bool GetCODE11(out CODE11_PARAMS pCode11)

Example

```
PCODE11_PARAMS pCode11 = new CODE11_PARAMS();
```

```

if(IMAGER_GetCODE11(pCode11) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetCODE11()", NULL, MB_TOPMOST);
m_bEnable = pCode11->bEnable;
m_bCDV = pCode11->bCDV;
m_nMinLen = pCode11->nMinLen;
m_nMaxLen = pCode11->nMaxLen;
delete pCode11;

```

4.2.15 IMAGER_GetCODE128

Description

Gets the option of CODE128 Barcode

Syntax

```
IMAGER_API BOOL IMAGER_GetCODE128(PCODE128_PARAMS pCode128);
```

Parameters

pCode128

Pointer to a CODE128_PARAMS structure to be filled in with the CODE128 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetCODE128

For .Net

Namespace : ImagerNet.Imager

Function : bool GetCODE128(out CODE128_PARAMS pCode128)

Example

```

PCODE128_PARAMS    pCode128 = new CODE128_PARAMS();
if(IMAGER_GetCODE128(pCode128) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetCODE128()", NULL, MB_TOPMOST);
m_bEnable = pCode128->bEnable;
m_nMinLen = pCode128->nMinLen;
m_nMaxLen = pCode128->nMaxLen;
delete pCode128;

```

4.2.16 IMAGER_GetCODE16K

Description

Gets the option of CODE16K Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetCODE16K(PCODE16K_PARAMS pCode16k);
```

Parameters

pCode16k

Pointer to a CODE16K_PARAMS structure to be filled in with the CODE16K common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetCODE16K

For .Net

Namespace : ImagerNet.Imager

Function : bool GetCODE16K(out CODE16K_PARAMS pCode16k)

Example

```
PCODE16K_PARAMS    pCode16k = new CODE16K_PARAMS();
if(!IMAGER_GetCODE16K(pCode16k) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetCODE16K()", NULL, MB_TOPMOST);
m_bEnable = pCode16k->bEnable;
m_nMinLen = pCode16k->nMinLen;
m_nMaxLen = pCode16k->nMaxLen;
delete pCode16k;
```

4.2.17 IMAGER_GetCODE39

Description

Gets the option of CODE39 Barcode

Syntax

```
IMAGER_API BOOL IMAGER_GetCODE39(PCODE39_PARAMS pCode39);
```

Parameters

pCode39

Pointer to a CODE39_PARAMS structure to be filled in with the CODE39 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetCODE39

For .Net

Namespace : ImagerNet.Imager

Function : bool GetCODE39(out CODE39_PARAMS pCode39)

Example

```
PCODE39_PARAMS pCode39 = new CODE39_PARAMS();
if(IMAGER_GetCODE39(pCode39) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetCODE39()", NULL, MB_TOPMOST);

m_bEnable = pCode39->bEnable;
m_nFormat = pCode39->nFormat;
m_bCDV = pCode39->bCDV;
m_bXCD = pCode39->bXCD;
m_bFullASCII = pCode39->bFullASCII;
m_nMinLen = pCode39->nMinLen;
m_nMaxLen = pCode39->nMaxLen;
delete pCode39;
```

4.2.18 IMAGER_GetCODE49

Description

Gets the option of CODE49 Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetCODE49(PCODE49_PARAMS pCode49);
```

Parameters

pCode49

Pointer to a CODE49_PARAMS structure to be filled in with the CODE49 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetCODE49

For .Net

Namespace : ImagerNet.Imager

Function : bool GetCODE49(out CODE49_PARAMS pCode49)

Example

```
PCODE49_PARAMS    pCode49 = new CODE49_PARAMS();
if(IMAGER_GetCODE49(pCode49) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetCODE49()", NULL, MB_TOPMOST);

m_bEnable = pCode49->bEnable;
m_nMinLen = pCode49->nMinLen;
m_nMaxLen = pCode49->nMaxLen;
delete pCode49;
```

4.2.19 IMAGER_GetCODE93

Description

Gets the option of CODE93 Barcode.

Syntax

IMAGER_API BOOL IMAGER_GetCODE93(PCODE93_PARAMS pCode93);

Parameters

pCode93

Pointer to a CODE93_PARAMS structure to be filled in with the CODE93 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetCODE93

For .Net

Namespace : ImagerNet.Imager

Function : bool GetCODE93(out CODE93_PARAMS pCode93)

Example

```
PCODE93_PARAMS    pCode93 = new CODE93_PARAMS();
if(IMAGER_GetCODE93(pCode93) == FALSE)
```

```

        ::MessageBox(NULL, L"Error : IMAGER_GetCODE93()", NULL, MB_TOPMOST);
m_bEnable    = pCode93->bEnable;
m_nMinLen    = pCode93->nMinLen;
m_nMaxLen    = pCode93->nMaxLen;
delete pCode93;

```

4.2.20 IMAGER_GetCOMPOSITE

Description

Gets the option of COMPOSITE Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetCOMPOSITE(PCOMPOSITE_PARAMS pComposite);
```

Parameters

pComposite

Pointer to a COMPOSITE_PARAMS structure to be filled in with the COMPOSITE common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetCOMPOSITE

For .Net

Namespace : ImagerNet.Imager

Function : bool GetCOMPOSITE(out COMPOSITE_PARAMS pComposite)

Example

```

PCOMPOSITE_PARAMS pComposite = new COMPOSITE_PARAMS();
if(IMAGER_GetCOMPOSITE(pComposite) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetCOMPOSITE()", NULL, MB_TOPMOST);
m_bEnable = pComposite->bEnable;
m_bComposite_Upc = pComposite->bComposite_Upc;
m_nMinLen = pComposite->nMinLen;
m_nMaxLen = pComposite->nMaxLen;
delete pComposite;

```

4.2.21 IMAGER_GetDATAMATRIX

Description

Gets the option of DATAMATRIX Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetDATAMATRIX(PDATAMATRIX_PARAMS pDataMatirx);
```

Parameters

pDataMatirx

Pointer to a DATAMATRIX_PARAMS structure to be filled in with the DATAMATRIX common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetDATAMATRIX

For .Net

Namespace : ImagerNet.Imager

Function : bool GetDATAMATRIX(out DATAMATRIX_PARAMS pDataMatirx)

Example

```
PDATAMATRIX_PARAMS    pDatamatrix = new DATAMATRIX_PARAMS();  
if(IMAGER_GetDATAMATRIX(pDatamatrix) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetDATAMATRIX()", NULL, MB_TOPMOST);  
m_bEnable = pDatamatrix->bEnable;  
m_nMinLen = pDatamatrix->nMinLen;  
m_nMaxLen = pDatamatrix->nMaxLen;  
delete pDatamatrix;
```

4.2.22 IMAGER_GetDecOption

Description

Gets the option of Decoder.

Syntax

```
IMAGER_API BOOL IMAGER_GetDecOption(PDECOPTION_PARAMS pDecOption);
```

Parameters

pDecOption

Pointer to a DECOPTION_PARAMS structure to be filled in with the decoder parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetDecOption

For .Net

Namespace : ImagerNet.Imager

Function : bool GetDecOption(out DECOPTION_PARAMS pDecOption)

Example

```
PDECOPTION_PARAMS pDecOption = new DECOPTION_PARAMS();
if(IMAGER_GetDecOption(pDecOption) == FALSE)
{
    ::MessageBox(NULL, L"ERROR : IMAGER_GetDecOption()", NULL, MB_TOPMOST);
    return FALSE;
}
m_ctrlComboDecodeMode.SetCurSel(pDecOption->nDecodeMode);
m_nLinearRange = pDecOption->nLinearRange;
m_nPrintWeight = pDecOption->nPrintWeight;
m_nMaxDecode = pDecOption->nMaxDecode;
m_nMaxSearch = pDecOption->nMaxSearch;
m_bVideoReverse = pDecOption->bVideoReverse;
delete pDecOption;
```

4.2.23 IMAGER_GetDeviceType

Description

Gets the type of device.

Syntax

```
IMAGER_API SCAN_DEVICE_TYPE IMAGER_GetDeviceType();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : ImagerNet.Imager

Function : SCAN_DEVICE_TYPE GetDeviceType()

Example

None

4.2.24 IMAGER_GetEAN13

Description

Gets the option of EAN-13 Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetEAN13(PEAN13_PARAMS pEan13);
```

Parameters

pEan13

Pointer to a EAN13_PARAMS structure to be filled in with the EAN13 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetEAN13

For .Net

Namespace : ImagerNet.Imager

Function : bool GetEAN13(out EAN13_PARAMS pEan13)

Example

```
PEAN13_PARAMS pEan13 = new EAN13_PARAMS();  
if(IMAGER_GetEAN13(pEan13) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetEAN13()", NULL, MB_TOPMOST);  
m_bEnable = pEan13->bEnable;  
m_bXCD = pEan13->bXCD;  
m_bAddOn = pEan13->bAddOn;  
delete pEan13;
```

4.2.25 IMAGER_GetEAN8

Description

Gets the option of EAN-8 Barcode

Syntax

```
IMAGER_API BOOL IMAGER_GetEAN8(PEAN8_PARAMS pEan8);
```

Parameters

pEan8

Pointer to a MSI_PARAMS structure to be filled in with the MSI common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetEAN8

For .Net

Namespace : ImagerNet.Imager

Function : bool GetEAN8(out EAN8_PARAMS pEan8)

Example

```
PEAN8_PARAMS pEan8 = new EAN8_PARAMS();  
if(!IMAGER_GetEAN8(pEan8) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetEAN8()", NULL, MB_TOPMOST);  
m_bEnable = pEan8->bEnable;  
m_bXCD = pEan8->bXCD;  
m_bAddOn = pEan8->bAddOn;  
delete pEan8;
```

4.2.26 IMAGER_GetIATA25

Description

Gets the option of IATA25 Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetIATA25(PIATA25_PARAMS plata25);
```

Parameters

plata25

Pointer to a IATA25_PARAMS structure to be filled in with the IATA25common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetIATA25

For .Net

Namespace : ImagerNet.Imager

Function : bool GetIATA25(out IATA25_PARAMS plata25)

Example

```
PIATA25_PARAMS    plata25 = new IATA25_PARAMS();
if(IMAGER_GetIATA25(plata25) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetIATA25()", NULL, MB_TOPMOST);

m_ bEnable = plata25->bEnable;
m_nMinLen = plata25->nMinLen;
m_nMaxLen = plata25->nMaxLen;
delete plata25;
```

4.2.27 IMAGER_GetINT25

Description

Sets the option of INT25 Barcode.

Syntax

IMAGER_API BOOL IMAGER_GetINT25(PINT25_PARAMS pInt25);

Parameters

pInt25

Pointer to a INT25_PARAMS structure to be filled in with the INT25 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetINT25

For .Net

Namespace : ImagerNet.Imager

Function : public bool GetINT25(out INT25_PARAMS pInt25)

Example

```
PINT25_PARAMS pInt25 = new INT25_PARAMS();  
if(IMAGER_GetINT25(pInt25) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetINT25()", NULL, MB_TOPMOST);  
m_bEnable = pInt25->bEnable;  
m_bCDV = pInt25->bCDV;  
m_bXCD = pInt25->bXCD;  
m_nMinLen = pInt25->nMinLen;  
m_nMaxLen = pInt25->nMaxLen;  
delete pInt25;
```

4.2.28 IMAGER_GetKOREAPOST

Description

Gets the option of KOREAPOST Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetKOREAPOST(PKOREAPOST_PARAMS pKoreaPost);
```

Parameters

pKoreaPost

Pointer to a KOREAPOST_PARAMS structure to be filled in with the KOREAPOST common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetKOREAPOST

For .Net

Namespace : ImagerNet.Imager

Function : bool GetKOREAPOST(out KOREAPOST_PARAMS pKoreaPost)

Example

```
PKOREAPOST_PARAMS pKoreapost= new KOREAPOST_PARAMS();  
if(IMAGER_GetKOREAPOST(pKoreapost) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetKOREAPOST()", NULL, MB_TOPMOST);  
m_bEnable = pKoreapost->bEnable;
```

```
m_nMinLen = pKoreapost->nMinLen;  
m_nMaxLen = pKoreapost->nMaxLen;  
delete pKoreapost;
```

4.2.29 IMAGER_GetMATRIX25

Description

Gets the option of MATRIX25 Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetMATRIX25(PMATRIX25_PARAMS pMatrix25);
```

Parameters

pMatrix25

Pointer to a MATRIX25_PARAMS structure to be filled in with the MATRIX25 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetMATRIX25

For .Net

Namespace : ImagerNet.Imager

Function : bool GetMATRIX25(out MATRIX25_PARAMS pMatrix25)

Example

```
PMATRIX25_PARAMS pMatrix25 = new MATRIX25_PARAMS();  
if(IMAGER_GetMATRIX25(pMatrix25) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetMATRIX25()", NULL, MB_TOPMOST);  
m_bEnable = pMatrix25->bEnable;  
m_nMinLen = pMatrix25->nMinLen;  
m_nMaxLen = pMatrix25->nMaxLen;  
delete pMatrix25;
```

4.2.30 IMAGER_GetMAXICODE

Description

Gets the option of MAXICODE Barcode

Syntax

IMAGER_API BOOL IMAGER_GetMAXICODE(PMAXICODE_PARAMS pMaxiCode);

Parameters

pMaxiCode

Pointer to a MAXICODE_PARAMS structure to be filled in with the MAXICODE common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetMAXICODE

For .Net

Namespace : ImagerNet.Imager

Function : bool GetMAXICODE(out MAXICODE_PARAMS pMaxiCode)

Example

```
PMAXICODE_PARAMS    pMaxicode = new MAXICODE_PARAMS();
if(!IMAGER_GetMAXICODE(pMaxicode) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetMAXICODE()", NULL, MB_TOPMOST);
m_ bEnable = pMaxicode->bEnable;
m_nMinLen = pMaxicode->nMinLen;
m_nMaxLen = pMaxicode->nMaxLen;
delete pMaxicode;
```

4.2.31 IMAGER_GetMICROPDF

Description

Gets the option of MICROPDF Barcode

Syntax

IMAGER_API BOOL IMAGER_GetMICROPDF(PMICROPDF_PARAMS pMicroPdf);

Parameters

pMicroPdf

Pointer to a MICROPDF_PARAMS structure to be filled in with the MICROPDF common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetMICROPDF

For .Net

Namespace : ImagerNet.Imager

Function : bool GetMICROPDF(out MICROPDF_PARAMS pMicroPdf)

Example

```
PMICROPDF_PARAMS pMicropdf = new MICROPDF_PARAMS();  
if(IMAGER_GetMICROPDF(pMicropdf) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetMICROPDF()", NULL, MB_TOPMOST);  
m_bEnable = pMicropdf->bEnable;  
m_nMinLen = pMicropdf->nMinLen;  
m_nMaxLen = pMicropdf->nMaxLen;  
delete pMicropdf;
```

4.2.32 IMAGER_GetMSI

Description

Gets the option of MSI Barcode

Syntax

IMAGER_API BOOL IMAGER_GetMSI(PMSI_PARAMS pMsi);

Parameters

pMsi

Pointer to a MSI_PARAMS structure to be filled in with the MSI common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetMSI

For .Net

Namespace : ImagerNet.Imager

Function : bool GetMSI(out MSI_PARAMS pMsi)

Example

```
PMSI_PARAMS pMsi = new MSI_PARAMS();
```

```

if(IMAGER_GetMSI(pMsi) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetMSI()", NULL, MB_TOPMOST);
m_bEnable = pMsi->bEnable;
m_bXCD = pMsi->bXCD;
m_nMinLen = pMsi->nMinLen;
m_nMaxLen = pMsi->nMaxLen;
delete pMsi;

```

4.2.33 IMAGER_GetOCR

Description

Gets the option of OCR Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetOCR(POCR_PARAMS pOcr);
```

Parameters

pOcr

Pointer to a OCR_PARAMS structure to be filled in with the OCR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetOCR

For .Net

Namespace : ImagerNet.Imager

Function : bool GetOCR(out OCR_PARAMS pOcr)

Example

```

POCR_PARAMS pOcr = new OCR_PARAMS();
if(IMAGER_GetOCR(pOcr) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetOcr()", NULL, MB_TOPMOST)
m_ctrlComboMode.InsertString(0, L"OCR_DISABLED");
m_ctrlComboMode.InsertString(1, L"OCR_A");
m_ctrlComboMode.InsertString(2, L"OCR_B");
m_ctrlComboMode.InsertString(3, L"OCR_MONEY");
m_ctrlComboMode.InsertString(4, L"OCR_MICR_UNSUPPORTED");

```



```

m_ctrlComboMode.SetCurSel(pOcr->nMode);
m_bEnable = pOcr->bEnable;
m_strEditTemplate = (CString)pOcr->szTemplate;
m_strEditGroupG = (CString)pOcr->szGroupG;
m_strEditGroupH = (CString)pOcr->szGroupH;
m_strCheckChar = (CString)pOcr->szCheckChar;
delete pOcr;

```

4.2.34 IMAGER_GetOption

Description

Gets the option of imager.

Syntax

```
IMAGER_API BOOL IMAGER_GetOption(PDECODER_PARAMS pOption);
```

Parameters

pOption

Pointer to a DECODER_PARAMS structure to be filled in with the scanner parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetOption

For .Net

Namespace : ImagerNet.Imager

Function : bool GetOption(out DECODER_PARAMS pOption)

Example

```

PDECODER_PARAMS pOption = new DECODER_PARAMS();
if(IMAGER_GetOption(pOption) == FALSE)
{
    ::MessageBox(NULL, L"ERROR : IMAGER_GetOption()", NULL, MB_TOPMOST);
    return FALSE;
}
m_nSound = pOption->nSound;
m_bCentering = pOption->bCentering;

```

```

m_bVibrate = pOption->bVibrate;
m_bAimID = pOption->bXmitAimID;
m_bContinueMode = pOption->bContinueMode;
m_bHexMode = pOption->bHexMode;
m_ctrlComboTimeOut.SetCurSel(pOption->nTimeOut - 1);
m_ctrlComboLightMode.SetCurSel(pOption->nLightMode);
delete pOption;

```

4.2.35 IMAGER_GetPDF417

Description

Gets the option of PDF417 Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetPDF417(PDF417_PARAMS pPdf417);
```

Parameters

pPdf417

Pointer to a PDF417_PARAMS structure to be filled in with the PDF417 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetPDF417

For .Net

Namespace : ImagerNet.Imager

Function : bool GetPDF417(out PDF417_PARAMS pPdf417)

Example

```

PDF417_PARAMS      pPdf417 = new PDF417_PARAMS();
if(IMAGER_GetPDF417(pPdf417) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetPDF417()", NULL, MB_TOPMOST);
m_bEnable = pPdf417->bEnable;
m_nMinLen = pPdf417->nMinLen;
m_nMaxLen = pPdf417->nMaxLen;
delete pPdf417;

```

4.2.36 IMAGER_GetPLANET

Description

Gets the option of PLANET Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetPLANET(PPLANET_PARAMS pPlanet);
```

Parameters

pPlanet

Pointer to a PLANET_PARAMS structure to be filled in with the PLANET common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetPLANET

For .Net

Namespace : ImagerNet.Imager

Function : public bool GetPLANET(out PLANET_PARAMS pPlanet)

Example

```
PPLANET_PARAMS pPlanet = new PLANET_PARAMS();  
if(IMAGER_GetPLANET(pPlanet) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetPLANET()", NULL, MB_TOPMOST);  
m_bEnable = pPlanet->bEnable;  
m_bXCD = pPlanet->bXCD;  
delete pPlanet;
```

4.2.37 IMAGER_GetPLESSEY

Description

Gets the option of PLESSEY Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetPLESSEY(PPLESSEY_PARAMS pPlessey);
```

Parameters

pPlessey

Pointer to a PLESSEY_PARAMS structure to be filled in with the PLESSEY common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetPLESSEY

For .Net

Namespace : ImagerNet.Imager

Function : bool GetPLESSEY(out PLESSEY_PARAMS pPlessey)

Example

```
PPLESSEY_PARAMS      pPlessey = new PLESSEY_PARAMS();
if(IMAGER_GetPLESSEY(pPlessey) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetPLESSEY()", NULL, MB_TOPMOST);
m_bEnable = pPlessey->bEnable;
m_nMinLen = pPlessey->nMinLen;
m_nMaxLen = pPlessey->nMaxLen;
delete pPlessey;
```

4.2.38 IMAGER_GetPOSICODE

Description

Gets the option of POSICODE Barcode.

Syntax

IMAGER_API BOOL IMAGER_GetPOSICODE(PPOSICODE_PARAMS pPosiCode);

Parameters

pPosiCode

Pointer to a POSICODE_PARAMS structure to be filled in with the POSICODE common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetPOSICODE

For .Net

Namespace : ImagerNet.Imager

Function : bool GetPOSICODE(out POSICODE_PARAMS pPosiCode);

Example

```
PPOSICODE_PARAMS pPosicode = new POSICODE_PARAMS();
if(IMAGER_GetPOSIcode(pPosicode) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetPOSIcode()", NULL, MB_TOPMOST);
m_bEnable = pPosicode->bEnable;
m_bPosi_Lim1 = pPosicode->bPosi_Lim1;
m_bPosi_Lim2 = pPosicode->bPosi_Lim2;
m_nMinLen = pPosicode->nMinLen;
m_nMaxLen = pPosicode->nMaxLen;
delete pPosicode;
```

4.2.39 IMAGER_GetPOSTNET

Description

Gets the option of POSTNET Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetPOSTNET(PPOSTNET_PARAMS pPostNet);
```

Parameters

pPostNet

Pointer to a POSTNET_PARAMS structure to be filled in with the POSTNET common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetPOSTNET

For .Net

Namespace : ImagerNet.Imager

Function : bool GetPOSTNET(out POSTNET_PARAMS pPostNet)

Example

```
PPOSTNET_PARAMS pPostnet = new POSTNET_PARAMS();
if(IMAGER_GetPOSTNET(pPostnet) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetPOSTNET()", NULL, MB_TOPMOST);
m_bEnable = pPostnet->bEnable;
m_bXCD = pPostnet->bXCD;
```

delete pPostnet;

4.2.40 IMAGER_GetQR

Description

Gets the option of QR Barcode

Syntax

```
IMAGER_API BOOL IMAGER_GetQR(PQR_PARAMS pQr);
```

Parameters

pQr

Pointer to a QR_PARAMS structure to be filled in with the QR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetQR

For .Net

Namespace : ImagerNet.Imager

Function : bool GetQR(out QR_PARAMS pQr)

Example

```
PQR_PARAMS      pQr = new QR_PARAMS();  
if(!IMAGER_GetQR(pQr) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetQR()", NULL, MB_TOPMOST);  
m_bEnable = pQr->bEnable;  
m_nMinLen = pQr->nMinLen;  
m_nMaxLen = pQr->nMaxLen;  
delete pQr;
```

4.2.41 IMAGER_GetRSS

Description

Gets the option of RSS Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetRSS(PRSS_PARAMS pRss);
```

Parameters

pRss

Pointer to a RSS_PARAMS structure to be filled in with the RSS common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetRSS

For .Net

Namespace : ImagerNet.Imager

Function : bool GetRSS(out RSS_PARAMS pRss)

Example

```
PRSS_PARAMS pRss = new RSS_PARAMS();  
if(IMAGER_GetRSS(pRss) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetRSS()", NULL, MB_TOPMOST);  
m_bEnable = pRss->bEnable;  
m_bRssLim = pRss->bRssLim;  
m_bRssExp = pRss->bRssExp;  
m_nMinLen = pRss->nMinLen;  
m_nMaxLen = pRss->nMaxLen;  
delete pRss;
```

4.2.42 IMAGER_GetScanByteData

Description

Gets the ScanData which is read by imager.

Syntax

IMAGER_API BOOL IMAGER_GetScanByteData(BYTE *pbyBarType, BYTE *pbyBarData, int *pnLength);

Parameters

pszBarType

Pointer to barcode type.

pszBarData

Pointer to barcode data

pnLength

Pointer to Length of barcode data

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : ImagerNet.Imager

Function : `bool GetScanDataBytes(ref byte[] pbyBarType, ref byte [] pbyBarData, ref int pnLength)`

Example

```
BYTE pbyBarType[128] = {0, };
```

```
BYTE pbyBarData[8192] = {0, };
```

```
Int pnLength=0;
```

```
IMAGER_GetScanByteData(pbyBarType, pbyBarData, &pnLength);
```

4.2.43 IMAGER_GetScanData

Description

Gets the ScanData which is read by imager.

Syntax

```
IMAGER_API BOOL IMAGER_GetScanData(TCHAR *pszBarType, TCHAR *pszBarData);
```

Parameters

pszBarType

Pointer to barcode type.

pszBarData

Pointer to barcode data

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : ImagerNet.Imager

Function : `bool GetScanData(StringBuilder szBarType, StringBuilder szBarData)`

Example

```
TCHAR  szBarType[1024] = {0, };
TCHAR  szBarData[1024] = {0, };
IMAGER_GetScanData(szBarType, szBarData);
```

4.2.44 IMAGER_GetSTRT25

Description

Gets the option of STRT25 Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetSTRT25(PSTRT25_PARAMS pStrt25);
```

Parameters

pStrt25

Pointer to a STRT25_PARAMS structure to be filled in with the STRT25 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetSTRT25

For .Net

Namespace : ImagerNet.Imager

Function : bool GetSTRT25(out STRT25_PARAMS pStrt25)

Example

```
PSTRT25_PARAMS  pStrt25 = new STRT25_PARAMS();
if(IMAGER_GetSTRT25(pStrt25) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetSTRT25()", NULL, MB_TOPMOST);
m_bEnable = pStrt25->bEnable;
m_nMinLen = pStrt25->nMinLen;
m_nMaxLen = pStrt25->nMaxLen;
delete pStrt25;
```

4.2.45 IMAGER_GetSymbology

Description

Gets Enable/Disable of Symbologies.

Syntax

```
IMAGER_API BOOL IMAGER_GetSymbology(PDECODER pSymbology);
```

Parameters

pSymbology

Pointer to a DECODER structure to be filled in with the symbologies enable or disable.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetSymbology

For .Net

Namespace : ImagerNet.Imager

Function : bool GetSymbology(out DECODER pSymbology)

Example

```
PDECODER pSym = new DECODER();
BOOL bEnable[47] = {0, };
int i = 0;
if(IMAGER_GetSymbology(pSym) == FALSE)
    ::MessageBox(NULL, L"ERROR : IMAGER_GetSymbology()", NULL, MB_TOPMOST);
bEnable[0] = pSym->bAZTEC;
bEnable[1] = pSym->bCODABAR;
bEnable[2] = pSym->bCODE11;
bEnable[3] = pSym->bCODE128;
bEnable[4] = pSym->bCODE39;
bEnable[5] = pSym->bCODE49;
bEnable[6] = pSym->bCODE93;
bEnable[7] = pSym->bCOMPOSITE;
bEnable[8] = pSym->bDATAMATRIX;
bEnable[9] = pSym->bEAN8;
bEnable[10] = pSym->bEAN13;
bEnable[11] = pSym->bINT25;
bEnable[12] = pSym->bMAXICODE;
bEnable[13] = pSym->bMICROPDF;
```

```
bEnable[14] = pSym->bOCR;
bEnable[15] = pSym->bPDF417;
bEnable[16] = pSym->bPOSTNET;
bEnable[17] = pSym->bQR;
bEnable[18] = pSym->bRSS;
bEnable[19] = pSym->bUPCA;
bEnable[20] = pSym->bUPCE;
bEnable[21] = pSym->bISBT;
bEnable[22] = pSym->bBPO;
bEnable[23] = pSym->bCANPOST;
bEnable[24] = pSym->bAUSPOST;
bEnable[25] = pSym->bIATA25;
bEnable[26] = pSym->bCODABLOCK;
bEnable[27] = pSym->bJAPOST;
bEnable[28] = pSym->bPLANET;
bEnable[29] = pSym->bDUTCHPOST;
bEnable[30] = pSym->bMSI;
bEnable[31] = pSym->bTLCODE39;
bEnable[32] = pSym->bSTRT25;
bEnable[33] = pSym->bMATRIX25;
bEnable[34] = pSym->bPLESSEY;
bEnable[35] = pSym->bCHINAPOST;
bEnable[36] = pSym->bKOREAPOST;
bEnable[37] = pSym->bTELEPEN;
bEnable[38] = pSym->bCODE16K;
bEnable[39] = pSym->bPOSICODE;
bEnable[40] = pSym->bCOUPONCODE;
bEnable[41] = pSym->bUSPS4CB;
bEnable[42] = pSym->bIDTAG;
bEnable[43] = pSym->bLABEL;
bEnable[44] = pSym->bGS1_128;
bEnable[45] = pSym->bHANXIN;
bEnable[46] = pSym->bGRIDMATRIX;
for(i=0; i<47; i++)
```

```
{
    m_ctrlListSymbology.SetCheck(i, bEnable[i]);
}

delete pSym;
```

4.2.46 IMAGER_GetTELEPEN

Description

Gets the option of TELEPEN Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetTELEPEN(PTELEPEN_PARAMS pTelepen);
```

Parameters

pTelepen

Pointer to a TELEPEN_PARAMS structure to be filled in with the TELEPEN common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetTELEPEN

For .Net

Namespace : ImagerNet.Imager

Function : bool GetTELEPEN(out TELEPEN_PARAMS pTelepen)

Example

```
PTELEPEN_PARAMS pTelepen = new TELEPEN_PARAMS();
if(IMAGER_GetTELEPEN(pTelepen) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetTELEPEN()", NULL, MB_TOPMOST);

m_bEnable = pTelepen->bEnable;
m_bNumeric = pTelepen->bNumeric;
m_nMinLen = pTelepen->nMinLen;
m_nMaxLen = pTelepen->nMaxLen;

delete pTelepen;
```

4.2.47 IMAGER_GetUPCA

Description

Gets the option of UPC-A Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetUPCA(PUPCA_PARAMS pUpca);
```

Parameters

pUpca

Pointer to a UPCA_PARAMS structure to be filled in with the UPC-A common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetUPCA

For .Net

Namespace : ImagerNet.Imager

Function : bool GetUPCA(out UPCA_PARAMS pUpca)

Example

```
PUPCA_PARAMS pUpca = new UPCA_PARAMS();
if(IMAGER_GetUPCA(pUpca) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetUPCA()", NULL, MB_TOPMOST);
m_bEnable = pUpca->bEnable;
m_bXCD = pUpca->bXCD;
m_bXNum = pUpca->bXNum;
m_bAddOn = pUpca->bAddOn;
delete pUpca;
```

4.2.48 IMAGER_GetUPCE

Description

Gets the option of UPC-E Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_GetUPCE(PUPCE_PARAMS pUpce);
```

Parameters

pUpce

Pointer to a UPCE_PARAMS structure to be filled in with the UPC-E common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetUPCE

For .Net

Namespace : ImagerNet.Imager

Function : bool GetUPCE(out UPCE_PARAMS pUpce)

Example

```
PUPCE_PARAMS pUpce = new UPCE_PARAMS();
if(IMAGER_GetUPCE(pUpce) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetUPCE()", NULL, MB_TOPMOST);
m_bEnable = pUpce->bEnable;
m_bXCD = pUpce->bXCD;
m_bXNum = pUpce->bXNum;
m_bAddOn = pUpce->bAddOn;
delete pUpce;
```

4.2.49 IMAGER_GetVersionInfo

Description

Gets the information of imager driver and dll version.

Syntax

```
IMAGER_API BOOL IMAGER_GetVersionInfo(TCHAR* pszVersion);
```

Parameters

pszVersion

Pointer to a TCHAR to be filled in with the version info.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

none

For .Net

Namespace : ImagerNet.Imager

Function : string GetVersionInfo()

Example

```
TCHAR szVersionInfo[1024] = {0, };  
IMAGER_GetVersionInfo(szVersionInfo);
```

4.2.50 IMAGER_IQGetBarcodeData

Description

Gets the ScanData which is read by imager.

Syntax

```
IMAGER_API BOOL IMAGER_IQGetBarcodeData(TCHAR *pszBarData);
```

Parameters

pszBarData

Pointer to barcode data.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : ImagerNet.Imager

Function : bool IQGetBarcodeData(StringBuilder szBarData)

Example

```
TCHAR  szBarData[1024] = {0, };  
IMAGER_IQGetBarcodeData(szBarData);
```

4.2.51 IMAGER_IQGetOption

Description

Gets the option of IQ Imaging

Syntax

```
IMAGER_API BOOL IMAGER_IQGetOption(PIQ_PARAMS plQOption);
```

Parameters

plQOption

Pointer to an IQ_PARAMS structure to be filled in with the IQ Imaging parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : ImagerNet.Imager

Function : bool IQGetOption(out IQ_PARAMS plQOption)

Example

```
PIQ_PARAMS plQOption = new IQ_PARAMS();
if(IMAGER_IQGetOption(plQOption) == FALSE)
{
    ::MessageBox(NULL, L"Error : IMAGER_IQGetOption()", NULL, MB_TOPMOST);
    return FALSE;
}
m_nIQType = plQOption->nIQType;
m_strIQSaveFolder = plQOption->szSaveFolder;
m_strIQFileName = plQOption->szFileName;
m_strSaveFormat = L".BMP";
m_ctrlComboSaveMode.SetCurSel(plQOption->nSaveMode);
delete plQOption;
```

4.2.52 IMAGER_IQImagingStart

Description

Starts IQ Imaging.

Syntax

IMAGER_API BOOL IMAGER_IQImagingStart();

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_IQImagingStop

For .Net

Namespace : ImagerNet.Imager

Function : bool IQImagingStart()

Example

None

4.2.53 IMAGER_IQImagingStop

Description

Stops IQ Imaging.

Syntax

```
IMAGER_API BOOL IMAGER_IQImagingStop();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_IQImagingStart

For .Net

Namespace : ImagerNet.Imager

Function : bool IQImagingStop()

Example

None

4.2.54 IMAGER_IQInit

Description

Initializes IQ imaging

Syntax

```
IMAGER_API BOOL IMAGER_IQInit(HWND hPictureWnd);
```

Parameters

hPictureWnd

Window that will show preview.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_IQUnInit

For .Net

Namespace : ImagerNet.Imager

Function : bool IQInit(IntPtr hPictureWnd);

Example

None

4.2.55 IMAGER_IQSetOption

Description

Sets the option of IQ Imaging.

Syntax

IMAGER_API BOOL IMAGER_IQSetOption(PIQ_PARAMS pIQOption);

Parameters

pIQOption

Pointer to a IQ_PARAMS structure holding the IQ Imaging parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_IQGetOption

For .Net

Namespace : ImagerNet.Imager

Function : bool IQSetOption(ref IQ_PARAMS pIQOption)

Example

```
PIQ_PARAMS pIQOption = new IQ_PARAMS();  
pIQOption->nSaveMode = m_ctrlComboSaveMode.GetCurSel();  
pIQOption->nIQType = m_nIQType;  
wsprintf(pIQOption->szSaveFolder, L"%s", m_strIqSaveFolder);
```

```
wsprintf(plQOption->szFileName, L"%s", m_strIqFileName);  
if(IMAGER_IQSetOption(plQOption) == FALSE)  
{  
    ::MessageBox(NULL, L"Error : IMAGER_IQSetOption()", NULL, MB_TOPMOST);  
    return FALSE;  
}  
delete plQOption;
```

4.2.56 IMAGER_IQUnInit

Description

Uninitializes IQ imaging.

Syntax

```
IMAGER_API BOOL IMAGER_IQUnInit();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_IQInit

For .Net

Namespace : ImagerNet.Imager

Function : public bool IQUnInit()

Example

None

4.2.57 IMAGER_Open

Description

Opens an imager.

Syntax

```
IMAGER_API BOOL IMAGER_Open();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_Close

For .Net

Namespace : ImagerNet.Imager

Function : bool Open()

Example

None

4.2.58 IMAGER_Read

Description

Starts the beaming of imager

Syntax

```
IMAGER_API BOOL IMAGER_Read();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_ReadCancel

For .Net

Namespace : ImagerNet.Imager

Function : bool Read()

Example

None

4.2.59 IMAGER_ReadCancel

Description

Stops the beaming of imager

Syntax

IMAGER_API BOOL IMAGER_ReadCancel();

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_Read

For .Net

Namespace : ImagerNet.Imager

Function : bool ReadCancel()

Example

None

4.2.60 IMAGER_SetAZTEC

Description

Sets the option of AZTEC Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetAZTEC(PAZTEC_PARAMS pAztec);

Parameters

pAztec

Pointer to a AZTEC_PARAMS structure holding the AZTEC common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetAZTEC

For .Net

Namespace : ImagerNet.Imager

Function : bool SetAZTEC(ref AZTEC_PARAMS pAztec)

Example

PAZTEC_PARAMS pAztec = new AZTEC_PARAMS();

```

pAztec->bEnable = m_bEnable;
pAztec->nMinLen = m_nMinLen;
pAztec->nMaxLen = m_nMaxLen;
if(IMAGER_SetAZTEC(pAztec) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_ SetAZTEC ()", NULL, MB_TOPMOST);
delete pAztec;

```

4.2.61 IMAGER_SetCenteringWindow

Description

Enables centering window function.

Syntax

```
IMAGER_API BOOL IMAGER_SetCenteringWindow(RECT* pRect);
```

Parameters

pRect

Defines the region of the image.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetCenteringWindow

For .Net

Namespace : ImagerNet.Imager

Function : bool SetCenteringWindow(ref RECT_PARAM pRect)

Example

```

RECT rect;
rect.top = m_nEditTop;
rect.bottom = m_nEditBottom;
rect.left = m_nEditLeft;
rect.right = m_nEditRight;
if(IMAGER_SetCenteringWindow(&rect) == FALSE)
{
    ::MessageBox(NULL, L"ERROR : IMAGER_SetCenteringWindow()", NULL, MB_TOPMOST);
    return FALSE;
}

```

}

4.2.62 IMAGER_SetCHINAPOST

Description

Sets the option of CHINAPOST Barcode

Syntax

```
IMAGER_API BOOL IMAGER_SetCHINAPOST(PCHINAPOST_PARAMS pChinaPost);
```

Parameters

pChinaPost

Pointer to a CHINAPOST_PARAMS structure holding the CHINAPOST common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetCHINAPOST

For .Net

Namespace : ImagerNet.Imager

Function : bool SetCHINAPOST(ref CHINAPOST_PARAMS pChinaPost);

Example

```
PCHINAPOST_PARAMS pChinapost = new CHINAPOST_PARAMS();
pChinapost->bEnable = m_bEnable;
pChinapost->nMinLen = m_nMinLen;
pChinapost->nMaxLen = m_nMaxLen;
if(IMAGER_SetCHINAPOST(pChinapost) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_SetCHINAPOST()", NULL, MB_TOPMOST);
delete pChinapost;
```

4.2.63 IMAGER_SetCODABAR

Description

Sets the option of CODABAR Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetCODABAR(PCODABAR_PARAMS pCodabar);
```

Parameters

pCodabar

Pointer to a CODABAR_PARAMS structure holding the CODABAR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetCODABAR

For .Net

Namespace : ImagerNet.Imager

Function : bool SetCODABAR(ref CODABAR_PARAMS pCodabar)

Example

```
PCODABAR_PARAMS  pCodabar = new CODABAR_PARAMS();
pCodabar->bEnable = m_bEnable;
pCodabar->bCDV = m_bCDV;
pCodabar->bXCD = m_bXCD;
pCodabar->bXSS = m_bXSS;
pCodabar->nMinLen = m_nMinLen;
pCodabar->nMaxLen = m_nMaxLen;
if(IMAGER_SetCODABAR(pCodabar) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_SetCODABAR()", NULL, MB_TOPMOST);
delete pCodabar;
```

4.2.64 IMAGER_SetCODABLOCK

Description

Sets the option of CODABLOCK Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetCODABLOCK(PCODABLOCK_PARAMS pCodablock);

Parameters

pCodablock

Pointer to a CODABLOCK_PARAMS structure holding the CODABLOCK common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetCODABLOCK

For .Net

Namespace : ImagerNet.Imager

Function : bool SetCODABLOCK(ref CODABLOCK_PARAMS pCodablock)

Example

```
PCODABLOCK_PARAMS pCodablock = new CODABLOCK_PARAMS();  
pChinapost->bEnable = m_bEnable;  
pChinapost->nMinLen = m_nMinLen;  
pChinapost->nMaxLen = m_nMaxLen;  
if(IMAGER_SetCODABLOCK(pCodablock) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_SetCODABLOCK()", NULL, MB_TOPMOST);  
delete pChinapost;
```

4.2.65 IMAGER_SetCODE11

Description

Sets the option of CODE11 Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetCODE11(PCODE11_PARAMS pCode11);

Parameters

pCode11

Pointer to a CODE11_PARAMS structure holding the CODE11 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetCODE11

For .Net

Namespace : ImagerNet.Imager

Function : bool SetCODE11(ref CODE11_PARAMS pCode11)

Example

```
PCODE11_PARAMS pCode11 = new CODE11_PARAMS();
```

```

pCode11->bEnable = m_bEnable;
pCode11->bCDV = m_bCDV;
pCode11->nMinLen = m_nMinLen;
pCode11->nMaxLen = m_nMaxLen;
if(!IMAGER_SetCODE11(pCode11) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetCODE11()", NULL, MB_TOPMOST);
delete pCode11;

```

4.2.66 IMAGER_SetCODE128

Description

Sets the option of CODE128 Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetCODE128(PCODE128_PARAMS pCode128);
```

Parameters

pCode128

Pointer to a CODE128_PARAMS structure holding the CODE128 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetCODE128

For .Net

Namespace : ImagerNet.Imager

Function : bool SetCODE128(ref CODE128_PARAMS pCode128)

Example

```

PCODE128_PARAMS pCode128 = new CODE128_PARAMS();
pCode128->bEnable = m_bEnable;
pCode128->nMinLen = m_nMinLen;
pCode128->nMaxLen = m_nMaxLen;
if(!IMAGER_SetCODE128(pCode128) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetCODE128()", NULL, MB_TOPMOST);
delete pCode128;

```

4.2.67 IMAGER_SetCODE16K

Description

Sets the option of CODE16K Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetCODE16K(PCODE16K_PARAMS pCode16k);
```

Parameters

pCode16k

Pointer to a CODE16K_PARAMS structure holding the CODE16K common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetCODE16K

For .Net

Namespace : ImagerNet.Imager

Function : bool SetCODE16K(ref CODE16K_PARAMS pCode16k);

Example

```
PCODE16K_PARAMS pCode16k = new CODE16K_PARAMS();  
pCode16k->bEnable = m_bEnable;  
pCode16k->nMinLen = m_nMinLen;  
pCode16k->nMaxLen = m_nMaxLen;  
if(IMAGER_SetCODE16K(pCode16k) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_SetCODE16K()", NULL, MB_TOPMOST);  
delete pCode16k;
```

4.2.68 IMAGER_SetCODE39

Description

Sets the option of CODE39 Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetCODE39(PCODE39_PARAMS pCode39);
```

Parameters

pCode39

Pointer to a CODE39_PARAMS structure holding the CODE39 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetCODE39

For .Net

Namespace : ImagerNet.Imager

Function : bool SetCODE39(ref CODE39_PARAMS pCode39);

Example

```
PCODE39_PARAMS pCode39 = new CODE39_PARAMS();
pCode39->bEnable = m_bEnable;
pCode39->nFormat = m_nFormat;
pCode39->bCDV = m_bCDV;
pCode39->bXCD = m_bXCD;
pCode39->bFullASCII = m_bFullASCII;
pCode39->nMinLen = m_nMinLen;
pCode39->nMaxLen = m_nMaxLen;
if(IMAGER_SetCODE39(pCode39) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetCODE39()", NULL, MB_TOPMOST);
delete pCode39;
```

4.2.69 IMAGER_SetCODE49

Description

Sets the option of CODE49 Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetCODE49(PCODE49_PARAMS pCode49);

Parameters

pCode49

Pointer to a CODE49_PARAMS structure holding the CODE49 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetCODE49

For .Net

Namespace : ImagerNet.Imager

Function : bool SetCODE49(ref CODE49_PARAMS pCode49)

Example

```
PCODE49_PARAMS pCode49 = new CODE49_PARAMS();  
pCode49->bEnable = m_bEnable;  
pCode49->nMinLen = m_nMinLen;  
pCode49->nMaxLen = m_nMaxLen;  
if(IMAGER_SetCODE49(pCode49) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetCODE49()", NULL, MB_TOPMOST);  
delete pCode49;
```

4.2.70 IMAGER_SetCODE93

Description

Sets the option of CODE93 Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetCODE93(PCODE93_PARAMS pCode93);

Parameters

pCode93

Pointer to a CODE93_PARAMS structure holding the CODE93 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetCODE93

For .Net

Namespace : ImagerNet.Imager

Function : bool SetCODE93(ref CODE93_PARAMS pCode93);

Example

```
PCODE93_PARAMS pCode93 = new CODE93_PARAMS();  
pCode93->bEnable = m_bEnable;
```

```

pCode93->nMinLen = m_nMinLen;
pCode93->nMaxLen = m_nMaxLen;
if(IMAGER_SetCODE93(pCode93) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetCODE93()", NULL, MB_TOPMOST);
delete pCode93;

```

4.2.71 IMAGER_SetCOMPOSITE

Description

Sets the option of COMPOSITE Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetCOMPOSITE(PCOMPOSITE_PARAMS pComposite);
```

Parameters

pComposite

Pointer to a COMPOSITE_PARAMS structure holding the COMPOSITE common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetCOMPOSITE

For .Net

Namespace : ImagerNet.Imager

Function : bool SetCOMPOSITE(ref COMPOSITE_PARAMS pComposite)

Example

```

PCOMPOSITE_PARAMS pComposite = new COMPOSITE_PARAMS();
pComposite->bEnable = m_bEnable;
pComposite->bComposite_Upc = m_bComposite_Upc;
pComposite->nMinLen = m_nMinLen;
pComposite->nMaxLen = m_nMaxLen;
if(IMAGER_SetCOMPOSITE(pComposite) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetCOMPOSITE()", NULL, MB_TOPMOST);
delete pComposite;

```

4.2.72 IMAGER_SetDATAMATRIX

Description

Sets the option of DATAMATRIX Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetDATAMATRIX(PDATAMATRIX_PARAMS pDataMatrix);
```

Parameters

pDataMatrix

Pointer to a DATAMATRIX_PARAMS structure holding the DATAMATRIX common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetDATAMATRIX

For .Net

Namespace : ImagerNet.Imager

Function : bool SetDATAMATRIX(ref DATAMATRIX_PARAMS pDataMatrix)

Example

```
PDATAMATRIX_PARAMS    pDataMatrix = new DATAMATRIX_PARAMS();
pDataMatrix->bEnable = m_bEnable;
pDataMatrix->nMinLen = m_nMinLen;
pDataMatrix->nMaxLen = m_nMaxLen;
if(IMAGER_SetCOMPOSITE(pComposite) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_GetCOMPOSITE()", NULL, MB_TOPMOST);
delete pDataMatrix;
```

4.2.73 IMAGER_SetDecOption

Description

Sets the option of Decoder.

Syntax

```
IMAGER_API BOOL IMAGER_SetDecOption(PDECOPTION_PARAMS pDecOption);
```

Parameters

pDecOption

Pointer to a DECOPTION_PARAMS structure holding the decoder parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetDecOption

For .Net

Namespace : ImagerNet.Imager

Function : bool SetDecOption(ref DECOPTION_PARAMS pDecOption)

Example

```
PDECOPTION_PARAMS pDecOption = new DECOPTION_PARAMS();
pDecOption->nDecodeMode = m_ctrlComboDecodeMode.GetCurSel();
pDecOption->nLinearRange = m_nLinearRange;
pDecOption->nPrintWeight = m_nPrintWeight;
pDecOption->nMaxDecode = m_nMaxDecode;
pDecOption->nMaxSearch = m_nMaxSearch;
pDecOption->bVideoReverse = m_bVideoReverse;
if(IMAGER_SetDecOption(pDecOption) == FALSE)
{
    ::MessageBox(NULL, L"ERROR : IMAGER_SetDecOption()", NULL, MB_TOPMOST);
    return FALSE;
}
delete pDecOption;
```

4.2.74 IMAGER_SetEAN13

Description

Sets the option of EAN-13 Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetEAN13(PEAN13_PARAMS pEan13);

Parameters

pEan13

Pointer to an EAN13_PARAMS structure holding the EAN-13 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetEAN13

For .Net

Namespace : ImagerNet.Imager

Function : bool SetEAN13(ref EAN13_PARAMS pEan13)

Example

```
PEAN13_PARAMS pEan13 = new EAN13_PARAMS();  
pEan13->bEnable = m_bEnable;  
pEan13->bXCD = m_bXCD;  
pEan13->bAddOn = m_bAddOn;  
if(IMAGER_SetEAN13(pEan13) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_SetEAN13()", NULL, MB_TOPMOST);  
delete pEan13;
```

4.2.75 IMAGER_SetEAN8

Description

Sets the option of EAN-8 Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetEAN8(PEAN8_PARAMS pEan8);

Parameters

pEan8

Pointer to an EAN8_PARAMS structure holding the EAN-8 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetEAN8

For .Net

Namespace : ImagerNet.Imager

Function : bool SetEAN8(ref EAN8_PARAMS pEan8)

Example

```
PEAN8_PARAMS pEan8 = new EAN8_PARAMS();
```

```

pEan8->bEnable = m_bEnable;
pEan8->bXCD = m_bXCD;
pEan8->bAddOn = m_bAddOn;
if(IMAGER_SetEAN8(pEan8) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetEAN8()", NULL, MB_TOPMOST);
delete pEan8;

```

4.2.76 IMAGER_SetIATA25

Description

Sets the option of IATA25 Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetIATA25(PIATA25_PARAMS plata25);
```

Parameters

plata25

Pointer to an IATA25_PARAMS structure holding the IATA25 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetIATA25

For .Net

Namespace : ImagerNet.Imager

Function : bool SetIATA25(ref IATA25_PARAMS plata25)

Example

```

PIATA25_PARAMS plata25 = new IATA25_PARAMS();
plata25->bEnable = m_bEnable;
plata25->nMinLen = m_nMinLen;
plata25->nMaxLen = m_nMaxLen;
if(IMAGER_SetIATA25(plata25) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_SetIATA25()", NULL, MB_TOPMOST);
delete plata25;

```

4.2.77 IMAGER_SetINT25

Description

Sets the option of INT25 Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetINT25(PINT25_PARAMS pInt25);
```

Parameters

pInt25

Pointer to an INT25_PARAMS structure holding the INT25 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetINT25

For .Net

Namespace : ImagerNet.Imager

Function : bool SetINT25(ref INT25_PARAMS pInt25)

Example

```
PINT25_PARAMS pInt25 = new INT25_PARAMS();
pInt25->bEnable = m_bEnable;
pInt25->bCDV = m_bCDV;
pInt25->bXCD = m_bXCD;
pInt25->nMinLen = m_nMinLen;
pInt25->nMaxLen = m_nMaxLen;
if(IMAGER_SetINT25(pInt25) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetINT25()", NULL, MB_TOPMOST);
delete pInt25;
```

4.2.78 IMAGER_SetKOREAPOST

Description

Sets the option of KOREAPOST Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetKOREAPOST(PKOREAPOST_PARAMS pKoreaPost);
```

Parameters

pKoreaPost

Pointer to a KOREAPOST_PARAMS structure holding the KOREAPOST common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetKOREAPOST

For .Net

Namespace : ImagerNet.Imager

Function : bool SetKOREAPOST(ref KOREAPOST_PARAMS pKoreaPost)

Example

```
PKOREAPOST_PARAMS pKoreapost = new KOREAPOST_PARAMS();  
pKoreapost->bEnable = m_bEnable;  
pKoreapost->nMinLen = m_nMinLen;  
pKoreapost->nMaxLen = m_nMaxLen;  
if(IMAGER_SetKOREAPOST(pKoreapost) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_SetKOREAPOST()", NULL, MB_TOPMOST);  
delete pKoreapost;
```

4.2.79 IMAGER_SetMATRIX25

Description

Sets the option of MATRIX25 Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetMATRIX25(PMATRIX25_PARAMS pMatrix25);

Parameters

pMatrix25

Pointer to a MATRIX25_PARAMS structure holding the MATRIX25 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetMATRIX25

For .Net

Namespace : ImagerNet.Imager

Function : bool SetMATRIX25(ref MATRIX25_PARAMS pMatrix25)

Example

```
PMATRIX25_PARAMS pMatrix25 = new MATRIX25_PARAMS();  
pMatrix25->bEnable = m_bEnable;  
pMatrix25->nMinLen = m_nMinLen;  
pMatrix25->nMaxLen = m_nMaxLen;  
if (IMAGER_SetMATRIX25(pMatrix25) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_SetMATRIX25()", NULL, MB_TOPMOST);  
delete pMatrix25;
```

4.2.80 IMAGER_SetMAXICODE

Description

Sets the option of MAXICODE Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetMAXICODE(PMAXICODE_PARAMS pMaxiCode);

Parameters

pMaxiCode

Pointer to a MAXICODE_PARAMS structure holding the MAXICODE common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetMAXICODE

For .Net

Namespace : ImagerNet.Imager

Function : bool SetMAXICODE(ref MAXICODE_PARAMS pMaxiCode)

Example

```
PMAXICODE_PARAMS pMaxicode= new MAXICODE_PARAMS();  
pMaxicode ->bEnable = m_bEnable;  
pMaxicode ->nMinLen = m_nMinLen;  
pMaxicode ->nMaxLen = m_nMaxLen;
```

```
if(IMAGER_SetMAXICODE(pMaxicode) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_SetMAXICODE()", NULL, MB_TOPMOST);
delete pMaxicode;
```

4.2.81 IMAGER_SetMICROPDF

Description

Sets the option of MICROPDF Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetMICROPDF(PMICROPDF_PARAMS pMicroPdf);
```

Parameters

pMicroPdf

Pointer to a MICROPDF_PARAMS structure holding the MICROPDF common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetMICROPDF

For .Net

Namespace : ImagerNet.Imager

Function : bool SetMICROPDF(ref MICROPDF_PARAMS pMicroPdf)

Example

```
PMICROPDF_PARAMS  pMicropdf = new MICROPDF_PARAMS();
pMicropdf->bEnable = m_bEnable;
pMicropdf->nMinLen = m_nMinLen;
pMicropdf->nMaxLen = m_nMaxLen;
if(IMAGER_SetMICROPDF(pMicropdf) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_SetMICROPDF()", NULL, MB_TOPMOST);
delete pMicropdf;
```

4.2.82 IMAGER_SetMSI

Description

Sets the option of MSI Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetMSI(PMSI_PARAMS pMsi);

Parameters

pMsi

Pointer to a MSI_PARAMS structure holding the MSI common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetMSI

For .Net

Namespace : ImagerNet.Imager

Function : bool SetMSI(ref MSI_PARAMS pMsi)

Example

```
PMSI_PARAMS pMsi = new MSI_PARAMS();
pMsi->bEnable = m_bEnable;
pMsi->bXCD = m_bXCD;
pMsi->nMinLen = m_nMinLen;
pMsi->nMaxLen = m_nMaxLen;
if(IMAGER_SetMSI(pMsi) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetMSI()", NULL, MB_TOPMOST);
delete pMsi;
```

4.2.83 IMAGER_SetOCR

Description

Sets the option of OCR Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetOCR(POCR_PARAMS pOcr);

Parameters

pOcr

Pointer to an OCR_PARAMS structure holding the OCR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetOCR

For .Net

Namespace : ImagerNet.Imager

Function : bool SetOCR(ref OCR_PARAMS pOcr)

Example

```
POCR_PARAMS pOcr = new OCR_PARAMS();
pOcr->bEnable = m_bEnable;
pOcr->nMode = (OCR_MODE)m_ctrlComboMode.GetCurSel();
wsprintf(pOcr->szTemplate, L"%s", m_strEditTemplate);
wsprintf(pOcr->szGroupG, L"%s", m_strEditGroupG);
wsprintf(pOcr->szGroupH, L"%s", m_strEditGroupH);
wsprintf(pOcr->szCheckChar, L"%s", m_strCheckChar);
if(IMAGER_SetOCR(pOcr) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_SetOcr()", NULL, MB_TOPMOST);
delete pOcr;
```

4.2.84 IMAGER_SetOption

Description

Sets the option of imager.

Syntax

```
IMAGER_API BOOL IMAGER_SetOption(PDECODER_PARAMS pOption);
```

Parameters

pOption

Pointer to a DECODER_PARAMS structure holding the imager parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetOption

For .Net

Namespace : ImagerNet.Imager

Function : bool SetOption(ref DECODER_PARAMS pOption)

Example

```
PDECODER_PARAMS pOption = new DECODER_PARAMS();
pOption->nSound = m_nSound;
pOption->bCentering = m_bCentering;
pOption->bVibrate = m_bVibrate;
pOption->bXmitAimID = m_bAimID;
pOption->bContinueMode = m_bContinueMode;
pOption->bHexMode = m_bHexMode;
pOption->nTimeOut = m_ctrlComboTimeOut.GetCurSel() + 1;
pOption->nLightMode = m_ctrlComboLightMode.GetCurSel();
if(IMAGER_SetOption(pOption) == FALSE)
{
    ::MessageBox(NULL, L"ERROR : IMAGER_SetOption()", NULL, MB_TOPMOST);
    return FALSE;
}
delete pOption;
```

4.2.85 IMAGER_SetPDF417

Description

Sets the option of PDF417 Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetPDF417(PPDF417_PARAMS pPdf417);

Parameters

pPdf417

Pointer to a PDF417_PARAMS structure holding the PDF417 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetPDF417

For .Net

Namespace : ImagerNet.Imager

Function : bool SetPDF417(ref PDF417_PARAMS pPdf417)

Example

```
PpPDF417_PARAMS      pPdf417 = new PDF417_PARAMS();  
pPdf417->bEnable = m_bEnable;  
pPdf417->nMinLen = m_nMinLen;  
pPdf417->nMaxLen = m_nMaxLen;  
if(IMAGER_SetPDF417(pPdf417) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_SetPDF417()", NULL, MB_TOPMOST);  
delete pPdf417;
```

4.2.86 IMAGER_SetPLANET

Description

Sets the option of PLANET Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetPLANET(PPLANET_PARAMS pPlanet);

Parameters

pPlanet

Pointer to a PLANET_PARAMS structure holding the PLANET common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetPLANET

For .Net

Namespace : ImagerNet.Imager

Function : bool SetPLANET(ref PLANET_PARAMS pPlanet)

Example

```
PPLANET_PARAMS pPlanet = new PLANET_PARAMS();  
pPlanet->bEnable = m_bEnable;  
pPlanet->bXCD = m_bXCD;  
if(IMAGER_SetPLANET(pPlanet) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_SetPLANET()", NULL, MB_TOPMOST);  
delete pPlanet;
```

4.2.87 IMAGER_SetPLESSEY

Description

Sets the option of PLESSEY Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetPLESSEY(PPLESSEY_PARAMS pPlessey);
```

Parameters

pPlessey

Pointer to a PLESSEY_PARAMS structure holding the PLESSEY common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetPLESSEY

For .Net

Namespace : ImagerNet.Imager

Function : bool SetPLESSEY(ref PLESSEY_PARAMS pPlessey)

Example

```
PPLESSEY_PARAMS      pPlessey = new PLESSEY_PARAMS();  
pPlessey->bEnable = m_bEnable;  
pPlessey->nMinLen = m_nMinLen;  
pPlessey->nMaxLen = m_nMaxLen;  
if(IMAGER_SetPLESSEY(pPlessey) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_SetPLESSEY()", NULL, MB_TOPMOST);  
delete pPlessey;
```

4.2.88 IMAGER_SetPOSICODE

Description

Sets the option of POSICODE Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetPOSICODE(PPOSICODE_PARAMS pPosiCode);
```

Parameters

pPosiCode

Pointer to a POSICODE_PARAMS structure holding the POSICODE common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetPOSI CODE

For .Net

Namespace : ImagerNet.Imager

Function : bool SetPOSI CODE(ref POSICODE_PARAMS pPosiCode)

Example

```
PPOSI CODE_PARAMS pPosicode = new POSICODE_PARAMS();
pPosicode->bEnable = m_bEnable;
pPosicode->bPosi_Lim1 = m_bPosi_Lim1;
pPosicode->bPosi_Lim2 = m_bPosi_Lim2;
pPosicode->nMinLen = m_nMinLen;
pPosicode->nMaxLen = m_nMaxLen;
if(IMAGER_SetPOSI CODE(pPosicode) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetPLANET()", NULL, MB_TOPMOST);
delete pPosicode;
```

4.2.89 IMAGER_SetPOSTNET

Description

Sets the option of POSTNET Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetPOSTNET(PPOSTNET_PARAMS pPostNet);
```

Parameters

pPostNet

Pointer to a POSTNET_PARAMS structure holding the POSTNET common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetPOSTNET

For .Net

Namespace : ImagerNet.Imager

Function : bool SetPOSTNET(ref POSTNET_PARAMS pPostNet)

Example

```
PPOSTNET_PARAMS pPostnet = new POSTNET_PARAMS();
pPostnet->bEnable = m_bEnable;
pPostnet->bXCD = m_bXCD;
if(IMAGER_SetPOSTNET(pPostnet) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetPOSTNET()", NULL, MB_TOPMOST);
delete pPostnet;
```

4.2.90 IMAGER_SetQR

Description

Sets the option of QR Barcode.

Syntax

IMAGER_API BOOL IMAGER_SetQR(PQR_PARAMS pQr);

Parameters

pQr

Pointer to a QR_PARAMS structure holding the QR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetQR

For .Net

Namespace : ImagerNet.Imager

Function : bool SetQR(ref QR_PARAMS pQr)

Example

```
PQR_PARAMS pQr = new QR_PARAMS();
pQr->bEnable = m_bEnable;
pQr->nMinLen = m_nMinLen;
pQr->nMaxLen = m_nMaxLen;
```

```
if(IMAGER_SetQR(pQr) == FALSE)
    ::MessageBox(NULL, L"Error : IMAGER_SetQR()", NULL, MB_TOPMOST);
delete pQr;
```

4.2.91 IMAGER_SetRSS

Description

Sets the option of RSS Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetRSS(PRSS_PARAMS pRss);
```

Parameters

pRss

Pointer to a RSS_PARAMS structure holding the RSS common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetRSS

For .Net

Namespace : ImagerNet.Imager

Function : bool SetRSS(ref RSS_PARAMS pRss)

Example

```
PRSS_PARAMS pRss = new RSS_PARAMS();
pRss->bEnable = m_bEnable;
pRss->bRssLim = m_bRssLim;
pRss->bRssExp = m_bRssExp;
pRss->nMinLen = m_nMinLen;
pRss->nMaxLen = m_nMaxLen;
if(IMAGER_SetRSS(pRss) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetRSS()", NULL, MB_TOPMOST);
delete pRss;
```

4.2.92 IMAGER_SetSTRT25

Description

Sets the option of STRT25 Barcode.

Syntax

```
IMAGER_API BOOL IMAGER_SetSTRT25(PSTRT25_PARAMS pStrt25);
```

Parameters

pStrt25

ex

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetSTRT25

For .Net

Namespace : ImagerNet.Imager

Function : bool SetSTRT25(ref STRT25_PARAMS pStrt25)

Example

```
PSTRT25_PARAMS pStrt25 = new STRT25_PARAMS();  
pStrt25->bEnable = m_bEnable;  
pStrt25->nMinLen = m_nMinLen;  
pStrt25->nMaxLen = m_nMaxLen;  
if(IMAGER_SetSTRT25(pStrt25) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_SetSTRT25()", NULL, MB_TOPMOST);  
delete pStrt25;
```

4.2.93 IMAGER_SetSymbology

Description

Sets the Enable/Disable of Symbologies.

Syntax

```
IMAGER_API BOOL IMAGER_SetSymbology(PDECODER pSymbology);
```

Parameters

pSymbology

Pointer to a DECODER structure holding the symbologies enable or disable.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetSymbology

For .Net

Namespace : ImagerNet.Imager

Function : bool SetSymbology(ref DECODER pSymbology)

Example

```
PDECODER pSym = new DECODER();
BOOLbEnable[47] = {0, };
int i = 0;
for(i=0; i<47; i++)
{
    bEnable[i] = m_ctrlListSymbology.GetCheck(i);
}
pSym->bAZTEC = bEnable[0];
pSym->bCODABAR = bEnable[1];
pSym->bCODE11 = bEnable[2];
pSym->bCODE128 = bEnable[3];
pSym->bCODE39 = bEnable[4];
pSym->bCODE49 = bEnable[5];
pSym->bCODE93 = bEnable[6];
pSym->bCOMPOSITE = bEnable[7];
pSym->bDATAMATRIX = bEnable[8];
pSym->bEAN8 = bEnable[9];
pSym->bEAN13 = bEnable[10];
pSym->bINT25 = bEnable[11];
pSym->bMAXICODE = bEnable[12];
pSym->bMICROPDF = bEnable[13];
pSym->bOCR = bEnable[14];
pSym->bPDF417 = bEnable[15];
pSym->bPOSTNET = bEnable[16];
pSym->bQR = bEnable[17];
pSym->bRSS = bEnable[18];
```



```

pSym->bUPCA = bEnable[19];
pSym->bUPCE = bEnable[20];
pSym->bISBT = bEnable[21];
pSym->bBPO = bEnable[22];
pSym->bCANPOST = bEnable[23];
pSym->bAUSPOST = bEnable[24];
pSym->bIATA25 = bEnable[25];
pSym->bCODABLOCK = bEnable[26];
pSym->bJAPOST = bEnable[27];
pSym->bPLANET = bEnable[28];
pSym->bDUTCHPOST = bEnable[29];
pSym->bMSI = bEnable[30];
pSym->bTLCODE39 = bEnable[31];
pSym->bSTRT25 = bEnable[32];
pSym->bMATRIX25 = bEnable[33];
pSym->bPLESSEY = bEnable[34];
pSym->bCHINAPOST = bEnable[35];
pSym->bKOREAPOST = bEnable[36];
pSym->bTELEPEN = bEnable[37];
pSym->bCODE16K = bEnable[38];
pSym->bPOSI CODE = bEnable[39];
pSym->bCOUPONCODE = bEnable[40];
pSym->bUSPS4CB = bEnable[41];
pSym->bIDTAG = bEnable[42];
pSym->bLABEL = bEnable[43];
pSym->bGS1_128 = bEnable[44];
pSym->bHANXIN = bEnable[45];
pSym->bGRIDMATRIX = bEnable[46];
if(IMAGER_SetSymbology(pSym) == FALSE)
    ::MessageBox(NULL, L"ERROR : IMAGER_SetSymbology()", NULL, MB_TOPMOST);
delete pSym;

```

4.2.94 IMAGER_SetSymbologyAll

Description

Enables all of Symbologies.

Syntax

```
IMAGER_API BOOL IMAGER_SetSymbologyAll();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetSymbologyDefault

For .Net

Namespace : ImagerNet.Imager

Function : bool SetSymbologyAll()

Example

None

4.2.95 IMAGER_SetSymbologyDefault

Description

Initializes all option of scanner.

Syntax

```
IMAGER_API BOOL IMAGER_SetSymbologyDefault();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_SetSymbologyAll

For .Net

Namespace : ImagerNet.Imager

Function : bool SetSymbologyDefault()

Example

None

4.2.96 IMAGER_SetTELEPEN

Description

Sets the option of TELEPEN Barcode

Syntax

```
IMAGER_API BOOL IMAGER_SetTELEPEN(PTELEPEN_PARAMS pTelepen);
```

Parameters

pTelepen

Pointer to a TELEPEN_PARAMS structure holding the TELEPEN common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetTELEPEN

For .Net

Namespace : ImagerNet.Imager

Function : bool SetTELEPEN(ref TELEPEN_PARAMS pTelepen)

Example

```
PTELEPEN_PARAMS pTelepen = new TELEPEN_PARAMS();  
pTelepen->bEnable = m_bEnable;  
pTelepen->bNumeric = m_bNumeric;  
pTelepen->nMinLen = m_nMinLen;  
pTelepen->nMaxLen = m_nMaxLen;  
if(IMAGER_SetTELEPEN(pTelepen) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_SetTELEPEN()", NULL, MB_TOPMOST);  
delete pTelepen;
```

4.2.97 IMAGER_SetUPCA

Description

Sets the option of UPC-A Barcode

Syntax

```
IMAGER_API BOOL IMAGER_SetUPCA(PUPCA_PARAMS pUpca);
```

Parameters

pUpca

Pointer to a UPCA_PARAMS structure holding the UPC-A common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetUPCA

For .Net

Namespace : ImagerNet.Imager

Function : bool SetUPCA(ref UPCA_PARAMS pUpca)

Example

```
PUPCA_PARAMS pUpca = new UPCA_PARAMS();
pUpca->bEnable = m_bEnable;
pUpca->bXCD = m_bXCD;
pUpca->bXNum = m_bXNum;
pUpca->bAddOn = m_bAddOn;
if(IMAGER_SetUPCA(pUpca) == FALSE)
    ::MessageBox(NULL, L"Error : SCAN_SetUPCA()", NULL, MB_TOPMOST);
delete pUpca;
```

4.2.98 IMAGER_SetUPCE

Description

Sets the option of UPC-E Barcode

Syntax

IMAGER_API BOOL IMAGER_SetUPCE(PUPCE_PARAMS pUpce);

Parameters

pUpce

Pointer to a UPCE_PARAMS structure holding the UPC-E common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

IMAGER_GetUPCE

For .Net

Namespace : ImagerNet.Imager

Function : bool SetUPCE(ref UPCE_PARAMS pUpce)

Example

```
PUPCE_PARAMS pUpce = new UPCE_PARAMS();  
pUpce->bEnable = m_bEnable;  
pUpce->bXCD = m_bXCD;  
pUpce->bXNum = m_bXNum;  
pUpce->bAddOn = m_bAddOn;  
if(IMAGER_SetUPCE(pUpce) == FALSE)  
    ::MessageBox(NULL, L"Error : SCAN_SetUPCE()", NULL, MB_TOPMOST);  
delete pUpce;
```

4.3 LRSCANNER (Long-Range)

Status Return value

Please refer to the below table for the status value definition.

Status Value Definition	Code	Meaning
TRUE	1	Success
FALSE	0	General Error

Define
#define WM_SCAN_DATA WM_APP + 350
Enum
<pre>typedef enum { SOUND_DEFAULT = 0, SOUND_BEEP, SOUND_NONE } SCAN_SOUND; typedef enum { DEVICE_M3SKY = 0, DEVICE_M3SKYSAM, DEVICE_MM3, DEVICE_M3ORANGE, DEVICE_M3SMART_CE, DEVICE_M3SMART_WM, DEVICE_M3GREEN, DEVICE_M3T, DEVICE_M3POS, DEVICE_M3ORANGEPLUS, DEVICE_M3ORANGEPLUS_CE, DEVICE_M3BLACK, DEVICE_M3UL10_CE, DEVICE_UNKNOWN } SCAN_DEVICE_TYPE;</pre>
Structure
<pre>typedef struct _DECODER{ BYTE bAUSPOST; BYTE bAZTEC; BYTE bBPO; BYTE bCANADAPOST; BYTE bCODABAR; BYTE bCODABLOCK; BYTE bCODE11;</pre>

```

BYTE bCODE39;
BYTE bCODE93;
BYTE bCODE128;
BYTE bDATAMATRIX;
BYTE bDUTCHPOST;
BYTE bUPCA;
BYTE bUPCE;
BYTE bEAN8;
BYTE bEAN13;
BYTE bGS1_COMPOSITE;
BYTE bGS1_DATABAR;
BYTE bINFOMAIL;
BYTE bINT25;
BYTE bJAPANPOST;
BYTE bMATRIX2OF5;
BYTE bMAXICODE;
BYTE bMSI;
BYTE bPDF417;
BYTE bMICRO_PDF417;
BYTE bPLANET;
BYTE bPLESSEY;
BYTE bPOSTNET;
BYTE bQRCODE;
BYTE bSTANDARD2OF5;
BYTE bSWEDENPOST;
BYTE bTELEPEN;
BYTE bTLC39;
}DECODER, *PDECODER;
typedef struct _DECODER_PARAMS{
    BYTE bContinueMode;
    BYTE bXmitAimID;
    BYTE bVibrate;
    BYTE b1DDecodeMode;
    BYTE bCenterDecode;
    int  nSound;
    int  nTimeOut;
    int  nSecurityLevel;
}DECODER_PARAMS, *PDECODER_PARAMS;
typedef struct _CODABAR_PARAMS{
    BYTE  bEnable;
    BYTE  bXSS;
    BYTE  bCDV;
    BYTE  bXCD;
}CODABAR_PARAMS, *PCODABAR_PARAMS;
typedef struct _CODABLOCK_PARAMS{
    BYTE  bEnable;

```

```

    BYTE    bCodaBlock_F;
}CODABLOCK_PARAMS, *PCODABLOCK_PARAMS;

typedef struct _CODE11_PARAMS{
    BYTE    bEnable;
    BYTE    bCDV;
    BYTE    bXCD;
}CODE11_PARAMS, *PCODE11_PARAMS;

typedef struct _CODE39_PARAMS{
    BYTE    bEnable;
    BYTE    bCDV;
    BYTE    bXCD;
    BYTE    bFullASCII;
}CODE39_PARAMS, *PCODE39_PARAMS;

typedef struct _CODE128_PARAMS{
    BYTE    bEnable;
    BYTE    bGs1_128;
    BYTE    bIsbt128;
}CODE128_PARAMS, *PCODE128_PARAMS;

typedef struct _UPCA_PARAMS{
    BYTE    bEnable;
    BYTE    bXNum;
    BYTE    bXCD;
    BYTE    bAddOn;
    BYTE    bUPCA_AS_EAN13;
}UPCA_PARAMS, *PUPCA_PARAMS;

typedef struct _UPCE_PARAMS{
    BYTE    bEnable;
    BYTE    bXNum;
    BYTE    bXCD;
    BYTE    bUPCE_AS_UPCA;
}UPCE_PARAMS, *PUPCE_PARAMS;

typedef struct _EAN8_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
    BYTE    bEAN8_AS_EAN13;
}EAN8_PARAMS, *PEAN8_PARAMS;

typedef struct _EAN13_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
    BYTE    bAddOn;
    BYTE    bIsxN;
}EAN13_PARAMS, *PEAN13_PARAMS;

typedef struct _GS1COMPOSITE_PARAMS{
    BYTE    bEnable;
    BYTE    bCC_C;
}GS1COMPOSITE_PARAMS, *PGS1COMPOSITE_PARAMS;

```



```
typedef struct _GS1DATABAR_PARAMS{
    BYTE    bEnable;
    BYTE    bGS1LIM;
    BYTE    bGS1EXP;
}GS1DATABAR_PARAMS, *PGS1DATABAR_PARAMS;

typedef struct _INT25_PARAMS{
    BYTE    bEnable;
    BYTE    bCDV;
    BYTE    bXCD;
}INT25_PARAMS, *PINT25_PARAMS;

typedef struct _MSI_PARAMS{
    BYTE    bEnable;
    BYTE    bCDV;
    BYTE    bXCD;
}MSI_PARAMS, *PMSI_PARAMS;

typedef struct _PLESSEY_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
}PLESSEY_PARAMS, *PPLESSEY_PARAMS;

typedef struct _POSTNET_PARAMS{
    BYTE    bEnable;
    BYTE    bXCD;
}POSTNET_PARAMS, *PPOSTNET_PARAMS;

typedef struct _STANDARD2OF5_PARAMS{
    BYTE    bEnable;
    BYTE    bCDV;
    BYTE    bXCD;
}STANDARD2OF5_PARAMS, *PSTANDARD2OF5_PARAMS;

typedef struct _TELEPEN_PARAMS{
    BYTE    bEnable;
    BYTE    bNumeric;
}TELEPEN_PARAMS, *PTELEPEN_PARAMS;
```

Functions for LRScanner

Name	Description
LRSCAN_Close	Closes an open scanner
LRSCAN_GetCODABAR	Gets the option of CODABAR Barcode
LRSCAN_GetCODABLOCK	Gets the option of CODABLOCK Barcode
LRSCAN_GetCODE11	Gets the option of CODE11 Barcode
LRSCAN_GetCODE128	Gets the option of CODE128 Barcode
LRSCAN_GetCODE39	Gets the option of CODE39 Barcode
LRSCAN_GetDeviceType	Get the type of device
LRSCAN_GetEAN13	Gets the option of EAN-13 Barcode
LRSCAN_GetEAN8	Gets the option of EAN-8 Barcode
LRSCAN_GetGS1COMPOSITE	Gets the option of GS1COMPOSITE Barcode
LRSCAN_GetGS1DATABAR	Gets the option of GS1DATABAR Barcode
LRSCAN_GetiINT25	Gets the option of INT25 Barcode
LRSCAN_GetMSI	Gets the option of MSI Barcode
LRSCAN_GetOption	Gets the option of Scanner
LRSCAN_GetPLESSEY	Gets the option of PLESSEY Barcode
LRSCAN_GetPOSTNET	Gets the option of POSTNET Barcode
LRSCAN_GetScanData	Gets the ScanData which is read by scanner
LRSCAN_GetSTANDARD2OF5	Gets the option of STANDARD2OF5 Barcode
LRSCAN_GetSymbology	Gets Enable/Disable of Symbologies
LRSCAN_GetTELEPEN	Gets the option of TELEPEN Barcode
LRSCAN_GetUPCA	Gets the option of UPC-A Barcode
LRSCAN_GetUPCE	Gets the option of UPC-E Barcode
LRSCAN_GetVersionInfo	Gets the information of scanner engine and dll version
LRSCAN_Open	Opens a scanner
LRSCAN_Read	Starts the beaming of scanner
LRSCAN_ReadCancel	Stops the beaming of scanner
LRSCAN_SetCODABAR	Gets the option of CODABAR Barcode
LRSCAN_SetCODABLOCK	Gets the option of CODABLOCK Barcode
LRSCAN_SetCODE11	Sets the option of CODE11 Barcode
LRSCAN_SetCODE128	Sets the option of CODE128 Barcode

LRSCAN_SetCODE39	Gets the option of CODE39 Barcode
LRSCAN_SetEAN13	Gets the option of EAN-13 Barcode
LRSCAN_SetEAN8	Gets the option of EAN-8 Barcode
LRSCAN_SetGS1COMPOSITE	Gets the option of GS1COMPOSITE Barcode
LRSCAN_SetGS1DATABAR	Gets the option of GS1DATABAR Barcode
LRSCAN_SetINT25	Gets the option of INT25 Barcode
LRSCAN_SetMSI	Sets the option of Scanner
LRSCAN_SetOption	Sets the option of Scanner
LRSCAN_SetPLESSEY	Gets the option of PLESSEY Barcode
LRSCAN_SetPOSTNET	Gets the option of POSTNET Barcode
LRSCAN_SetSTANDARD2OF5	Gets the option of STANDARD2OF5 Barcode
LRSCAN_SetSymbology	Sets the Enable/Disable of Symbologies
LRSCAN_SetSymbologyAll	Enables all of Symbologies
LRSCAN_SetSymbologyDefault	Initializes all option of scanner
LRSCAN_SetTELEPEN	Sets the option of TELEPEN Barcode
LRSCAN_SetUPCA	Sets the option of UPC-A Barcode
LRSCAN_SetUPCE	Sets the option of UPC-E Barcode

4.3.1 LRSCAN_Close

Description

Closes an open scanner.

Syntax

```
LRSCANNER_API BOOL LRSCAN_Close();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_Open

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool Close()

Example

```
if(LRSCAN_Close() == FALSE)
{
    ::MessageBox(NULL, L"Error : LRSCAN_Close()", NULL, MB_TOPMOST);
}
```

4.3.2 LRSCAN_GetCODABAR

Description

Gets the option of CODABAR Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetCODABAR(PCODABAR_PARAMS pCodabar);
```

Parameters

pCodabar

Pointer to a CODABAR_PARAMS structure to be filled in with the CODABAR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetCODABAR

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetCODABAR(out CODABAR_PARAMS pCodabar)

Example

```
PCODABAR_PARAMS  pCodabar = new CODABAR_PARAMS();
if(LRSCAN_GetCODABAR(pCodabar) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_GetCODABAR()", NULL, MB_TOPMOST);

m_bEnable = pCodabar->bEnable;
m_Bcdv = pCodabar->bCDV;
m_bXCD = pCodabar->bXCD;
m_bXSS = pCodabar->bXSS;
delete pCodabar;
```

4.3.3 LRSCAN_GetCODABLOCK

Description

Gets the option of CODABLOCK Barcode.

Syntax

LRSCANNER_API BOOL LRSCAN_GetCODABLOCK(PCODABLOCK_PARAMS pCodablock);

Parameters

pCodablock

Pointer to a CODABLOCK_PARAMS structure to be filled in with the CODABLOCK common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetCODABLOCK

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetCODABLOCK(out CODABLOCK_PARAMS pCodablock)

Example

```
PCODABLOCK_PARAMS  pCodablock = new CODABLOCK_PARAMS();
```

```

if(LRSCAN_GetCODABLOCK(pCodablock) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_GetCODABLOCK()", NULL, MB_TOPMOST);
m_bEnable = pCodablock->bEnable;
m_bCodablockF = pCodablock->bCodaBlock_F;
delete pCodablock;

```

4.3.4 LRSCAN_GetCODE11

Description

Gets the option of CODE11 Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetCODE11(PCODE11_PARAMS pCode11);
```

Parameters

pCode11

Pointer to a CODE11_PARAMS structure to be filled in with the CODE11 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetCODE11

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetCODE11(out CODE11_PARAMS pCode11)

Example

```

PCODE11_PARAMS pCode11 = new CODE11_PARAMS();
if(LRSCAN_GetCODE11(pCode11) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_GetCODE11()", NULL, MB_TOPMOST);
m_bEnable = pCode11->bEnable;
m_nCDV = pCode11->bCDV;
m_bXCD = pCode11->bXCD;
delete pCode11;

```

4.3.5 LRSCAN_GetCODE128

Description

Gets the option of CODE128 Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetCODE128(PCODE128_PARAMS pCode128);
```

Parameters

pCode128

Pointer to a CODE128_PARAMS structure to be filled in with the CODE128 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetCODE128

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetCODE128(out CODE128_PARAMS pCode128)

Example

```
PCODE128_PARAMS pCode128 = new CODE128_PARAMS();
if(LRSCAN_GetCODE128(pCode128) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_GetCODE128()", NULL, MB_TOPMOST);
m_bEnable = pCode128->bEnable;
m_bGs1128 = pCode128->bGs1_128;
m_blsbt128 = pCode128->blsbt128;
delete pCode128;
```

4.3.6 LRSCAN_GetCODE39

Description

Gets the option of CODE39 Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetCODE39(PCODE39_PARAMS pCode39);
```

Parameters

pCode39

Pointer to a CODE39_PARAMS structure to be filled in with the CODE39 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetCODE39

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetCODE39(out CODE39_PARAMS pCode39)

Example

```
PCODE39_PARAMS pCode39 = new CODE39_PARAMS();  
if(LRSCAN_GetCODE39(pCode39) == FALSE)  
    ::MessageBox(NULL, L"Error : IMAGER_GetCODE39()", NULL, MB_TOPMOST);  
m_bEnable = pCode39->bEnable;  
m_bCDV = pCode39->bCDV;  
m_bXCD = pCode39->bXCD;  
m_bFullASCII = pCode39->bFullASCII;  
delete pCode39;
```

4.3.7 LRSCAN_GetDeviceType

Description

Gets the type of device.

Syntax

```
LRSCANNER_API SCAN_DEVICE_TYPE LRSCAN_GetDeviceType();
```

Parameters

None

Return Value

The return value is SCAN_DEVICE_TYPE.

Remarks

LRSCAN_GetDeviceType is only used after LRSCAN_OPEN.

See Also

None

For .Net

Namespace : LRScannerNet.LRScanner

Function : SCAN_DEVICE_TYPE GetDeviceType()

Example

None

4.3.8 LRSCAN_GetEAN13

Description

Gets the type of device.

Syntax

```
LRSCANNER_API SCAN_DEVICE_TYPE LRSCAN_GetDeviceType();
```

Parameters

None

Return Value

The return value is SCAN_DEVICE_TYPE.

Remarks

LRSCAN_GetDeviceType is only used after LRSCAN_OPEN.

See Also

None

For .Net

Namespace : LRScannerNet.LRScanner

Function : SCAN_DEVICE_TYPE GetDeviceType()

Example

None

4.3.9 LRSCAN_GetEAN13

Description

Gets the option of EAN-13 Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetEAN13(PEAN13_PARAMS pEan13);
```

Parameters

pEan13

Pointer to an EAN13_PARAMS structure to be filled in with the EAN-13 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetEAN13

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetEAN13(out EAN13_PARAMS pEan13)

Example

```
PEAN13_PARAMS pEan13 = new EAN13_PARAMS();  
if(LRSCAN_GetEAN13(pEan13) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_GetEAN13()", NULL, MB_TOPMOST);  
m_bEnable = pEan13->bEnable;  
m_bIsxn = pEan13->bIsxn;  
m_bXCD = pEan13->bXCD;  
m_bAddOn = pEan13->bAddOn;  
delete pEan13;
```

4.3.10 LRSCAN_GetEAN8

Description

Gets the option of EAN-8 Barcode.

Syntax

LRSCANNER_API BOOL LRSCAN_GetEAN8(PEAN8_PARAMS pEan8);

Parameters

pEan8

Pointer to a EAN8_PARAMS structure to be filled in with the EAN-8 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetEAN8

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetEAN8(out EAN8_PARAMS pEan8)

Example

```
PEAN8_PARAMS pEan8 = new EAN8_PARAMS();  
if(LRSCAN_GetEAN8(pEan8) == FALSE)
```

```

        ::MessageBox(NULL, L"Error : LRSCAN_GetEAN8()", NULL, MB_TOPMOST);
m_bEnable = pEan8->bEnable;
m_bXCD = pEan8->bXCD;
m_bEan8AsEan13 = pEan8->bEAN8_AS_EAN13;
delete pEan8;

```

4.3.11 LRSCAN_GetGS1COMPOSITE

Description

Gets the option of GS1COMPOSITE Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetGS1COMPOSITE(PGS1COMPOSITE_PARAMS pGs1Composite);
```

Parameters

pGs1Composite

Pointer to a GS1COMPOSITE_PARAMS structure to be filled in with the GS1COMPOSITE common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetGS1COMPOSITE

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetGS1COMPOSITE(out GS1COMPOSITE_PARAMS pGs1Composite)

Example

```

PGS1COMPOSITE_PARAMS  pGs1Composite = new GS1COMPOSITE_PARAMS();
if(LRSCAN_GetGS1COMPOSITE(pGs1Composite) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_GetGS1COMPOSITE()", NULL, MB_TOPMOST);
m_bEnable = pGs1Composite->bEnable;
m_bCC_C = pGs1Composite->bCC_C;
delete pGs1Composite;

```

4.3.12 LRSCAN_GetGS1DATABAR

Description

Gets the option of GS1DATABAR Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetGS1DATABAR(PGS1DATABAR_PARAMS pGs1Databar);
```

Parameters

pGs1Databar

Pointer to a GS1DATABAR_PARAMS structure to be filled in with the GS1DATABAR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetGS1DATABAR

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetGS1DATABAR(out GS1DATABAR_PARAMS pGs1Databar)

Example

```
PGS1DATABAR_PARAMS pGs1Databar = new GS1DATABAR_PARAMS();  
if(LRSCAN_GetGS1DATABAR(pGs1Databar) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_GetGS1DATABAR()", NULL, MB_TOPMOST);  
m_bEnable = pGs1Databar->bEnable;  
m_bGs1Lim = pGs1Databar->bGS1LIM;  
m_bGs1Exp = pGs1Databar->bGS1EXP;  
delete pGs1Databar;
```

4.3.13 LRSCAN_GetINT25

Description

Gets the option of INT25 Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetINT25(PINT25_PARAMS pInt25);
```

Parameters

pInt25

Pointer to an INT25_PARAMS structure to be filled in with the INT25 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetINT25

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetINT25(out INT25_PARAMS pInt25)

Example

```
PINT25_PARAMS pInt25 = new INT25_PARAMS();
if(LRSCAN_GetINT25(pInt25) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_GetINT25()", NULL, MB_TOPMOST);
m_bEnable = pInt25->bEnable;
m_bCDV = pInt25->bCDV;
m_bXCD = pInt25->bXCD;
delete pInt25;
```

4.3.14 LRSCAN_GetMSI

Description

Gets the option of MSI Barcode.

Syntax

LRSCANNER_API BOOL LRSCAN_GetMSI(PMSI_PARAMS pMsi);

Parameters

pMsi

Pointer to a MSI_PARAMS structure to be filled in with the MSI common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetMSI

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetMSI(out MSI_PARAMS pMsi)

Example

```
PMSI_PARAMS pMsi = new MSI_PARAMS();
```

```

if(LRSCAN_GetMSI(pMsi) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_GetMSI()", NULL, MB_TOPMOST);
m_bEnable = pMsi->bEnable;
m_nCDV = pMsi->bCDV;
m_bXCD = pMsi->bXCD;
delete pMsi;

```

4.3.15 LRSCAN_GetOption

Description

Gets the option of Scanner.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetOption(PDECODER_PARAMS pOption);
```

Parameters

pOption

Pointer to a DECODER_PARAMS structure to be filled in with the scanner parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetOption

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetOption(out DECODER_PARAMS pOption)

Example

```

PDECODER_PARAMS pOption = new DECODER_PARAMS();
if(LRSCAN_GetOption(pOption) == FALSE)
    ::MessageBox(NULL, L"ERROR : LRSCAN_GetOption()", NULL, MB_TOPMOST);
m_nSound = pOption->nSound;
m_bContinueMode = pOption->bContinueMode;
m_bAimID = pOption->bXmitAimID;
m_bVibrate = pOption->bVibrate;
m_b1DDecode = pOption->b1DDecodeMode;
m_bCenterDecode = pOption->bCenterDecode;

```

```
m_ctrlComboTimeOut.SetCurSel(pOption->nTimeOut - 1);  
m_ctrlComboSecurityLevel.SetCurSel(pOption->nSecurityLevel - 1);  
delete pOption;
```

4.3.16 LRSCAN_GetPLESSEY

Description

Gets the option of PLESSEY Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetPLESSEY(PPLESSEY_PARAMS pPlessey);
```

Parameters

pPlessey

Pointer to a PLESSEY_PARAMS structure to be filled in with the PLESSEY common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetPLESSEY

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetPLESSEY(out PLESSEY_PARAMS pPlessey)

Example

```
PPLESSEY_PARAMS pPlessey = new PLESSEY_PARAMS();  
if(LRSCAN_GetPLESSEY(pPlessey) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_GetPLESSEY()", NULL, MB_TOPMOST);  
m_bEnable = pPlessey->bEnable;  
m_bXCD = pPlessey->bXCD;  
delete pPlessey;
```

4.3.17 LRSCAN_GetPOSTNET

Description

Gets the option of POSTNET Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetPOSTNET(PPOSTNET_PARAMS pPostNet);
```

Parameters

pPostNet

Pointer to a POSTNET_PARAMS structure to be filled in with the POSTNET common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetPOSTNET

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetPOSTNET(out POSTNET_PARAMS pPostNet)

Example

```
PPOSTNET_PARAMS pPostnet = new POSTNET_PARAMS();
if(LRSCAN_GetPOSTNET(pPostnet) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_GetPOSTNET()", NULL, MB_TOPMOST);
m_bEnable = pPostnet->bEnable;
m_bXCD = pPostnet->bXCD;
delete pPostnet;
```

4.3.18 LRSCAN_GetScanData

Description

Gets the ScanData which is read by scanner.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetScanData(TCHAR *pszBarType, TCHAR *pszBarData);
```

Parameters

pszBarType

Pointer to barcode type.

pszBarData

Pointer to barcode data

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetScanData(StringBuilder szBarType, StringBuilder szBarData)

Example

```
TCHAR  szBarType[1024] = {0, };
TCHAR  szBarData[1024] = {0, };
LRSCAN_GetScanData(szBarType, szBarData);
```

4.3.19 LRSCAN_GetSTANDARD2OF5

Description

Gets the option of STANDARD2OF5 Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetSTANDARD2OF5(PSTANDARD2OF5_PARAMS pStandard2of5);
```

Parameters

pStandard2of5

Pointer to a STANDARD2OF5_PARAMS structure to be filled in with the STANDARD2OF5 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetSTANDARD2OF5

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetSTANDARD2OF5(out STANDARD2OF5_PARAMS pStandard2of5)

Example

```
PSTANDARD2OF5_PARAMS pStandard2of5 = new STANDARD2OF5_PARAMS();
if(LRSCAN_GetSTANDARD2OF5(pStandard2of5) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_GetSTANDARD2OF5()", NULL, MB_TOPMOST);
m_bEnable = pStandard2of5->bEnable;
m_bCDV = pStandard2of5->bCDV;
m_bXCD = pStandard2of5->bXCD;
```

delete pStandard2of5;

4.3.20 LRSCAN_GetSymbology

Description

Gets Enable/Disable of Symbologies.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetSymbology(PDECODER pSym);
```

Parameters

pSymbology

Pointer to a DECODER structure to be filled in with the symbologies enable or disable.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetSymbology

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetSymbology(out DECODER pSymbology)

Example

```
PDECODER pSym = new DECODER();
BOOL bEnable[34] = {0, };
int i = 0;
if(LRSCAN_GetSymbology(pSym) == FALSE)
    ::MessageBox(NULL, L"ERROR : LRSCAN_GetSymbology()", NULL, MB_TOPMOST);
bEnable[0] = pSym->bAUSPOST;
bEnable[1] = pSym->bAZTEC;
bEnable[2] = pSym->bBPO;
bEnable[3] = pSym->bCANADAPOST;
bEnable[4] = pSym->bCODABAR;
bEnable[5] = pSym->bCODABLOCK;
bEnable[6] = pSym->bCODE11;
bEnable[7] = pSym->bCODE39;
bEnable[8] = pSym->bCODE93;
```

```

bEnable[9] = pSym->bCODE128;
bEnable[10] = pSym->bDATAMATRIX;
bEnable[11] = pSym->bDUTCHPOST;
bEnable[12] = pSym->bUPCA;
bEnable[13] = pSym->bUPCE;
bEnable[14] = pSym->bEAN8;
bEnable[15] = pSym->bEAN13;
bEnable[16] = pSym->bGS1_COMPOSITE;
bEnable[17] = pSym->bGS1_DATABAR;
bEnable[18] = pSym->bINFOMAIL;
bEnable[19] = pSym->bINT25;
bEnable[20] = pSym->bJAPANPOST;
bEnable[21] = pSym->bMATRIX2OF5;
bEnable[22] = pSym->bMAXICODE;
bEnable[23] = pSym->bMSI;
bEnable[24] = pSym->bPDF417;
bEnable[25] = pSym->bMICRO_PDF417;
bEnable[26] = pSym->bPLANET;
bEnable[27] = pSym->bPLESSEY;
bEnable[28] = pSym->bPOSTNET;
bEnable[29] = pSym->bQRCODE;
bEnable[30] = pSym->bSTANDARD2OF5;
bEnable[31] = pSym->bSWEDENPOST;
bEnable[32] = pSym->bTELEPEN;
bEnable[33] = pSym->bTLC39;
for(i=0; i<34; i++)
{
    m_ctrlListSymbology.SetCheck(i, bEnable[i]);
}
delete pSym;

```

4.3.21 LRSCAN_GetTELEPEN

Description

Gets the option of TELEPEN Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetTELEPEN(PTELEPEN_PARAMS pTelepen);
```

Parameters

pTelepen

Pointer to a TELEPEN_PARAMS structure to be filled in with the TELEPEN common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetTELEPEN

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetTELEPEN(out TELEPEN_PARAMS pTelepen)

Example

```
PTELEPEN_PARAMS pTelepen = new TELEPEN_PARAMS();  
if(LRSCAN_GetTELEPEN(pTelepen) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_GetTELEPEN()", NULL, MB_TOPMOST);  
m_bEnable = pTelepen->bEnable;  
m_bNumeric = pTelepen->bNumeric;  
delete pTelepen;
```

4.3.22 LRSCAN_GetUPCA

Description

Gets the option of UPC-A Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetUPCA(PUPCA_PARAMS pUpca);
```

Parameters

pUpca

Pointer to a UPCA_PARAMS structure to be filled in with the UPC-A common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetUPCA

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetUPCA(out UPCA_PARAMS pUpca)

Example

```
PUPCA_PARAMS pUpca = new UPCA_PARAMS();  
if(LRSCAN_GetUPCA(pUpca) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_GetUPCA()", NULL, MB_TOPMOST);  
m_bEnable = pUpca->bEnable;  
m_bXCD = pUpca->bXCD;  
m_bXNum = pUpca->bXNum;  
m_bUpcaAsEan13 = pUpca->bUPCA_AS_EAN13;  
m_bAddOn = pUpca->bAddOn;  
delete pUpca;
```

4.3.23 LRSCAN_GetUPCE

Description

Gets the option of UPC-E Barcode.

Syntax

LRSCANNER_API BOOL LRSCAN_GetUPCE(PUPCE_PARAMS pUpce);

Parameters

pUpce

Pointer to a UPCE_PARAMS structure to be filled in with the UPC-E common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetUPCE

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool GetUPCE(out UPCE_PARAMS pUpce)

Example

```

PUPCE_PARAMS pUpce = new UPCE_PARAMS();
if(LRSCAN_GetUPCE(pUpce) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_GetUPCE()", NULL, MB_TOPMOST);
m_bEnable = pUpce->bEnable;
m_bXCD = pUpce->bXCD;
m_bXNum = pUpce->bXNum;
m_bUpceAsUpca = pUpce->bUPCE_AS_UPCA;
delete pUpce;

```

4.3.24 LRSCAN_GetVersionInfo

Description

Gets the information of scanner engine and dll version.

Syntax

```
LRSCANNER_API BOOL LRSCAN_GetVersionInfo(TCHAR* pszVersion);
```

Parameters

pszVersion

Pointer to a TCHAR to be filled in with the version info.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : LRScannerNet.LRScanner

Function : string GetVersionInfo()

Example

```
TCHAR szVersionInfo[1024] = {0, };
```

```
SCAN_GetVersionInfo(szVersionInfo);
```

4.3.25 LRSCAN_Open

Description

Opens a scanner.

Syntax

LRSCANNER_API BOOL LRSCAN_Open();

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_Close

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool Open()

Example

```
if(LRSCAN_Open() == FALSE)
{
    ::MessageBox(NULL, L"Error : LRSCAN_Open()", NULL, MB_TOPMOST);
}
```

4.3.26 LRSCAN_Read

Description

Starts the beaming of scanner.

Syntax

LRSCANNER_API BOOL LRSCAN_Read();

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_ReadCancel

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool Read()

Example

None

4.3.27 LRSCAN_ReadCancel

Description

Stops the beaming of scanner.

Syntax

```
LRSCANNER_API BOOL LRSCAN_ReadCancel();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_Read

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool ReadCancel()

Example

None

4.3.28 LRSCAN_SetCODABAR

Description

Sets the option of CODABAR Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetCODABAR(PCODABAR_PARAMS pCodabar);
```

Parameters

pCodabar

Pointer to a CODABAR_PARAMS structure holding the CODABAR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetCODABAR

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetCODABAR(ref CODABAR_PARAMS pCodabar)

Example

```
PCODABAR_PARAMS  pCodabar = new CODABAR_PARAMS();
pCodabar->bEnable = m_bEnable;
pCodabar->bCDV = m_bCDV;
pCodabar->bXCD = m_bXCD;
pCodabar->bXSS = m_bXSS;
if(LRSCAN_SetCODABAR(pCodabar) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_SetCODABAR()", NULL, MB_TOPMOST);
delete pCodabar;
```

4.3.29 LRSCAN_SetCODABLOCK

Description

Sets the option of CODABLOCK Barcode.

Syntax

LRSCANNER_API BOOL LRSCAN_SetCODABLOCK(PCODABLOCK_PARAMS pCodablock);

Parameters

pCodablock

Pointer to a CODABLOCK_PARAMS structure holding the CODABLOCK common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetCODABLOCK

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetCODABLOCK(ref CODABLOCK_PARAMS pCodablock)

Example

```
PCODABLOCK_PARAMS  pCodablock = new CODABLOCK_PARAMS();
```

```

pCodablock->bEnable = m_bEnable;
pCodablock->bCodaBlock_F = m_bCodablockF;
if(LRSCAN_SetCODABLOCK(pCodablock) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_SetCODABLOCK()", NULL, MB_TOPMOST);
delete pCodablock;

```

4.3.30 LRSCAN_SetCODE11

Description

Sets the option of CODE11 Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetCODE11(PCODE11_PARAMS pCode11);
```

Parameters

pCode11

Pointer to a CODE11_PARAMS structure holding the CODE11 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetCODE11

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetCODE11(ref CODE11_PARAMS pCode11)

Example

```

PCODE11_PARAMS pCode11 = new CODE11_PARAMS();
pCode11->bEnable = m_bEnable;
pCode11->bCDV = m_nCDV;
pCode11->bXCD = m_bXCD;
if(LRSCAN_SetCODE11(pCode11) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_SetCODE11()", NULL, MB_TOPMOST);
delete pCode11;

```

4.3.31 LRSCAN_SetCODE128

Description

Sets the option of CODE128 Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetCODE128(PCODE128_PARAMS pCode128);
```

Parameters

pCode128

Pointer to a CODE128_PARAMS structure holding the CODE128 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetCODE128

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetCODE128(ref CODE128_PARAMS pCode128)

Example

```
PCODE128_PARAMS pCode128 = new CODE128_PARAMS();  
pCode128->bEnable = m_bEnable;  
pCode128->bGs1_128 = m_bGs1128;  
pCode128->blsbt128 = m_blsbt128;  
if(LRSCAN_SetCODE128(pCode128) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_GetCODE128()", NULL, MB_TOPMOST);  
delete pCode128;
```

4.3.32 LRSCAN_SetCODE39

Description

Sets the option of CODE39 Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetCODE39(PCODE39_PARAMS pCode39);
```

Parameters

pCode39

Pointer to a CODE39_PARAMS structure holding the CODE39 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetCODE39

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetCODE39(ref CODE39_PARAMS pCode39)

Example

```
PCODE39_PARAMS pCode39 = new CODE39_PARAMS();  
pCode39->bEnable = m_bEnable;  
pCode39->bCDV = m_bCDV;  
pCode39->bXCD = m_bXCD;  
pCode39->bFullASCII = m_bFullASCII;  
if(LRSCAN_SetCODE39(pCode39) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_GetCODE39()", NULL, MB_TOPMOST);  
delete pCode39;
```

4.3.33 LRSCAN_SetEAN13

Description

Sets the option of EAN-13 Barcode.

Syntax

LRSCANNER_API BOOL LRSCAN_SetEAN13(PEAN13_PARAMS pEan13);

Parameters

pEan13

Pointer to an EAN13_PARAMS structure holding the EAN13 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetEAN13

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetEAN13(ref EAN13_PARAMS pEan13)

Example

```
PEAN13_PARAMS pEan13 = new EAN13_PARAMS();  
pEan13->bEnable = m_bEnable;  
pEan13->bISxN = m_bIsxn;  
pEan13->bXCD = m_bXCD;  
pEan13->bAddOn = m_bAddOn;  
if(LRSCAN_SetEAN13(pEan13) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_SetEAN13()", NULL, MB_TOPMOST);  
delete pEan13;
```

4.3.34 LRSCAN_SetEAN8

Description

Sets the option of EAN-8 Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetEAN8(PEAN8_PARAMS pEan8);
```

Parameters

pEan8

Pointer to a EAN8_PARAMS structure holding the EAN8 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetEAN8

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetEAN8(ref EAN8_PARAMS pEan8)

Example

```
PEAN8_PARAMS pEan8 = new EAN8_PARAMS();  
pEan8->bEnable = m_bEnable;  
pEan8->bXCD = m_bXCD;  
pEan8->bEAN8_AS_EAN13 = m_bEan8AsEan13;  
if(LRSCAN_SetEAN8(pEan8) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_SetEAN8()", NULL, MB_TOPMOST);
```

delete pEan8;

4.3.35 LRSCAN_SetGS1COMPOSITE

Description

Sets the option of GS1COMPOSITE Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetGS1COMPOSITE(PGS1COMPOSITE_PARAMS pGs1Composite);
```

Parameters

pGs1Composite

Pointer to a GS1COMPOSITE_PARAMS structure holding the GS1COMPOSITE common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetGS1COMPOSITE

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetGS1COMPOSITE(ref GS1COMPOSITE_PARAMS pGs1Composite)

Example

```
PGS1COMPOSITE_PARAMS pGs1Composite = new GS1COMPOSITE_PARAMS();
pGs1Composite->bEnable = m_bEnable;
pGs1Composite->bCC_C = m_bCC_C;
if(LRSCAN_SetGS1COMPOSITE(pGs1Composite) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_SetGS1COMPOSITE()", NULL, MB_TOPMOST);
delete pGs1Composite;
```

4.3.36 LRSCAN_SetGS1DATABAR

Description

Sets the option of GS1DATABAR Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetGS1DATABAR(PGS1DATABAR_PARAMS pGs1Databar);
```

Parameters

pGs1Databar

Pointer to a GS1DATABAR_PARAMS structure holding the GS1DATABAR common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetGS1DATABAR

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetGS1DATABAR(ref GS1DATABAR_PARAMS pGs1Databar)

Example

```
PGS1DATABAR_PARAMS pGs1Databar = new GS1DATABAR_PARAMS();
pGs1Databar->bEnable = m_bEnable;
pGs1Databar->bGS1LIM = m_bGs1Lim;
pGs1Databar->bGS1EXP = m_bGs1Exp;
if(LRSCAN_SetGS1DATABAR(pGs1Databar) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_SetGS1DATABAR()", NULL, MB_TOPMOST);
delete pGs1Databar;
```

4.3.37 LRSCAN_SetINT25

Description

Sets the option of INT25 Barcode.

Syntax

LRSCANNER_API BOOL LRSCAN_SetINT25(PINT25_PARAMS pInt25);

Parameters

pInt25

Pointer to a INT25_PARAMS structure holding the INT25 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetINT25

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetINT25(ref INT25_PARAMS pInt25)

Example

```
PINT25_PARAMS pInt25 = new INT25_PARAMS();  
pInt25->bEnable = m_bEnable;  
pInt25->bCDV = m_bCDV;  
pInt25->bXCD = m_bXCD;  
if(LRSCAN_SetINT25(pInt25) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_SetINT25()", NULL, MB_TOPMOST);  
delete pInt25;
```

4.3.38 LRSCAN_SetMSI

Description

Sets the option of MSI Barcode.

Syntax

LRSCANNER_API BOOL LRSCAN_SetMSI(PMSI_PARAMS pMsi);

Parameters

pMsi

Pointer to a MSI_PARAMS structure holding the MSI common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetMSI

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetMSI(ref MSI_PARAMS pMsi)

Example

```
PMSI_PARAMS pMsi = new MSI_PARAMS();  
pMsi->bEnable = m_bEnable;  
pMsi->bCDV = m_nCDV;  
pMsi->bXCD = m_bXCD;  
if(LRSCAN_SetMSI(pMsi) == FALSE)
```



```
        ::MessageBox(NULL, L"Error : LRSCAN_SetMSI()", NULL, MB_TOPMOST);  
delete pMsi;
```

4.3.39 LRSCAN_SetOption

Description

Sets the option of Scanner.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetOption(PDECODER_PARAMS pOption);
```

Parameters

pOption

Pointer to a DECODER_PARAMS structure holding the scanner parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetOption

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetOption(ref DECODER_PARAMS pOption)

Example

```
PDECODER_PARAMS pOption = new DECODER_PARAMS();  
pOption->nSound = m_nSound;  
pOption->bContinueMode = m_bContinueMode;  
pOption->bXmitAimID = m_bAimID;  
pOption->bVibrate = m_bVibrate;  
pOption->b1DDecodeMode = m_b1DDecode;  
pOption->bCenterDecode = m_bCenterDecode;  
pOption->nTimeOut = m_ctrlComboTimeOut.GetCurSel() + 1;  
pOption->nSecurityLevel = m_ctrlComboSecurityLevel.GetCurSel() + 1;  
if(LRSCAN_SetOption(pOption) == FALSE)  
{  
    ::MessageBox(NULL, L"ERROR : LRSCAN_SetOption()", NULL, MB_TOPMOST);  
    return FALSE;  
}
```

```
}
```

```
delete pOption;
```

4.3.40 LRSCAN_SetPLESSEY

Description

Sets the option of PLESSEY Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetPLESSEY(PPLESSEY_PARAMS pPlessey);
```

Parameters

pPlessey

Pointer to a PLESSEY_PARAMS structure holding the PPLESSEY common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetPLESSEY

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetPLESSEY(ref PLESSEY_PARAMS pPlessey)

Example

```
PPLESSEY_PARAMS pPlessey = new PLESSEY_PARAMS();  
pPlessey->bEnable = m_bEnable;  
pPlessey->bXCD = m_bXCD;  
if(LRSCAN_SetPLESSEY(pPlessey) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_SetPLESSEY()", NULL, MB_TOPMOST);  
delete pPlessey;
```

4.3.41 LRSCAN_SetPOSTNET

Description

Sets the option of POSTNET Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetPOSTNET(PPOSTNET_PARAMS pPostNet);
```

Parameters

pPostNet

Pointer to a POSTNET_PARAMS structure holding the POSTNET common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetPOSTNET

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetPOSTNET(ref POSTNET_PARAMS pPostNet)

Example

```
PPOSTNET_PARAMS pPostnet = new POSTNET_PARAMS();
pPostnet->bEnable = m_bEnable;
pPostnet->bXCD = m_bXCD;
if(LRSCAN_SetPOSTNET(pPostnet) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_SetPOSTNET()", NULL, MB_TOPMOST);
delete pPostnet;
```

4.3.42 LRSCAN_SetSTANDARD2OF5

Description

Sets the option of STANDARD2OF5 Barcode.

Syntax

LRSCANNER_API BOOL LRSCAN_SetSTANDARD2OF5(PSTANDARD2OF5_PARAMS pStandard2of5);

Parameters

pStandard2of5

Pointer to a STANDARD2OF5_PARAMS structure holding the STANDARD2OF5 common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetSTANDARD2OF5

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetSTANDARD2OF5(ref STANDARD2OF5_PARAMS pStandard2of5)

Example

```
PSTANDARD2OF5_PARAMS pStandard2of5 = new STANDARD2OF5_PARAMS();  
pStandard2of5->bEnable = m_bEnable;  
pStandard2of5->bCDV = m_bCDV;  
pStandard2of5->bXCD = m_bXCD;  
if(LRSCAN_SetSTANDARD2OF5(pStandard2of5) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_SetSTANDARD2OF5()", NULL, MB_TOPMOST);  
delete pStandard2of5;
```

4.3.43 LRSCAN_SetSymbology

Description

Sets Enable/Disable of Symbologies.

Syntax

LRSCANNER_API BOOL LRSCAN_SetSymbology(PDECODER pSym);

Parameters

pSymbology

Pointer to a DECODER structure holding the symbologies enable or disable.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetSymbology

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetSymbology(ref DECODER pSym)

Example

```
PDECODER pSym = new DECODER();  
BOOLbEnable[34] = {0, };  
int i = 0;  
for(i=0; i<34; i++)  
{
```

```
        bEnable[i] = m_ctrlListSymbology.GetCheck(i);
    }
    pSym->bAUSPOST = bEnable[0];
    pSym->bAZTEC = bEnable[1];
    pSym->bBPO = bEnable[2];
    pSym->bCANADAPOST = bEnable[3];
    pSym->bCODABAR = bEnable[4];
    pSym->bCODABLOCK = bEnable[5];
    pSym->bCODE11 = bEnable[6];
    pSym->bCODE39 = bEnable[7];
    pSym->bCODE93 = bEnable[8];
    pSym->bCODE128 = bEnable[9];
    pSym->bDATAMATRIX = bEnable[10];
    pSym->bDUTCHPOST = bEnable[11];
    pSym->bUPCA = bEnable[12];
    pSym->bUPCE = bEnable[13];
    pSym->bEAN8 = bEnable[14];
    pSym->bEAN13 = bEnable[15];
    pSym->bGS1_COMPOSITE = bEnable[16];
    pSym->bGS1_DATABAR = bEnable[17];
    pSym->bINFOMAIL = bEnable[18];
    pSym->bINT25 = bEnable[19];
    pSym->bJAPANPOST = bEnable[20];
    pSym->bMATRIX2OF5 = bEnable[21];
    pSym->bMAXICODE = bEnable[22];
    pSym->bMSI = bEnable[23];
    pSym->bPDF417 = bEnable[24];
    pSym->bMICRO_PDF417 = bEnable[25];
    pSym->bPLANET = bEnable[26];
    pSym->bPLESSEY = bEnable[27];
    pSym->bPOSTNET = bEnable[28];
    pSym->bQRCODE = bEnable[29];
    pSym->bSTANDARD2OF5 = bEnable[30];
    pSym->bSWEDENPOST = bEnable[31];
```

```
pSym->bTELEPEN = bEnable[32];  
pSym->bTLC39 = bEnable[33];  
if(LRSCAN_SetSymbology(pSym) == FALSE)  
    ::MessageBox(NULL, L"ERROR : LRSCAN_SetSymbology()", NULL, MB_TOPMOST);  
delete pSym;
```

4.3.44 LRSCAN_SetSymbologyAll

Description

Enables all of Symbologies.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetSymbologyAll();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetSymbologyDefault

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetSymbologyAll();

Example

None

4.3.45 LRSCAN_SetSymbologyDefault

Description

Initializes all option of scanner

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetSymbologyDefault();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_SetSymbologyAll

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetSymbologyDefault()

Example

None

4.3.46 LRSCAN_SetTELEPEN

Description

Sets the option of TELEPEN Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetTELEPEN(PTELEPEN_PARAMS pTelepen);
```

Parameters

pTelepen

Pointer to a TELEPEN_PARAMS structure holding the TELEPEN common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetTELEPEN

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetTELEPEN(ref TELEPEN_PARAMS pTelepen);

Example

```
PTELEPEN_PARAMS pTelepen = new TELEPEN_PARAMS();  
pTelepen->bEnable = m_bEnable;  
pTelepen->bNumeric = m_bNumeric;  
if(LRSCAN_SetTELEPEN(pTelepen) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_SetTELEPEN()", NULL, MB_TOPMOST);  
delete pTelepen;
```

4.3.47 LRSCAN_SetUPCA

Description

Sets the option of UPC-A Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetUPCA(PUPCA_PARAMS pUpca);
```

Parameters

pUpca

Pointer to a UPCA_PARAMS structure holding the UPC-A common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetUPCA

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetUPCA(ref UPCA_PARAMS pUpca);

Example

```
PUPCA_PARAMS pUpca = new UPCA_PARAMS();  
pUpca->bEnable = m_bEnable;  
pUpca->bXCD = m_bXCD;  
pUpca->bXNum = m_bXNum;  
pUpca->bUPCA_AS_EAN13 = m_bUpcaAsEan13;  
pUpca->bAddOn = m_bAddOn;  
if(LRSCAN_SetUPCA(pUpca) == FALSE)  
    ::MessageBox(NULL, L"Error : LRSCAN_SetUPCA()", NULL, MB_TOPMOST);  
delete pUpca;
```

4.3.48 LRSCAN_SetUPCE

Description

Sets the option of UPC-E Barcode.

Syntax

```
LRSCANNER_API BOOL LRSCAN_SetUPCE(PUPCE_PARAMS pUpce);
```


Parameters

pUpce

Pointer to a UPCE_PARAMS structure holding the UPC-E common parameters.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

LRSCAN_GetUPCE

For .Net

Namespace : LRScannerNet.LRScanner

Function : bool SetUPCE(ref UPCE_PARAMS pUpce)

Example

```
PUPCE_PARAMS pUpce = new UPCE_PARAMS();
pUpce->bEnable = m_bEnable;
pUpce->bXCD = m_bXCD;
pUpce->bXNum = m_bXNum;
pUpce->bUPCE_AS_UPCA = m_bUpceAsUpca;
if(LRSCAN_GetUPCE(pUpce) == FALSE)
    ::MessageBox(NULL, L"Error : LRSCAN_GetUPCE()", NULL, MB_TOPMOST);
delete pUpce;
```

4.4 CAMERA (CE)

Status Return value

Please refer to the below table for the status value definition.

Status Value Definition	Code	Meaning
TRUE	1	Success
FALSE	0	General Error

Define
<pre>//Image Effect #define OPTION_IMAGE_NOMAL_EFFECT 0 #define OPTION_IMAGE_SEPIA_EFFECT 1 #define OPTION_IMAGE_BLACKWHITE_EFFECT 2 #define OPTION_IMAGE_NEGATIVE_EFFECT 3 #define OPTION_IMAGE_UVRED_EFFECT 4 #define OPTION_IMAGE_SMART_AQUA_EFFECT 4 #define OPTION_IMAGE_UVBLUE_EFFECT 5 #define OPTION_IMAGE_UVGREEN_EFFECT 6 //Resolution #define OPTION_RESOLUTION_2048X1536 5 #define OPTION_RESOLUTION_1600X1200 4 #define OPTION_RESOLUTION_1280X1024 0 #define OPTION_RESOLUTION_640X480 1 #define OPTION_RESOLUTION_320X240 2 //Save Format #define OPTION_SAVE_FORMAT_BMP 0 #define OPTION_SAVE_FORMAT_JPG 1 //Image File Name Prefix #define OPTION_IMAGE_FILENAME_PREF_DATE 0 #define OPTION_IMAGE_FILENAME_PREF_SERIAL 1 // AF Status Massege #define WM_STATUS_AF (WM_USER + 0x1002) // GPS Status Messasge #define WM_GPS_ON (WM_USER + 0x1003) // Convert 2D Imager #define WM_CONVERT (WM_USER + 232) // Capture Status Message #define WM_SHOW_CAP_STATUS (WM_USER + 5)</pre>
Enum
<pre>enum AF_STATUS { AF_STATUS_IDLE = -1,</pre>

```

    AF_STATUS_START = 1,
    AF_STATUS_FINISH = 0
};

enum MODEL_TYPE
{
    MODEL_GREEN_13M,
    MODEL_GREEN,
    MODEL_T,
    MODEL_POS,
    MODEL_SMART,
    MODEL_ORANGE_CE,
    MODEL_UL10,
    MODEL_BLACK_CE,
    MODEL_UNKNOWN
};

enum CAP_STATUS
{
    CAP_STATUS_START = 0x1,
    CAP_STATUS_IMG_TRANSFORM = 0x2,
    CAP_STATUS_IMG_SCALE_UP = 0x4,
    CAP_STATUS_IMG_SAVE = 0x8,
    CAP_STATUS_END = 0x10,
    CAP_STATUS_ALL_PROCESS_END = 0x16,
    CAP_STATUS_COPY_DATA = 0x20
};

enum SCANNER_TYPE{
    SCANNER_OPTICON,
    SCANNER_SYMBOL,
    SCANNER_INTERMEC,
    SCANNER_HHP,
    SCANNER_ERR = -1
};

```

Structure

```

typedef struct
{
    INT    nResolution;    // Resolution
    INT    nSaveFormat;    // SaveFormat
    INT    nJpegQuality;   // Jpeg Quality
    INT    nNamePrefix;    // Image File Name Prefix
    INT    nImgEffect;     // Image Effect
    INT    nImgbalance;    // Image Balance
    INT    nImgRotatesave; //Image Rotate
    INT    nImgRotateview; //Image Rotate
} CAMERA_OPTION, *LPCAMERA_OPTION;

typedef struct _ExifInfo

```

```
{  
    TCHAR TitleName[21];  
    TCHAR Make[21];  
    TCHAR Model[21];  
}ExifInfo;
```

Functions for Camera (CE)

Name	Description
CAM_Open	Opens the Camera.
CAM_Close	Closes the Camera.
CAM_Capture	Captures the Still Picture.
CAM_PreviewStart	Starts viewing the Preview screen.
CAM_PreviewStop	Stops viewing the Preview screen.
CAM_GetLastSaveFilePath	Gets name of the latest file.
CAM_AutoFocus	Focuses automatically.
CAM_EnableAutoAF	Performs AF by automatically recognizing preview image changes.
CAM_Zoom	Zooms in and/or out.
CAM_SetCameraOption	Sets the Camera options.
CAM_GetCameraOption	Gets currently set Options of Camera.
CAM_Brightness	Changes the Brightness of Camera.
CAM_FlashOn	Turns on the Flash.
CAM_FlashOff	Turns off the Flash.
CAM_RawData	Gets the raw data of Preview screen.
CAM_InsertExifInformation	Insert EXIF information to Jpeg image.
CAM_UseGPSExifData	Insert GPS information in EXIF Information.
CAM_RegisterMsgWnd	Register the window handle getting Window Message from Camera.
CAM_GetScannerType	Gets Scanner Type of device.
CAM_GetVersion	Gets dll version.

4.4.1 CAM_Open

Description

Opens the Camera.

Syntax

```
MODEL_TYPE CAM_Open(HWND hMainWnd, HWND hPreviewWnd);
```

Parameters

hMainWnd

Windows Handle of Camera Application

hPreviewWnd

Windows handle to show image

Return Value

MODEL_TYPE is enum type value. Gets model name of device.

Remarks

None

See Also

CAM_Close

For .Net

Namespace : CamNet.Cam

Function : Cam.MODEL_TYPE Open(IntPtr hMainWnd, IntPtr hPreviewWnd)

Example

```
MODEL_TYPE m_Model;

m_Model = CAM_Open(m_hWnd, m_ctrPreview.m_hWnd) ;

if(m_Model == MODEL_UNKNOWN)
{
    ::MessageBox(NULL, L"Open error", L"Open", NULL);
    return;
}
```

4.4.2 CAM_Close

Description

Closes the Camera.

Syntax

```
CAM_EXPORTS BOOL CAM_Close()
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_Open

For .Net

Namespace : CamNet.Cam

Function : bool Close()

Example

```
CAM_Close();
```

4.4.3 CAM_Capture

Description

Captures the Still Picture.

Syntax

```
CAM_EXPORTS BOOL CAM_Capture(LPCTSTR strFilePath);
```

Parameters

strFilePath

Input the name and/or route of file to be stored.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : CamNet.Cam

Function : bool Capture(string strPath)

Example

```
TCHAR m_tzSavePath[256] = {0x00};
```

```
if(!m_bCapturing)
```

```
{
```

```
CAM_Capture(m_tzSavePath);  
}
```

4.4.4 CAM_PreviewStart

Description

Starts viewing the Preview screen.

Syntax

```
CAM_EXPORTS BOOL CAM_PreviewStart();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_PreviewStop

For .Net

Namespace : CamNet.Cam

Function : bool PreviewStart()

Example

```
CAM_PreviewStart();
```

4.4.5 CAM_PreviewStop

Description

Stops viewing the Preview screen.

Syntax

```
CAM_EXPORTS BOOL CAM_PreviewStop();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_PreviewStart

For .Net

Namespace : CamNet.Cam

Function : bool PreviewStop()

Example

```
CAM_PreviewStop();
```

4.4.6 CAM_GetLastSaveFilePath

Description

Gets name of the latest file.

Syntax

```
CAM_EXPORTS BOOL CAM_GetLastSaveFilePath(LPTSTR strOutFilePath);
```

Parameters

strOutFilePath

Obtains the last captured picture's name.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_Capture

For .Net

Namespace : CamNet.Cam

Function : bool GetLastSaveFilePath(out string strFilePath);

Example

```
TCHAR m_tzSavePath[256] = {0x00};
```

```
CAM_GetLastSaveFilePath(m_tzSavePath);
```

4.4.7 CAM_AutoFocus

Description

Focuses automatically.

Syntax

```
CAM_EXPORTS BOOL CAM_AutoFocus();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_EnableAutoAF

For .Net

Namespace : CamNet.Cam

Function : bool AutoFocus()

Example

```
CAM_AutoFocus();
```

4.4.8 CAM_EnableAutoAF

Description

Performs AF by automatically recognizing preview image changes.

Syntax

```
CAM_EXPORTS BOOL CAM_EnableAutoAF(BOOL bEnable);
```

Parameters

bEnable

Performs AF by automatically recognizing preview image changes. (supported in M3T)

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_AutoFocus

For .Net

Namespace : CamNet.Cam

Function : bool EnableAutoAF(bool bEnable)

Example

```
CAM_EnableAutoAF(TRUE);
```

4.4.9 CAM_Zoom

Description

Zooms in and/or out.

Syntax

```
CAM_EXPORTS BOOL CAM_Zoom(int nZoom);
```

Parameters

index

Sets zoom function according to Index.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : CamNet.Cam

Function : int Zoom(int index)

Example

```
CAM_Zoom(1);
```

4.4.10 CAM_SetCameraOption

Description

Sets the Camera options.

Syntax

```
CAM_EXPORTS BOOL CAM_SetCameraOption(LPCAMERA_OPTION option, LPTSTR strSaveFolder);
```

Parameters

option

LPCAMERA_OPTION Structure pointer inputs Option Value to set.

strSaveFolder

Inputs file name and/or route to be stored.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_GetCameraOption

For .Net

Namespace : CamNet.Cam

Function : bool SetCameraOption(ref Cam.CAMERA_OPTION option, string strSavePath)

Example

```
CAMERA_OPTION CamOption;  
CamOption.nImgEffect = OPTION_IMAGE_NOMAL_EFFECT;  
CamOption.nResolution = OPTION_RESOLUTION_1280X1024;  
CAM_SetCameraOption(&m_CamOption, L"Flash Disk\\Camera");
```

4.4.11 CAM_GetCameraOption

Description

Gets currently set Options of Camera.

Syntax

```
CAM_EXPORTS BOOL CAM_GetCameraOption(LPCAMERA_OPTION option, LPTSTR strSaveFolder);
```

Parameters

option

LPCAMERA_OPTION Structure pointer obtains the Option Value.

strSaveFolder

Obtains file name and/or route to be stored.

Return Value

Nonzero indicates success. Zero indicates failure

Remarks

None

See Also

CAM_SetCameraOption

For .Net

Namespace : CamNet.Cam

Function : bool SetCameraOption(ref Cam.CAMERA_OPTION option, string strSavePath)

Example

```
TCHAR tzSavePath[260] = {0x00};  
CAMERA_OPTION CamOption;  
CAM_GetCameraOption(&CamOption, m_tzSavePath);
```

4.4.12 CAM_Brightness

Description

Changes the Brightness of Camera.

Syntax

```
CAM_EXPORTS BOOL CAM_Brightness(int nBrightness);
```

Parameters

nBrightness

Sets the brightness.

Return Value

Nonzero indicates success. Zero indicates failure

Remarks

None

See Also

CAM_SetCameraOption, CAM_GetCameraOption

For .Net

Namespace : CamNet.Cam

Function : bool Brightness(int nBrightness);

Example

```
CAM_Brightness(2);
```

4.4.13 CAM_FlashOn

Description

Turns on the Flash.

Syntax

```
CAM_EXPORTS BOOL CAM_FlashOn();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_FlashOff

For .Net

Namespace : CamNet.Cam

Function : bool FlashOn()

Example

```
CAM_FlashOn();
```

4.4.14 CAM_FlashOff

Description

Turns off the Flash.

Syntax

```
CAM_EXPORTS BOOL CAM_FlashOff();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_FlashOn

For .Net

Namespace : CamNet.Cam

Function : bool FlashOff()

Example

```
CAM_FlashOff();
```

4.4.15 CAM_RawData

Description

Obtains data of Preview screen.

Syntax

```
CAM_EXPORTS void CAM_RawData(byte* data);
```

Parameters

data

Obtains data of current preview screen.

Return Value

void

Remarks

None

See Also

None

For .Net

Namespace : CamNet.Cam

Function : void RawData(byte[] data)

Example

None

4.4.16 CAM_InsertExifInformation

Description

Inserts EXIF information to Jpeg image.

Syntax

```
CAM_API BOOL CAM_InsertExifInformation(TCHAR* FileName, ExifInfo info);
```

Parameters

FileName

File name of JPEG which EXIF Information is to be stored.

Info

ExifInfo Structure.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_UseGPSExifData

For .Net

Namespace : CamNet.Cam

Function : bool InsertExifInformation(string FileName, Cam.ExifInfo e)

Example

```
ExifInfo info;
```

```
_tcscopy(info.TitleName, _T("Picture0.jpg"));
```

```
_tcscopy(info.Make, _T("M3 Mobile Co.Ltd"));
```

```
_tcscopy(info.Model, _T("M3 SMART"));
```

```
CAM_InsertExifInformation(L"Flash Disk\\Camera\\Photo\\Picture0.jpg", info);
```

4.4.17 CAM_UseGPSExifData

Description

Inserts GPS information to EXIF Information.

Syntax

```
CAM_API BOOL CAM_UseGPSExifData(BOOL bUse);
```

Parameters

bUse

Whether the GPS information is included in EXIF Information or not.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

Once GPS function is set to use, GPS satellite should be visualized. GPS information starts to be inserted in EXIF information if GPS signal can be received from satellite.

See Also

CAM_InsertExifInformation

For .Net

Namespace : CamNet.Cam

Function : bool UseGPSExifData(bool bUse)

Example

```
CAM_UseGPSExifData(TRUE);
```

4.4.18 CAM_RegisterMsgWnd

Description

Register the window handle getting Window Message from Camera.

Syntax

```
CAM_EXPORTS void CAM_RegisterMsgWnd(HWND hWnd);
```

Parameters

hWnd

Window Handle that gets messages from library

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : CamNet.Cam

Function : void RegisterMsgWnd(IntPtr hWnd);

Example

None

4.4.19 CAM_GetScannerType

Description

Gets Scanner Type of device.

Syntax

```
CAM_EXPORTS SCANNER_TYPE CAM_GetScannerType();
```

Parameters

None

Return Value

SCANNER_TYPE is Enum type value. Gets scanner type of device.

Remarks

None

See Also

None

For .Net

Namespace : CamNet.Cam

Function : bool GetScannerType()

Example

```
SCANNER_TYPE type = CAM_GetScannerType();
```

4.4.20 CAM_GetVersion

Description

Gets current version of Camera DLL.

Syntax

```
CAM_EXPORTS BOOL CAM_GetVersion(TCHAR* tzDllVersion, TCHAR* tzReleaseDate, TCHAR* tzPixels);
```

Parameters

tzDllVersion

Obtains DLL version.

tzReleaseDate

Obtains released date of DLL.

tzPixels

Obtains pixels of Camera.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : CamNet.Cam

Function : bool GetVersion()

Example

None

4.5 CAMERA (WM)

Status Return value

Please refer to the below table for the status value definition.

Status Value Definition	Code	Meaning
TRUE	1	Success
FALSE	0	General Error

Define
<pre>#define WM_STATUS_AF WM_USER + 916 #define WM_CONVERT WM_USER + 232 #define WM_GPS_ONWM_USER + 1020</pre>
Enum
<pre>typedef enum { MODEL_SKY, MODEL_MM3, MODEL_ORANGE, MODEL_SMART, MODEL_ORANGE_PLUS, MODEL_BLACK, MODEL_UNKONWN } MODEL_TYPE; typedef enum { SCANNER_OPTICON, SCANNER_SYMBOL, SCANNER_INTERMEC, SCANNER_HHP } SCANNER_TYPE; typedef enum { STILL_MODE, VIDEO_MODE, CLOSE_MODE } CAMERA_MODE; typedef enum { VIDEO_ASF, VIDEO_WMV } VIDEO_TYPE; typedef enum { AF_MANUAL, AF_ALWAYS, AF_AUTO } AF_MODE;</pre>

```
typedef enum{
    AF_STATUS_READY,
    AF_STATUS_START,
    AF_STATUS_FINISH
} AF_STATUS;
typedef enum{
    WB_Auto,
    WB_Sunny,
    WB_Cloudy,
    WB_Fluorescent,
    WB_Incandescent
}WHITE_BALANCE;
typedef enum{
    SAVE_DATE=0,
    SAVE_CUSTIM=1
}SAVE_TYPE;
```

Structure

```
typedef struct _ExifInfo
{
    TCHAR TitleName[21];
    TCHAR Make[21];
    TCHAR Model[21];
}ExifInfo;
```

Functions for Camera (WM)

Name	Description
CAM_Open	Opens the Camera.
CAM_Close	Closes the Camera.
CAM_SetPreviewWindow	Sets the size of Preview screen.
CAM_PreviewStart	Starts viewing the Preview screen.
CAM_PreviewStop	Stops viewing the Preview screen.
CAM_GetRegCapturePath	Get save filename. (old version)
CAM_GetRegCapturePathEx	Get save filename.
CAM_SetRegCapturePath	Set save filename.
CAM_Capture	Captures the Still Picture.
CAM_CaptureEx	Captures the Still Picture.
CAM_EnableShutterSound	Set shutter sound enable/disable.
CAM_VideoStart	Starts Videoing.
CAM_VideoStop	Stops Videoing.
CAM_VideoStopEx	Stops Videoing.
CAM_GetFlashState	Sets whether the flash is on or not when capturing picture.
CAM_AlwaysFlash	Turns on the Flash all the time.
CAM_CaptureFlash	Sets whether the flash is on or not when capturing picture.
CAM_AutoFocus	Focuses automatically.
CAM_SetAfMode	Sets AF Mode.
CAM_Zoom	Zooms in and/or out.
CAM_SetResolution	Sets Resolution of picture.
CAM_GetResolution	Gets setting value of resolution.
CAM_SetQuality	Sets quality of Jpeg Still Shot.
CAM_GetQuality	Gets quality value of Jpeg Still shot.
CAM_SetBrightness	Sets brightness of camera.
CAM_GetBrightness	Gets brightness of camera.
CAM_SetWhiteBalance	Sets white balance of camera.
CAM_GetWhiteBalance	Gets white balance of camera.
CAM_SetContrast	Sets contrast of camera.
CAM_GetContrast	Gets contrast of camera.

<u>CAM_SetSharpness</u>	Sets sharpness of camera.
<u>CAM_GetSharpness</u>	Gets sharpness of camera.
<u>CAM_SetHistoEqual</u>	Sets whether performing of Histogram Equalization when taking pictures.
<u>CAM_GetHistoEqual</u>	Gets whether performing of Histogram Equalization when taking pictures.
<u>CAM_RegisterPreview</u>	Registers Callback function to get the preview screen.
<u>CAM_RegisterMsgWnd</u>	Registers Window to recive the message from Camera DLL.
<u>CAM_InsertExifInformation</u>	Insert EXIF information to Jpeg image.
<u>CAM_GetRegExifEnable</u>	Get use EXIF function.
<u>CAM_UseGPSExifData</u>	Insert GPS information in EXIF Information.
<u>CAM_GetRegExifGpsEnable</u>	Get use GPS function.
<u>CAM_GetRegInsertDateTimeEnable</u>	Get use insert datetime function.
<u>CAM_SetInsertDateTimeEnable</u>	Set insert datetime function enable/disable.
<u>CAM_GetModelType</u>	Gets model type of device.
<u>CAM_GetScannerType</u>	Gets Scanner Type of device.
<u>CAM_GetVersion</u>	Gets dll version.

4.5.1 CAM_Open

Description

Opens the Camera.

Syntax

```
CAM_API BOOL CAM_Open(HWND hWnd, CAMERA_MODE cameramode, VIDEO_TYPE videotype);
```

Parameters

hWnd

Windows Handle of Camera Application

cameramode

Sets Camera mode. CAMERA_MODE is enum data. If this value is STILL_MODE, setting camera to make still image. If this value is VIDEO_MODE, setting camera to make video.

Videotype

Sets Video mode. VIDEO_TYPE is enum data. If this value is VIDEO_WMV, videos file is saved *.wmv. If this value is VIDEO_ASF, videos file is saved *.asf.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_Close

For .Net

Namespace : CamNet.Cam

Function : bool Open()

Example

```
MODEL_TYPE m_Model;  
m_Model = CAM_Open(m_hWnd, m_ctrPreview.m_hWnd) ;  
if(m_Model == MODEL_UNKNOWN)  
{  
    ::MessageBox(NULL, L"Open error", L"Open", NULL);  
    return;  
}
```

4.5.2 CAM_Close

Description

Closes the Camera.

Syntax

```
CAM_API BOOL CAM_Close();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_Open

For .Net

Namespace : CamNet.Cam

Function : bool Close()

Example

```
CAM_Close();
```

4.5.3 CAM_SetPreviewWindow

Description

Sets the size of Preview screen.

Syntax

```
CAM_API BOOL CAM_SetPreviewWindow(long left, long top, long width, long height);
```

Parameters

left

Assigns left location of preview screen.

Top

Assigns upper location of preview screen.

width

Assigns width of preview screen.

height

Assigns height of preview screen.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_PreviewStart, CAM_PreviewStop

For .Net

Namespace : CamNet.Cam

Function : bool SetPreviewWindow(long left, long top, long width, long height)

Example

```
CAM_SetPreviewWindow(80,90,320,240); // in MM3
```

4.5.4 CAM_PreviewStart

Description

Starts viewing the Preview screen.

Syntax

```
CAM_API BOOL CAM_PreviewStart();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_PreviewStop

For .Net

Namespace : CamNet.Cam

Function : bool PreviewStart()

Example

```
CAM_PreviewStart();
```

4.5.5 CAM_PreviewStop

Description

Stops viewing the Preview screen.

Syntax

```
CAM_API BOOL CAM_PreviewStop();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_PreviewStart

For .Net

Namespace : CamNet.Cam

Function : bool PreviewStop()

Example

```
CAM_PreviewStop();
```

4.5.6 CAM_GetRegCapturePath

Description

Get save filename..

Syntax

```
CAM_API BOOL CAM_GetRegCapturePath(TCHAR* tzFullName);
```

Parameters

tzFullName

Filename in capture.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_SetRegCapturePath

For .Net

Namespace : CamNet.Cam

Function : bool GetRegCapturePath(StringBuilder strPath)

Example

```
TCHAR m_filename[260]={0,};
```

```
CAM_GetRegCapturePath(m_filename);
```

4.5.7 CAM_GetRegCapturePathEx

Description

Get save filename.

Syntax

```
CAM_API BOOL CAM_GetRegCapturePathEx(TCHAR* tzPath, TCHAR* tzName);
```

Parameters

tzPath

Save path in capture.

tzName

Save name in capture.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_SetRegCapturePath

For .Net

Namespace : CamNet.Cam

Function : bool GetRegCapturePathEx(StringBuilder strPath, StringBuilder strName)

Example

```
TCHAR m_szPath[260]={0,};
```

```
TCHAR m_szName[260]={0,};
```

```
CAM_GetRegCapturePathEx(m_szPath, m_szName);
```

4.5.8 CAM_SetRegCapturePath

Description

Set save filename.

Syntax

```
CAM_API BOOL CAM_SetRegCapturePath(TCHAR* tzPath, TCHAR* tzName);
```

Parameters

strPath

Set save filepath.

strName

Set save filename.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_GetRegCapturePathEx

For .Net

Namespace : CamNet.Cam

Function : bool SetRegCapturePath(StringBuilder strPath, StringBuilder strName)

Example

```
TCHAR* strPath=_T("\\Flash Disk\\Camera\\");
```

```
TCHAR* strName=_T("123.jpg");
```

```
CAM_SetRegCapturePath(strPath, strName);
```

4.5.9 CAM_Capture

Description

Captures the Still Picture. This is an old version. New version uses CAM_CaptureEx.

Syntax

```
CAM_API BOOL CAM_Capture(const TCHAR* tzInFileName, TCHAR* tzOutFileName, BOOL  
bAutolnit);
```

Parameters

tzInFileName

Input file name.

btzOutFileName

Obtains file name.

bAutolnit

Auto preview enable/disable after capture.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : CamNet.Cam

Function : `bool Capture(string tzInFileName, char[] tzOutFileName, bool bAutoInit)`

Example

```
TCHAR tzFile[MAX_PATH] = {0x00};
TCHAR tzOutFile[MAX_PATH] = {0x00};
memset(tzFile, 0x00, sizeof(TCHAR) * MAX_PATH);
memset(tzOutFile, 0x00, sizeof(TCHAR) * MAX_PATH);
wsprintf(tzFile, L"\\Flash Disk\\Cam\\Picture1.jpg");
m_bAutoInit = TRUE;
CAM_Capture(tzFile, tzOutFile, m_bAutoInit);
```

4.5.10 CAM_CaptureEx

Description

Captures the Still Picture.

Syntax

`CAM_API BOOL CAM_CaptureEx(SAVE_TYPE nSaveType, BOOL bAutoInit, TCHAR* tzOutFileName);`

Parameters

nSaveType

SAVE_TYPE.

bAutoInit

Initiates camera automatically after capturing pictures.

tzOutFileName

Obtains file name.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : `CamNet.Cam`

Function : `bool CaptureEx(string tzInFileName, char[] tzOutFileName, bool bAutoInit)`

Example

```
TCHAR tzOutFile[MAX_PATH] = {0x00};
memset(tzOutFile, 0x00, sizeof(TCHAR) * MAX_PATH);
```

```
m_bAutoInit = TRUE;  
CAM_CaptureEx(SAVE_DATE, bAutoInit, m_tzOutFile);
```

4.5.11 CAM_EnableShutterSound

Description

Enable/disable shutter sound in capture.

Syntax

```
CAM_API BOOL CAM_EnableShutterSound(BOOL bEnable);
```

Parameters

bEnable.

TRUE is enable, False is disable.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None.

For .Net

Namespace : CamNet.Cam

Function : bool EnableShutterSound()

Example

```
CAM_EnableShutterSound(TRUE);
```

4.5.12 CAM_VideoStart

Description

Starts Video capturing.

Syntax

```
CAM_API BOOL CAM_VideoStart();
```

Parameters

None.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_VideoStop

For .Net

Namespace : CamNet.Cam

Function : bool VideoStart()

Example

```
CAM_VideoStart();
```

4.5.13 CAM_VideoStop

Description

Stops Video capturing

Syntax

```
CAM_API BOOL CAM_VideoStop(const TCHAR* tzInFileName, TCHAR* tzOutFileName);
```

Parameters

tzInFileName

Input filename.

tzOutFileName

Obtains file name.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_VideoStart

For .Net

Namespace : CamNet.Cam

Function : bool VideoStop(char[] tzInFileName, char[] tzOutFileName)

Example

```
TCHAR tzFile[MAX_PATH] = {0x00};  
TCHAR tzOutFile[MAX_PATH] = {0x00};  
memset(tzFile, 0x00, sizeof(TCHAR) * MAX_PATH);  
memset(tzOutFile, 0x00, sizeof(TCHAR) * MAX_PATH);  
wsprintf(tzFile, L"\\Flash Disk\\Camera\\ ");  
CAM_VideoStop(tzFile, tzOutFile);
```

4.5.14 CAM_VideoStopEx

Description

Stops Video capturing

Syntax

```
CAM_API BOOL CAM_VideoStopEx(SAVE_TYPE nSaveType, TCHAR* tzOutFileName);
```

Parameters

nSaveType

SAVE_TYPE.

tzOutFileName

Obtains file name.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_VideoStart

For .Net

Namespace : CamNet.Cam

Function : bool VideoStopEx(SAVE_TYPE nSaveType, char[] tzOutFileName)

Example

```
TCHAR tzFile[MAX_PATH] = {0x00};  
TCHAR tzOutFile[MAX_PATH] = {0x00};  
memset(tzFile, 0x00, sizeof(TCHAR) * MAX_PATH);  
memset(tzOutFile, 0x00, sizeof(TCHAR) * MAX_PATH);  
wsprintf(tzFile, L"\\Flash Disk\\Camera\\");  
CAM_VideoStopEx(tzFile, tzOutFile);
```

4.5.15 CAM_GetFlashState

Description

Get Flash Status.

Syntax

```
CAM_API BOOL CAM_GetFlashState();
```

Parameters

None.

Return Value

Nonzero indicates flash on. Zero indicates flash off.

Remarks

None.

See Also

None.

For .Net

Namespace : CamNet.Cam

Function : bool GetFlashState()

Example

```
BOOL bFlashOn = CAM_GetFlashState();
```

4.5.16 CAM_AlwaysFlash

Description

Turns on the Flash all the time.

Syntax

```
CAM_API BOOL CAM_AlwaysFlash(BOOL bOn);
```

Parameters

bOn

Sets the flash mode.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

Flash should be turned off before the Camera program is closed.

See Also

CAM_CaptureFlash

For .Net

Namespace : CamNet.Cam

Function : bool AlwaysFlash(bool bOn)

Example

```
CAM_AlwaysFlash(true);
```

4.5.17 CAM_CaptureFlash

Description

Sets whether the flash is on or not when capturing picture

Syntax

```
CAM_API BOOL CAM_CaptureFlash(BOOL bOn);
```

Parameters

bOn

Sets the flash mode.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_AlwaysFlash

For .Net

Namespace : CamNet.Cam

Function : bool CaptureFlash(int bOn)

Example

```
CAM_CaptureFlash(true);
```

4.5.18 CAM_AutoFocus

Description

Focuses automatically.

Syntax

```
CAM_API BOOL CAM_AutoFocus();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_SetAfMode

For .Net

Namespace : CamNet.Cam

Function : bool AutoFocus()

Example

```
CAM_AutoFocus();
```

4.5.19 CAM_SetAfMode

Description

Sets AF Mode.

Syntax

```
CAM_API AF_MODE CAM_SetAfMode(AF_MODE mode);
```

Parameters

mode

Sets camera mode to perform the AutoFocus. AF_MODE is Enum data.

0 : Manual AutoFocus – Performs AF function when user sets.

1 : Always AutoFocus – Performs AF function before capturing pictures.

3 : Auto AutoFocus – Performs AF function as recognizing the preview screen change.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_AutoFocus

For .Net

Namespace : CamNet.Cam

Function : Cam.AF_MODE SetAfMode(Cam.AF_MODE mode)

Example

```
CAM_SetAfMode(0);
```

4.5.20 CAM_Zoom

Description

Zooms in and/or out.

Syntax

```
CAM_API BOOL CAM_Zoom(int index);
```

Parameters

index

Sets zoom function according to Index.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : CamNet.Cam

Function : int Zoom(int index)

Example

```
CAM_Zoom(1);
```

4.5.21 CAM_SetResolution

Description

Sets Resolution of picture.

Syntax

```
CAM_API BOOL CAM_SetResolution(int nResolution);
```

Parameters

nResolution

The range of value is 0 to 6.

(0 : 176*144, 1 : 320*240, 2 : 352*288, 3 : 640*480, 4 : 800*600, 5 : 1280*960, 6 : 1600*1200)

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_GetResolution

For .Net

Namespace : CamNet.Cam

Function : int SetResolution(int nResolution)

Example

```
CAM_SetResolution(0);
```

4.5.22 CAM_GetResolution

Description

Gets setting value of resolution.

Syntax

```
CAM_API int CAM_GetResolution();
```

Parameters

None

Return Value

Gets setting value of resolution.

(0 : 176*144, 1 : 320*240, 2 : 352*288, 3 : 640*480, 4 : 800*600, 5 : 1280*960, 6 : 1600*1200)

Remarks

None

See Also

CAM_SetResolution

For .Net

Namespace : CamNet.Cam

Function : int CAM_GetResolution ()

Example

```
int nResolution = CAM_GetResolution();
```

4.5.23 CAM_SetQuality

Description

Sets quality of Jpeg Still Shot.

Syntax

```
CAM_API BOOL CAM_SetQuality(int nQuality);
```

Parameters

nQuality

The value specifies quality (0 : Low, 1 : Normal, 2 : High)

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_GetQuality

For .Net

Namespace : CamNet.Cam

Function : bool SetQuality(int nQuality)

Example

CAM_SetQuality(0);

4.5.24 CAM_GetQuality

Description

Gets quality value of Jpeg Still shot.

Syntax

CAM_API int CAM_GetQuality();

Parameters

None

Return Value

Gets quality value of Jpeg Still Shot.

Remarks

None

See Also

CAM_SetQuality

For .Net

Namespace : CamNet.Cam

Function : int GetQuality()

Example

int nQuality = CAM_GetQuality();

4.5.25 CAM_SetBrightness

Description

Sets brightness of camera.

Syntax

CAM_API BOOL CAM_SetBrightness(int nBrightness);

Parameters

nBrightness

Value's range is +4~-4.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_GetBrightness

For .Net

Namespace : CamNet.Cam

Function : bool SetBrightness(int nBrightness)

Example

```
CAM_SetBrightness(0);
```

4.5.26 CAM_GetBrightness

Description

Gets brightness of camera.

Syntax

```
CAM_API int CAM_GetBrightness();
```

Parameters

None

Return Value

Gets birhgtness of camera.

Remarks

None

See Also

CAM_SetBrightness

For .Net

Namespace : CamNet.Cam

Function : int GetBrightness()

Example

```
int nBrightness = CAM_GetBrightness();
```

4.5.27 CAM_SetWhiteBalance

Description

Sets white balance of camera.

Syntax

```
CAM_API BOOL CAM_SetWhiteBalance(WHITE_BALANCE nWhiteBalance);
```

Parameters

nWhiteBalance

white balance value (0 : Auto, 1 : Sunny, 2 : Cloudy, 3 : Fluorescent, 4 : Incandescent).

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_GetWhiteBalance

For .Net

Namespace : CamNet.Cam

Function : bool SetWhiteBalance(Cam.WHITE_BALANCE nWhiteBalance)

Example

```
CAM_SetWhiteBalance(WB_Auto);
```

4.5.28 CAM_GetWhiteBalance

Description

Gets white balance of camera.

Syntax

```
CAM_API WHITE_BALANCE CAM_GetWhiteBalance();
```

Parameters

None

Return Value

WHITE_BALANCE is enum. (0 : Auto, 1 : Sunny, 2 : Cloudy, 3 : Fluorescent, 4 : Incandescent).

Remarks

None

See Also

CAM_SetWhiteBalance

For .Net

Namespace : CamNet.Cam

Function : Cam.WHITE_BALANCE GetWhiteBalance()

Example


```
WHITE_BALANCE wbValue = CAM_GetWhiteBalance();
```

4.5.29 CAM_SetContrast

Description

Sets contrast of camera.

Syntax

```
CAM_API BOOL CAM_SetContrast(int nContrast);
```

Parameters

nContrast

Inputs contrast value of camera.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_GetContrast

For .Net

Namespace : CamNet.Cam

Function : bool SetContrast(int nContrast)

Example

```
CAM_SetContrast(1);
```

4.5.30 CAM_GetContrast

Description

Gets contrast of camera.

Syntax

```
CAM_API int CAM_GetContrast();
```

Parameters

none

Return Value

Gets contrast of camera.

Remarks

None

See Also

CAM_SetContrast

For .Net

Namespace : CamNet.Cam

Function : int GetContrast()

Example

```
int nContrast = CAM_GetContrast();
```

4.5.31 CAM_SetSharpness

Description

Sets sharpness of camera.

Syntax

```
CAM_API BOOL CAM_SetSharpness(int nSharpness);
```

Parameters

nSharpness

Inputs sharpness value of camera.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_GetSharpness

For .Net

Namespace : CamNet.Cam

Function : bool SetSharpness(int nSharpness)

Example

```
CAM_SetSharpness(0);
```

4.5.32 CAM_GetSharpness

Description

Gets sharpness of camera.

Syntax

```
CAM_API int CAM_GetSharpness();
```

Parameters

none

Return Value

Gets sharpness value of camera.

Remarks

None

See Also

CAM_SetSharpness

For .Net

Namespace : CamNet.Cam

Function : int GetSharpness()

Example

```
Int nSharpness = CAM_GetSharpness();
```

4.5.33 CAM_SetHistoEqual

Description

Sets whether performing of Histogram Equalization when taking pictures.

Syntax

```
CAM_API void CAM_SetHistoEqual(BOOL Enable);
```

Parameters

Enable

Sets whether performing of Histogram Equalization when taking pictures.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_GetHistoEqual

For .Net

Namespace : CamNet.Cam

Function : void SetHistoEqual(bool Enable)

Example

```
CAM_SetHistoEqual(TRUE);
```

4.5.34 CAM_GetHistoEqual

Description

Gets whether performing of Histogram Equalization when taking pictures.

Syntax

```
CAM_API BOOL CAM_GetHistoEqual();
```

Parameters

None

Return Value

Gets whether performing of Histogram Equalization when taking pictures.

Remarks

None

See Also

CAM_SetHistoEqual

For .Net

Namespace : CamNet.Cam

Function : int GetHistoEqual()

Example

```
BOOL bEnableHisto = CAM_GetHistEqual();
```

4.5.35 CAM_RegisterPreview

Description

Registers Callback function to get the preview screen.

Syntax

```
CAM_API BOOL CAM_RegisterPreview(MANAGED_SAMPLEPROCESSEDPROC fpPreview);
```

Parameters

fpPreview

CallBack Procedure formed a MANAGED_SAMPLEPROCESSEDPROC function.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

MANAGED_SAMPLEPROCESSEDPROC

Example

None

4.5.36 CAM_RegisterMsgWnd

Description

Registers Window to receive the message from Camera DLL.

Syntax

```
CAM_API void CAM_RegisterMsgWnd(HWND hWnd);
```

Parameters

hWnd

Windows Handle that receives message from the library.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : CamNet.Cam

Function : void RegisterMsgWnd(IntPtr hWnd);

Example

None

4.5.37 CAM_InsertExifInformation

Description

Insert EXIF information to Jpeg image.

Syntax

```
CAM_API BOOL CAM_InsertExifInformation(TCHAR* FileName, ExifInfo info);
```

Parameters

FileName

Jpeg file name which includes the EXIF Information.

Info

ExifInfo Structure.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

CAM_GetRegExifEnable

For .Net

Namespace : CamNet.Cam

Function : bool InsertExifInformation(string FileName, Cam.ExifInfo e)

Example

```
ExifInfo info;  
_tcsncpy(info.TitleName, _T("Picture0.jpg"));  
_tcsncpy(info.Make, _T("M3 Mobile Co.Ltd"));  
_tcsncpy(info.Model, _T("M3 SKY"));  
CAM_InsertExifInformation(L"Flash Disk\\Camera\\Photo\\Picture0.jpg", info);
```

4.5.38 CAM_GetRegExifEnable

Description

Get use EXIF function.

Syntax

```
CAM_API BOOL CAM_GetRegExifEnable();
```

Parameters

None.

Return Value

Nonzero indicates enable. Zero indicates disable.

Remarks

None

See Also

CAM_InsertExifInformation

For .Net

Namespace : CamNet.Cam

Function : bool GetRegExifEnable()

Example

```
BOOL bExifEnable = CAM_GetRegExifEnable();
```

4.5.39 CAM_UseGPSExifData

Description

Insert GPS information in EXIF Information.

Syntax

```
CAM_API BOOL CAM_UseGPSExifData(BOOL bUse);
```

Parameters

bUse

Whether the GPS information is included in EXIF Information or not.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

Once GPS function is set to use, GPS satellite should be visualized. GPS information starts to be inserted in EXIF information if GPS signal can be received from satellite.

See Also

CAM_GetRegExifGpsEnable

For .Net

Namespace : CamNet.Cam

Function : bool UseGPSExifData(bool bUse)

Example

```
CAM_UseGPSExifData(TRUE);
```

4.5.40 CAM_GetRegExifGpsEnable

Description

Get use GPS function.

Syntax

```
CAM_API BOOL CAM_GetRegExifGpsEnable();
```

Parameters

None.

Return Value

Nonzero indicates enable. Zero indicates disable.

Remarks

None.

See Also

CAM_UseGpsExifData

For .Net

Namespace : CamNet.Cam

Function : bool GetRegExifGpsEnable()

Example

```
BOOL bGpsEnable = CAM_GetRegExifGpsEnable();
```

4.5.41 CAM_GetRegInsertDateTimeEnable

Description

Get use GPS function.

Syntax

```
CAM_API BOOL CAM_GetRegInsertDateTimeEnable();
```

Parameters

None.

Return Value

Nonzero indicates enable. Zero indicates disable.

Remarks

None.

See Also

CAM_SetInsertDateTimeEnable

For .Net

Namespace : CamNet.Cam

Function : bool GetRegInsertDateTimeEnable()

Example

```
BOOL bDateTimeEnable = CAM_GetRegInsertDateTimeEnable();
```

4.5.42 CAM_SetInsertDateTimeEnable

Description

Get use GPS function.

Syntax

```
CAM_API BOOL CAM_SetInsertDateTimeEnable(int bEnable);
```

Parameters

bEnable

1 is enable, 0 is disable.

Return Value

Nonzero indicates enable. Zero indicates disable.

Remarks

None.

See Also

CAM_GetRegInsertDateTimeEnable

For .Net

Namespace : CamNet.Cam

Function : bool SetInsertDateTimeEnable

Example

CAM_SetInsertDateTimeEnable(TRUE);

4.5.43 CAM_GetModelType

Description

Gets model type of device.

Syntax

CAM_API MODEL_TYPE CAM_GetModelType();

Parameters

None

Return Value

MODEL_TYPE is Enum Type value. Obtains model type of device.

Remarks

None

See Also

CAM_GetScannerType

For .Net

Namespace : CamNet.Cam

Function : MODEL_TYPE GetModelType();

Example

MODEL_TYPE type = CAM_GetModelType();

4.5.44 CAM_GetScannerType

Description

Gets Scanner Type of device.

Syntax

CAM_API SCANNER_TYPE CAM_GetScannerType();

Parameters

None

Return Value

SCANNER_TYPE is Enum type value. Obtains scanner type of device.

Remarks

None

See Also

CAM_GetModelType

For .Net

Namespace : CamNet.Cam

Function : bool GetScannerType()

Example

```
SCANNER_TYPE type = CAM_GetScannerType();
```

4.5.45 CAM_GetVersion

Description

Gets dll version.

Syntax

```
CAM_API BOOL CAM_GetVersion(TCHAR* tzDllVersion, TCHAR* tzDllRelease);
```

Parameters

tzDllVersion

Obtains DLL version.

tzDllRelease

Obtains released date of DLL.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : CamNet.Cam

Function : bool GetVersion()

Example

None

4.6 RFID (LF/HF)

Status Return value

Please refer to the below table for the status value definition.

Status Value Definition	Code	Meaning
TRUE	1	Success
FALSE	0	General Error

Enum
<pre>enum TAG_TYPE_HF { ISO_14443_TYPE_A, ISO_14443_TYPE_B, ICODE_UID, ICODE_EPC, ICODE, SR176, ACTIVATE_ALL_TAG, ISO_15693 }; enum TAG_TYPE_LF { ACTIVATE_ALL_TAGS, EM4x02, EM4x05, EM4x50, HITAG1_S, HITAG2, TI_RFID_SYSTEMS }; enum RFID_TYPE { RFID_LF = 0, RFID_HF, RFID_NOTHING }; enum READ_MODE { READ_ASYNC, READ_SYNC, READ_CONTINUOUS }; enum STATE_SOUND</pre>

```
{  
    STATE_MSG,  
    STATE_READ,  
    STATE_WRITE  
};  
enum RESULT_MODE  
{  
    MODE_HEX,  
    MODE_DEC,  
    MODE_CHAR  
};  
enum SOUND_MODE  
{  
    SOUND_NO = 0,  
    SOUND_1,  
    SOUND_2,  
    SOUND_3,  
    VIB_1,  
    VIB_2  
};
```

Functions for RFID (LF/HF)

Name	Description
RFID_Open	Opens RFID.
RFID_Close	Closes RFID
RFID_PowerSupply	Supplies power to RFID module.
RFID_GetType	Gets RFID Radio Type of device.
RFID_SelectTag	Searches and Selects one tag within Antenna field.
RFID_HighSelectTag	Searches one tag within Antenna field and selects as High baudrate.
RFID_LoginTag	Login the tag if necessary.
RFID_ReadBlock	Reads memory block of the tag.
RFID_WriteBlock	Writes data in memory block of the tag.
RFID_ReadMultiBlock	Reads multiple data blocks on a card.
RFID_WriteMultiBlock	Writes multiple data blocks on a card.
RFID_SetAntenna	Controls antenna power of the RFID Module.
RFID_GetVersion	Gets DLL information and FW information of reader.
RFID_EnableMultiTag	Enables Anti-Collision function so that multi tags can be read.
RFID_SendReadMultiTag	Sends commander reading multi tags to reader.
RFID_SendTransferCommand	Allows card-specific communication.
RFID_SendContinuousRead	Reads and displays serial numbers continuously while one or more tags remain in the field.
RFID_SendCommand	Sends a command to a reader specified by its handle
RFID_SendCommandGetData	Sends a command to a reader specified by its handle and receives data..
RFID_GetData	Gets the result that performs commander stored in reader.
RFID_CheckResult	Checks obtained result from reader if it is valid.
RFID_SoundPlay	Plays sound and vibration.
RFID_SetTagType	Sets up the reader for a specific tag type.
RFID_GetTagType	Gets tag type.
RFID_GetTagTypeToString	Gets tag type as string.
RFID_TagItInventory	Performs Inventory commander of TagIt tag.
RFID_TagItReadBlock	Reads memory block of TagIt tag.
RFID_TagItWriteBlock	Writes memory block of TagIt tag.

4.6.1 RFID_Open

Description

Opens RFID.

Syntax

```
RFID_API RFID_TYPE RFID_Open();
```

Parameters

None

Return Value

RFID_TYPE is enum value. Can be identified whether it is LF reader or HF reader.

Remarks

Instead of not supporting the Close function, PowerSupply function cuts the power then close the application. (in CFReader.dll version 4.1.1)

See Also

RFID_PowerSupply

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool PowerSupply(bool bOn);

Example

```
RFID_TYPE m_RfType = RFID_Open();
TCHAR tzRadioType[256] = {0x00};
if(m_RfType == RFID_LF)
{
    wsprintf(tzRadioType, L"Low Frequency");
}
else if(m_RfType == RFID_HF)
{
    wsprintf(tzRadioType, L"High Frequency");
}
else
{
    return;
}
```

4.6.2 RFID_Close

Description

Close RFID.

Syntax

```
RFID_API BOOL RFID_Close();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

RFID_Close is supported in CFReader.dll version 4.1.5.7. If earlier version is used, power off via RFID_PowerSupply then close the program in M3 SKY. In M3 ORANGE, closing the program will close RFID.

See Also

RFID_PowerSupply

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool Close(bool bOn);

Example

```
RFID_Close();
```

4.6.3 RFID_PowerSupply

Description

Supplies power to RFID module.

Syntax

```
RFID_API BOOL RFID_PowerSupply(BOOL bOn);
```

Parameters

bOn

Supplies power to reader.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

This function is not necessary in M3 ORANGE.

See Also

None

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool PowerSupply(bool bOn)

Example

```
RFID_PowerSupply(FALSE);
```

4.6.4 RFID_GetType

Description

Gets RFID Radio Type of device.

Syntax

```
RFID_API RFID_TYPE RFID_GetType();
```

Parameters

None

Return Value

RFID_TYPE is enum value. Can be identified whether it is LF reader or HF reader.

Remarks

None

See Also

None

For .Net

Namespace : RFIDNet.RFIDCommon

Function : RFID_TYPE GetRadioType();

Example

```
RFID_TYPE m_RfType = RFID_GetType();
TCHAR tzRadioType[256] = {0x00};
if(m_RfType == RFID_LF)
{
    wsprintf(tzRadioType, L"Low Frequency");
}
else if(m_RfType == RFID_HF)
{
    wsprintf(tzRadioType, L"High Frequency");
}
else
{
    return;
```


}

4.6.5 RFID_SelectTag

Description

Searches and Selects one tag within Antenna field.

Syntax

```
RFID_API BOOL RFID_SelectTag(LPWSTR strSerial);
```

Parameters

strSerial

Obtains serial number of selected tag.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

RFID_HighSelectTag, RFID_ReadBlock, RFID_WriteBlock

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool SelectTag(StringBuilder strSerial)

Example

```
TCHAR tzSerial[100] = {0x00};
```

```
Rfid_SelectTag(tzSerial);
```

4.6.6 RFID_HighSelectTag

Description

Searches one tag within Antenna field and selects as High baudrate.

Syntax

```
RFID_API BOOL RFID_HighSelectTag(LPWSTR strSerial);
```

Parameters

strSerial

Obtains serial number of selected tag.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

This function can be used when the tag allows High Baudrate communication.

See Also

RFID_SelectTag, RFID_ReadBlock, RFID_WriteBlock

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool HighSelectTag(StringBuilder strSerial)

Example

```
TCHAR tzSerial[100] = {0x00};  
Rfid_HighSelectTag(tzSerial);
```

4.6.7 RFID_LoginTag

Description

Login the tag if necessary.

Syntax

```
RFID_API TCHAR RFID_LoginTag(LPCWSTR strLogin);
```

Parameters

strLogin

Inputs password to login.

Return Value

Obtains result of login as Wide Character. 'L' means success.

Remarks

None

See Also

RFID_ReadBlock, RFID_WriteBlock

For .Net

Namespace : RFIDNet.RFIDCommon

Function : char LoginTag(string strLogin)

Example

```
TCHAR tzSerial[32] = {0x00};  
TCHAR tzData[nMaxData] = {0x00};  
memset(tzSerial, 0x00, sizeof(TCHAR) * 32);  
memset(tzData, 0x00, sizeof(TCHAR) * nMaxData);  
if(Rfid_SelectTag(tzSerial))  
{
```

```

TCHAR tRet = RFID_LoginTag(L"01AFFFFFFFFFFFFF");
if(tRet == 'L' || tRet == 'I')
{
    Rfid_ReadBlock(L"01", tzData);
}
}

```

4.6.8 RFID_ReadBlock

Description

Reads memory block of the tag.

Syntax

RFID_API BOOL RFID_ReadBlock(LPCWSTR strBlock, LPWSTR strData)

Parameters

strBlock

Inputs Block Number to read.

strData

Obtains read Block Data.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

RFID_WriteBlock

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool ReadBlock(string strBlock, StringBuilder strData)

Example

```

TCHAR tzSerial[32] = {0x00};
TCHAR tzData[nMaxData] = {0x00};
memset(tzSerial, 0x00, sizeof(TCHAR) * 32);
memset(tzData, 0x00, sizeof(TCHAR) * nMaxData);
if(Rfid_SelectTag(tzSerial))
{
    Rfid_ReadBlock(L"01", tzData);
}

```

```
}
```

4.6.9 RFID_WriteBlock

Description

Writes data in memory block of the tag.

Syntax

```
RFID_API BOOL RFID_WriteBlock(LPCWSTR strBlock, LPCWSTR strInData, LPWSTR strOutData)
```

Parameters

strBlock

Inputs Block Number to write.

strInData

Inputs Block Data to write.

strOutData

Obtains written result.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

RFID_ReadBlock, RFID_WriteMultiBlock

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool WriteBlock(string strBlock, string strInData, StringBuilder strOutData)

Example

```
TCHAR tzSerial[32] = {0x00};
TCHAR tzOutData[nMaxData] = {0x00};
memset(tzSerial, 0x00, sizeof(TCHAR) * 32);
memset(tzOutData, 0x00, sizeof(TCHAR) * 1024);
if(Rfid_SelectTag(tzSerial))
{
    Rfid_WriteBlock(L"01", L"00112233", tzOutData);
}
```

4.6.10 RFID_ReadMultiBlock

Description

Reads multiple data blocks on a card.

Syntax

RFID_API void RFID_ReadMultiBlock(LPCWSTR strStartBlock, LPCWSTR strNumBlock, LPWSTR strData)

Parameters

strStartBlock

Inputs Start Block Number to read.

strNumBlock

Inputs number of Memory Block to read.

strData

Obtains Block Data to read.

Return Value

None

Remarks

This function can be used upper RFID version of 1.22.

See Also

RFID_WriteBlock

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool ReadBlock(string strBlock, StringBuilder strData)

Example

None

4.6.11 RFID_WriteMultiBlock

Description

Writes multiple data blocks on a card.

Syntax

RFID_API void RFID_WriteMultiBlock(LPCWSTR strStartBlock, LPCWSTR strNumBlock, LPCWSTR strWriteData, LPWSTR strOut);

Parameters

strStartBlock

Inputs Start Block Number to write.

strNumBlock

Inputs number of Memory Block to write.

strWriteData

Inputs Block Data to write.

strOut

Obtains written result.

Return Value

void

Remarks

This function can be used upper RFID version of 1.22.

See Also

RFID_ReadBlock, RFID_ReadMultiBlock

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool WriteBlock(string strBlock, string strInData, StringBuilder strOutData)

Example

```
TCHAR tzSerial[32] ={0x00};
TCHAR tzOutData[nMaxData] = {0x00};
memset(tzSerial, 0x00, sizeof(TCHAR) * 32);
memset(tzOutData, 0x00, sizeof(TCHAR) * 1024);
if(Rfid_SelectTag(tzSerial))
{
    RFID_WriteMultiBlock(L"01", L"02", L"0011223344556677", tzOutData);
}
```

4.6.12 RFID_SetAntenna

Description

Controls antenna power of the RFID Module.

Syntax

RFID_API void RFID_SetAntenna(BOOL bSet);

Parameters

bSet

Sets the condition of Antenna.

Return Value

None

Remarks

Main battery can be saved by turning off the antenna.

See Also

None

For .Net

Namespace : RFIDNet.RFIDCommon

Function : void SetAntenna(bool bOn)

Example

```
if(bAntennaOn == TRUE)
{
    Rfid_SetAntenna(TRUE);
}
else
{
    Rfid_SetAntenna(FALSE);
}
```

4.6.13 RFID_GetVersion

Description

Gets DLL information and FW information of reader.

Syntax

```
RFID_API BOOL RFID_GetVersion(LPWSTR szFwVersion, LPWSTR szReaderDllVersion, LPWSTR szRfidDllVersion);
```

Parameters

szFwVersion

Obtains FW version of reader.

szReaderDllVersion

Obtains version of CFReader.dll.

szRfidDllVersion

Obtains version of RFID.dll.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool GetVersion(StringBuilder strFwVersion, StringBuilder strReaderDllVersion, StringBuilder strRfidDllVersion)

Example

```
TCHAR tzVersion[32] = {0x00};  
TCHAR tzVersion2[32] = {0x00};  
TCHAR tzVersion3[32] = {0x00};  
RFID_GetVersion(tzVersion, tzVersion2, tzVersion3);
```

4.6.14 RFID_EnableMultiTag

Description

Enables Anti-Collision function so that multi tags can be read.

Syntax

RFID_API BOOL RFID_EnableMultiTag(BOOL bEnable);

Parameters

bEnable

Enables Anti-Collision function of reader.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

RFID_SendReadMultiTag , RFID_GetData

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool EnableMultiTag(bool bEnable)

Example

```
RFID_EnableMultiTag(TRUE);
```

4.6.15 RFID_SendReadMultiTag

Description

Sends commander reading multi tags to reader.

Syntax

RFID_API void RFID_SendReadMultiTag();

Parameters

None

Return Value

None

Remarks

MultiTag can be read with Rfid_SendContinuousRead function after RFID_EnableMultiTag.

See Also

RFID_EnableMultiTag, RFID_GetData, Rfid_SendContinuousRead

For .Net

Namespace : RFIDNet.RFIDCommon

Function : void SendReadMultiTag()

Example

None

4.6.16 RFID_SendTransferCommand

Description

This command allows card-specific communication.

Syntax

RFID_API void RFID_SendTransferCommand(LPWSTR strData)

Parameters

strData

Normal mode

Downlink length (1 byte) <> 0 Option byte (1 byte) Data (n bytes)

Transmit/Receive mode

Downlink length (1 byte) = 0 Downlink length new (1 byte) Option byte (1 byte) Transmit byte (1 byte) Receive byte (1 byte) CRC Preset LSB (1 byte) CRC Preset MSB (1 byte) Data (n bytes).

Return Value

None

Remarks

None

See Also

RFID_GetData

For .Net

Namespace : RFIDNet.RFIDCommon

Function : void SendTransferCommand(string strData);

Example

None

4.6.17 RFID_SendContinuousRead

Description

Reads and displays serial numbers continuously while one or more tags remain in the field.

Syntax

```
RFID_API void RFID_SendContinuousRead(BOOL bStart);
```

Parameters

bStart

Sets whether starting Continuous read or not.

Return Value

None

Remarks

MultiTag can be read when EnableMultiTag is enabled.

See Also

RFID_GetData , RFID_EnableMultiTag

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool SendContinuousRead(bool bStart)

Example

```
RFID_SendContinuousRead(TRUE);  
BOOL bRead = TRUE;  
while(bRead)  
{  
    TCHAR szSerial[512] = {0x00,};  
    memset(szSerial, 0x00, sizeof(TCHAR) * 512);  
    RFID_GetData(szSerial);  
    if(wcslen(szSerial) > 2 && bRead)  
    {  
        // Print szSerial  
    }  
}  
RFID_SendContinuousRead(FALSE);
```

4.6.18 RFID_SendCommand

Description

The RFID_SendCommand function sends a command to a reader specified by its handle.

Syntax

```
RFID_API void RFID_SendCommand(LPCWSTR strCommand, LPCWSTR strData);
```

Parameters

strCommand

Zero terminated string with the command

strData

Zero terminated string containing the data

Return Value

None

Remarks

None

See Also

RFID_GetData

For .Net

Namespace : RFIDNet.RFIDCommon

Function : void SendCommand(string strCommand, string strData)

Example

```
RFID_SendCommand(L"s", L"");  
BOOL bRead = TRUE;  
while(bRead)  
{  
    TCHAR szSerial[512] = {0x00,};  
    memset(szSerial, 0x00, sizeof(TCHAR) * 512);  
    RFID_GetData(szSerial);  
    if(wcslen(szSerial) > 2 && bRead)  
    {  
        // Print szSerial  
    }  
}
```

4.6.19 RFID_SendCommandGetData

Description

The RFID_SendCommandGetData function sends a command to a reader specified by its handle and receives data..

Syntax

RFID_API void RFID_SendCommandGetData(LPCWSTR strCommand, LPCWSTR strInputData, LPWSTR strOutputData);

Parameters

strCommand

Zero terminated string with the command

strInputData

Zero terminated string containing the data

strOutputData

Gets the result that performs commander stored in reader

Return Value

None

Remarks

None

See Also

RFID_SendCommand , RFID_GetData

For .Net

Namespace : RFIDNet.RFIDCommon

Function : void SendCommandGetData(string strCommand, string strInputData, StringBuilder strOutputData)

Example

```
TCHAR szSerial[512] = {0x00,};  
memset(szSerial, 0x00, sizeof(TCHAR) * 512);  
RFID_SendCommandGetData(L"s", L"", szSerial);
```

4.6.20 RFID_GetData

Description

Gets the result that performs commander stored in reader.

Syntax

RFID_API BOOL RFID_GetData(LPWSTR strData)

Parameters

strData

Result stored in reader can be obtained.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

This function can obtain the result from commanders from Send functions.

See Also

RFID_SendReadMultiTag, RFID_SendTransferCommand, RFID_SendContinuousRead

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool GetData(StringBuilder strData)

Example

```
RFID_SendContinuousRead(TRUE);  
BOOL bRead = TRUE;  
while(bRead)  
{  
    TCHAR szSerial[512] = {0x00,};  
    memset(szSerial, 0x00, sizeof(TCHAR) * 512);  
    RFID_GetData(szSerial);  
    if(wcslen(szSerial) > 2 && bRead)  
    {  
        // Print szSerial  
    }  
}  
RFID_SendContinuousRead(FALSE);
```

4.6.21 RFID_CheckResult

Description

Checks obtained result from reader if it is valid.

Syntax

RFID_API BOOL RFID_CheckResult(TCHAR* tzInputResult, TCHAR* tzOutputResult)

Parameters

tzInputResult

Inputs result obtained from reader.

tzOutputResult

Obtains message if it is valid result from reader.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

RFID_GetData

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool CheckResult(string strInputResult, StringBuilder strOutputResult);

Example

```
TCHAR tzSerial[32] = {0x00};
TCHAR tzData[nMaxData] = {0x00};
memset(tzSerial, 0x00, sizeof(TCHAR) * 32);
memset(tzData, 0x00, sizeof(TCHAR) * nMaxData);
if(Rfid_SelectTag(tzSerial))
{
    Rfid_ReadBlock(L"01", tzData);
}
TCHAR tzMsg[1024] = {0x00};
memset(tzMsg, 0x00, sizeof(TCHAR) * 1024);
if(RFID_CheckResult(tzOutData, tzMsg))
{
    // "Result: Write Success"
    // print tzMsg
}
```

4.6.22 RFID_SoundPlay

Description

Plays sound and vibration.

Syntax

RFID_API BOOL RFID_SoundPlay(SOUND_MODE Sound)

Parameters

Sound

SOUND_MODE is enum value type. Plays sound and vibration.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : RFIDNet.RFIDCommon

Function : bool SoundPlay(SOUND_MODE sound)

Example

```
RFID_SoundPlay(SOUND_1);
```

4.6.23 RFID_SetTagType

Description

This Function sets up the reader for a specific tag type.

Syntax

```
RFID_API int RFID_SetTagType(int nType)
```

Parameters

nType

TAG_TYPE_HF is applied to HF RFID and TAG_TYPE_LF is applied to LF RFID.

Return Value

Current Tag Type is returned.

Remarks

None

See Also

RFID_GetTagType, RFID_GetTagTypeToString

For .Net

Namespace : RFIDNet.RFIDCommon

Function : int SetTagType(int Type)

Example

```
RFID_SetTagType(ISO_14443_TYPE_B);
```

4.6.24 RFID_GetTagType

Description

Gets tag type.

Syntax

```
RFID_API int RFID_GetTagType();
```

Parameters

None

Return Value

TAG_TYPE_HF is applied to HF RFID and TAG_TYPE_LF is applied to LF RFID.

Remarks

None

See Also

RFID_GetTagTypeToString, RFID_SetTagType

For .Net

Namespace : RFIDNet.RFIDCommon

Function : int GetTagType();

Example

```
int nType = GetTagType();
```

4.6.25 RFID_GetTagTypeToString

Description

Gets tag type as string.

Syntax

```
RFID_API void RFID_GetTagTypeToString(TCHAR* tzTagType);
```

Parameters

tzTagType

Obtains current TagType as string.

Return Value

None

Remarks

None

See Also

RFID_SetTagType, RFID_GetTagType

For .Net

Namespace : RFIDNet.RFIDCommon

Function : void GetTagTypeToString(StringBuilder strTagType)

Example

```
TCHAR tzType[260] = {0x00};  
RFID_GetTagTypeToString(tzType);.
```

4.6.26 RFID_TagItInventory

Description

Performs Inventory commander of TagIt tag.

Syntax

RFID_API BOOL RFID_TagItInventory(TCHAR* tzUID)

Parameters

tzUID

Obtains UID of tag.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

RFID_TagItReadBlock, RFID_TagItWriteBlock

For .Net

Namespace : RFIDNet. RFIDTagIt

Function : bool Inventory(StringBuilder strUID);

Example

```
TCHAR tzInventory[256] = {0x00};  
memset(tzInventory, 0x00, sizeof(TCHAR) * 256);  
if(!RFID_TagItInventory(tzInventory))  
    return;  
  
TCHAR tzResult[256] = {0x00};  
memset(tzResult, 0x00, sizeof(TCHAR) * 256);  
RFID_TagItReadBlock(1, tzResult);.
```

4.6.27 RFID_TagItReadBlock

Description

Reads memory block of TagIt tag.

Syntax

```
RFID_API BOOL RFID_TagItReadBlock(int nBlockNo, TCHAR* tzBlockData);
```

Parameters

nBlockNo

Inputs Block Number to read.

tzBlockData

Obtains Block Data to read.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

RFID_TagItInventory, RFID_TagItWriteBlock

For .Net

Namespace : RFIDNet. RFIDTagIt

Function : bool ReadBlock(int nBlockNo, StringBuilder strBlockData)

Example

```
TCHAR tzInventory[256] = {0x00};
memset(tzInventory, 0x00, sizeof(TCHAR) * 256);
if(!RFID_TagItInventory(tzInventory))
    return;

TCHAR tzResult[256] = {0x00};
memset(tzResult, 0x00, sizeof(TCHAR) * 256);
RFID_TagItReadBlock(1, tzResult);.
```

4.6.28 RFID_TagItWriteBlock

Description

Writes memory block of TagIt tag.

Syntax

```
RFID_API BOOL RFID_TagItWriteBlock(int nBlockNo, CONST TCHAR* tzWriteBlockData, TCHAR*
tzWriteResult)
```

Parameters

nBlockNo

Inputs Block Number to write.

tzWriteBlockData

Inputs Block Data to write.

tzWriteResult

Obtains written result.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

RFID_TagItInventory, RFID_TagItReadBlock

For .Net

Namespace : RFIDNet. RFIDTagIt

Function : bool WriteBlock(int nBlockNo, string strWriteBlockData, StringBuilder strWriteResult)

Example

```
TCHAR tzInventory[256] = {0x00};
memset(tzInventory, 0x00, sizeof(TCHAR) * 256);
if(!RFID_TagItInventory(tzInventory))
    return;

TCHAR tzResult[256] = {0x00};
memset(tzResult, 0x00, sizeof(TCHAR) * 256);
RFID_TagItWriteBlock(1, L"00112233",tzResult);.
```

4.7 UHF GUN READER

Status Return value

Please refer to the below table for the status value definition.

Status Value Definition	Code	Meaning
TRUE	1	Success
FALSE	0	General Error

Define
<pre>typedef unsigned __int8 INT8U; typedef unsigned __int16 INT16U; typedef unsigned __int32 INT32U; typedef signed __int32 BOOL32; typedef unsigned __int32 HANDLE32; typedef HANDLE32 RFID_RADIO_HANDLE; #define WM_MSG_INVENTORY 0x5001 #define WM_MSG_ACCESS 0x5002 #define WM_MSG_POWER 0x5010 #define UHF_OPEN_EVENT L"UHF_OPEN_EVENT"</pre>
Enum
<pre>typedef enum { BATTERY_ERROR, BATTERY_0, BATTERY_1, BATTERY_2, BATTERY_3, BATTERY_FULL }BATTERY_STATUS; typedef enum { TAG_OTHERERROR = 0x0, TAG_SUCCESS = 0x1, TAG_MEMORYOVERRUN = 0x3, TAG_MEMORYLOCKED = 0x4, TAG_INSUFFICIENTPOWER = 0xB, TAG_NONSPECIFICERROR = 0xF, MAC_NOERROR = 0x10, MAC_HANDLEMISSMATCH = 0x11, MAC_CRCERROR = 0x12,</pre>

```
MAC_NOTAGREPLY          = 0x13,
MAC_INVALIDPASSWD       = 0x14,
MAC_ZEROKILLPASSWD      = 0x15,
MAC_TAGLOST             = 0x16,
MAC_COMMANDFORMATERROR  = 0x17,
MAC_READCOUNTINVALID  = 0x18,
MAC_OUTOFRETRIES        = 0x19,
```

```
}RFIDErrorCode;
```

```
typedef enum
```

```
{
    TAG_RESERVED,
    TAG_EPC,
    TAG_TID,
    TAG_USER
```

```
}RFIDTagBank;
```

```
typedef enum
```

```
{
    RFID_REGION_KOREA_NEW,
    RFID_REGION_KOREA_WEAK,
    RFID_REGION_KOREA_OLD,
    RFID_REGION_USA,
    RFID_REGION_EURO,
    RFID_REGION_EURO_NEW,
    RFID_REGION_JAPAN,
    RFID_REGION_CHINA,
    RFID_REGION_AUSTRALIA,
    RFID_REGION_BRAZIL,
    RFID_REGION_MALAYSIA,
    RFID_REGION_TAIWAN
```

```
}RFIDRegion;
```

```
typedef enum
```

```
{
    SELECT_EPC = 1,
    SELECT_TID = 2,
    SELECT_USER = 3
```

```
}RFIDSelectBank;
```

```
typedef enum
```

```
{
    PERMISSION_ACCESSIBLE,
    PERMISSION_ALWAYS_ACCESSIBLE,
    PERMISSION_SECURED_ACCESSIBLE,
    PERMISSION_ALWAYS_NOT_ACCESSIBLE,
```

```

    PERMISSION_NO_CHANGE
}RFIDLockPermission;

enum
{
    /* Success */
    RFID_STATUS_OK,
    /* Attempted to open a radio that is already open */
    RFID_ERROR_ALREADY_OPEN = -9999, /* -9999 */
    /* Buffer supplied is too small */
    RFID_ERROR_BUFFER_TOO_SMALL, /* -9998 */
    /* General failure */
    RFID_ERROR_FAILURE, /* -9997 */
    /* Failed to load radio bus driver */
    RFID_ERROR_DRIVER_LOAD, /* -9996 */
    /* Library cannot use version of radio bus driver present on system */
    RFID_ERROR_DRIVER_MISMATCH, /* -9995 */
    /* This error code is no longer used, maintain slot in enum in case
    * anyone is using hard-coded error codes for some reason */
    RFID_ERROR_RESERVED_01, /* -9994 */
    /* Antenna number is invalid */
    RFID_ERROR_INVALID_ANTENNA, /* -9993 */
    /* Radio handle provided is invalid */
    RFID_ERROR_INVALID_HANDLE, /* -9992 */
    /* One of the parameters to the function is invalid */
    RFID_ERROR_INVALID_PARAMETER, /* -9991 */
    /* Attempted to open a non-existent radio */
    RFID_ERROR_NO_SUCH_RADIO, /* -9990 */
    /* Library has not been successfully initialized */
    RFID_ERROR_NOT_INITIALIZED, /* -9989 */
    /* Function not supported */
    RFID_ERROR_NOT_SUPPORTED, /* -9988 */
    /* Operation was cancelled by call to cancel operation, close radio, or
    * shut down the library */
    RFID_ERROR_OPERATION_CANCELLED, /* -9987 */
    /* Library encountered an error allocating memory */
    RFID_ERROR_OUT_OF_MEMORY, /* -9986 */
    /* The operation cannot be performed because the radio is currently busy */
    RFID_ERROR_RADIO_BUSY, /* -9985 */
    /* The underlying radio module encountered an error */
    RFID_ERROR_RADIO_FAILURE, /* -9984 */
    /* The radio has been detached from the system */
    RFID_ERROR_RADIO_NOT_PRESENT, /* -9983 */
    /* The RFID library function is not allowed at this time. */
    RFID_ERROR_CURRENTLY_NOT_ALLOWED, /* -9982 */
    /* The radio module's MAC firmware is not responding to requests. */

```

```

RFID_ERROR_RADIO_NOT_RESPONDING,          /* -9981 */
/* The MAC firmware encountered an error while initiating the nonvolatile */
/* memory update. The MAC firmware will return to its normal idle state */
/* without resetting the radio module.                                     */
RFID_ERROR_NONVOLATILE_INIT_FAILED,        /* -9980 */
/* An attempt was made to write data to an address that is not in the */
/* valid range of radio module nonvolatile memory addresses.          */
RFID_ERROR_NONVOLATILE_OUT_OF_BOUNDS,      /* -9979 */
/* The MAC firmware encountered an error while trying to write to the */
/* radio module's nonvolatile memory region.                           */
RFID_ERROR_NONVOLATILE_WRITE_FAILED,       /* -9978 */
/* The underlying transport layer detected that there was an overflow */
/* error resulting in one or more bytes of the incoming data being */
/* dropped. The operation was aborted and all data in the pipeline was */
/* flushed.                                                              */
RFID_ERROR_RECEIVE_OVERFLOW,               /* -9977 */
/* An unexpected value was returned to this function by the MAC firmware */
RFID_ERROR_UNEXPECTED_VALUE,              /* -9976 */
/* The MAC firmware encountered CRC errors while trying to */
/* write to the radio module's nonvolatile memory region.             */
RFID_ERROR_NONVOLATILE_CRC_FAILED,         /* -9975 */
/* The MAC firmware encountered unexpected values in the packet header */
RFID_ERROR_NONVOLATILE_PACKET_HEADER,     /* -9974 */
/* The MAC firmware received more than the specified maximum packet size */
RFID_ERROR_NONVOLATILE_MAX_PACKET_LENGTH  /* -9973 */
}typedef RFID_STATUS;

```

Structure

```

typedef struct {
    /* The major version (i.e, in 1.x.x.x, the 1) */
    INT32U major;
    /* The minor version (i.e., in x.1.x.x, the 1) */
    INT32U minor;
    /* The maintenance number (i.e., in x.x.1.x, the 1) */
    INT32U maintenance;
    /* The release number (i.e., in x.x.x.1, the 1) */
    INT32U release;
} RFID_VERSION;

typedef struct
{
    HWND hWnd; //Diag handle Receive for Message
    RFIDTagBank bank;
    INT8U wlength;
    INT8U offset;
    INT32U accpwd;
}

```

```
}RFIDReadCmd;
```

```
typedef struct
```

```
{  
    HWND hWnd; //Diag handle Receive for Message  
    RFIDTagBank bank;  
    INT8U wlength;  
    INT8U offset;  
    INT16U wdata[32];  
    INT32U accpwd;
```

```
}RFIDWriteCmd;
```

```
typedef struct
```

```
{  
    HWND hWnd; //Diag handle Receive for Message  
    RFIDLockPermission lockkillpwd;  
    RFIDLockPermission lockaccesspwd;  
    RFIDLockPermission lockepc;  
    RFIDLockPermission locktid;  
    RFIDLockPermission lockuser;  
    INT32U accpwd;
```

```
}RFIDLockCmd;
```

```
typedef struct
```

```
{  
    HWND hWnd; //Diag handle Receive for Message  
    INT32U killpwd;
```

```
}RFIDKillCmd;
```


Functions for UHF GUN READER

Name	Description
UHF_GetError	Check the last error
UHF_IsReady	Check UHF GUN READER's availability by checking the power
UHF_Init	Initialize RFID
UHF_Close	Close RFID
UHF_Inventory	Start Inventory by reading EPC data
UHF_InventoryStop	Stop Inventory
UHF_Read	Read memory from tag
UHF_Write	Write data to tag
UHF_Lock	Limit access to tag
UHF_Kill	Kill the tag (disabling the tag)
UHF_GetData	Get tag data from Inventory and Read
UHF_SetRegionFrequency	Set RFID frequency to suit regional regulation
UHF_GetRegionFrequency	Check current regional frequency setting
UHF_ReadBattery	Check battery voltage and ADC value
UHF_ReadBatteryStatus	Check battery level (0~4 level)
UHF_Version	Check DLL and F/W version

4.7.1 UHF_GetError

Description

Check the last error

Syntax

```
RFIDErrorCode UHF_GetError();
```

Parameters

None

Return Value

Retuens RFIDErrorCode of enum type

Remarks

None

See Also

None

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : RFIDUHF.RFIDErrorCode GetError();

Example

```
length = UHF_GetData(data);
if(length == 0)
{
    err = UHF_GetError();
    switch(err)
    {
        case TAG_OTHERERROR:
            MessageBox(_T("Tag Other Error!"));
            break;
        case TAG_SUCCESS:
            MessageBox(_T("SUCCESS!"));
            break;
        case TAG_MEMORYOVERRUN:
            MessageBox(_T("Tag Memory Overrun!"));
            break;
        case MAC_OUTOFRETRIES:
            MessageBox(_T("Out of Retries Error!"));
```

```

        break;
    default:
        MessageBox(_T("Unknown Error!"));
        break;
    }
}

```

4.7.2 UHF_IsReady

Description

Check UHF GUN READER's availability by checking the power

Syntax

```
BOOL UHF_IsReady();
```

Parameters

None

Return Value

TRUE = Success

FALSE = Fail

Remarks

Following cases will return false:

1. PDA detached from gun reader
2. RFID / SCAN switch is at SCAN position
3. Gun reader's battery is detached or empty
4. PDA in gun reader is synced with PC

See Also

None

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : bool IsReady();

Example

```

if(!UHF_IsReady())
    return;

```

4.7.3 UHF_Init

Description

Initialize RFID

Syntax

```
RFID_STATUS UHF_Init (HWND hDlgWnd);
```

Parameters

hDlgWnd

Reader power status message is passed to registered Windows Handle

Return Value

Returns RFID_STATUS of enum type

Remarks

When initializing, WM_MSG_POWER message is passed to registered Windows Handle. On change of power status, this message will return the status. If FALSE is returned, reader should be closed using UHF_Close. If it changes to TRUE, then UHF_Init should be used to re-initialize.

See Also

UHF_Close

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : RFIDUHF.RFID_STATUS Init();

Example

```
RFID_STATUS status = UHF_Init(m_hWnd);  
if(status != RFID_STATUS_OK)  
{  
    strstatus.Format(_T("RFID Init Error-[%x]"),status);  
    return FALSE;  
}
```

4.7.4 UHF_Close

Description

Close RFID

Syntax

```
RFID_STATUS UHF_Close();
```

Parameters

None

Return Value

Returns RFID_STATUS of enum type

Remarks

None

See Also

UHF_Init

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : RFIDUHF.RFID_STATUS Close();

Example

```
RFID_STATUS status;  
CString strstatus;  
status = UHF_Close();  
if(status != RFID_STATUS_OK)  
{  
    strstatus.Format(_T("%x"),status);  
    return FALSE;  
}
```

4.7.5 UHF_Inventory

Description

Start Inventory by reading EPC data

Syntax

```
void UHF_Inventory(HWND hWnd);
```

Parameters

hWnd

Register Windows Handle to receive data through inventory

Return Value

None

Remarks

EPC data is passed to registered Windows Handle through WM_MSG_INVENTORY

See Also

UHF_InventoryStop

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : void Inventory();

Example

```
UHF_Inventory(m_hWnd);
```

4.7.6 UHF_InventoryStop

Description

Stop Inventory

Syntax

```
void UHF_InventoryStop();
```

Parameters

None

Return Value

None

Remarks

None

See Also

UHF_Inventory

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : void InventoryStop();

Example

```
UHF_InventoryStop();
```

4.7.7 UHF_Read

Description

Read memory from tag

Syntax

```
RFID_STATUS UHF_Read(RFIDReadCmd *cmd);
```

Parameters

cmd

RFIDReadCmd structure is used to assign tag memory location

Return Value

Returns RFID_STATUS of enum type

Remarks

Windows Handle for receiving data reading message must be registered to hWnd of RFIDReadCmd structure. Data can be obtained using UHF_GetData once WM_MSG_ACCESS message is received.

See Also

UHF_GetData

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : RFIDUHF.RFID_STATUS Read(ref RFIDUHF.RFIDReadCmd cmd);

Example

```
RFID_STATUS status;
RFIDReadCmd readcmd;
INT32U accpwd = 0;
accpwd = wcstoul(m_strRWPwd, 0, 16);
readcmd.hWnd = this->m_hWnd;
readcmd.bank = (RFIDTagBank)m_cmbBank.GetCurSel();
readcmd.wlength = (INT8U)_ttoi(m_strLength);
readcmd.offset = (INT8U)_ttoi(m_strOffset);
readcmd.accpwd = accpwd;
status = UHF_Read(&readcmd);
if (RFID_STATUS_OK != status)
{
    // Fail!!
}
```

4.7.8 UHF_Write

Description

Write data to tag

Syntax

RFID_STATUS UHF_Write(RFIDWriteCmd *cmd);

Parameters

cmd

RFIDWriteCmd structure is used to assign tag memory location

Return Value

Returns RFID_STATUS of enum type

Remarks

Windows Handle for receiving data reading message must be registered to hWnd of RFIDWriteCmd structure. Result can be obtained using UHF_GetError once WM_MSG_ACCESS message is received.

See Also

UHF_GetError

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : RFIDUHF.RFID_STATUS Write(ref RFIDUHF.RFIDWriteCmd cmd);

Example

```
RFID_STATUS status;

RFIDWriteCmd writecmd;

WCHAR *str;

char str1[32][8] = {0,};

INT16U wdata[32] = {0,};

INT8U wlength;

INT8U offset;

INT32U accpwd = 0;

wlength = (INT8U)_ttoi(L"6");

offset = (INT8U)_ttoi(L"2");

accpwd = wcstoul(L"00000000", 0, 16);

str = (WCHAR*)((LPCWSTR)m_strWriteData); // 6 Word – 12 Byte

for(int i=0;i<wlength;i++)
{
    sprintf(str1[i], "%c%c%c%c", str[i*4], str[i*4+1], str[i*4+2], str[i*4+3]);
    wdata[i] = (INT16U)strtoul(str1[i], 0, 16);
}

writecmd.hWnd = this->m_hWnd;

writecmd.bank = (RFIDTagBank)m_cmbBank.GetCurSel();

writecmd.wlength = wlength;

writecmd.offset = offset;

memcpy(writecmd.wdata, wdata, wlength*2);

writecmd.accpwd = accpwd;

status = UHF_Write(&writecmd);

if (RFID_STATUS_OK != status)
{
    MessageBox(_T("RFID Write Fail!"));
```


}

4.7.9 UHF_Lock

Description

Limit access to tag

Syntax

```
RFID_STATUS UHF_Lock(RFIDLockCmd *cmd);
```

Parameters

cmd

RFIDLockCmd structure is used to limit access to tag

Return Value

Returns RFID_STATUS of enum type

Remarks

Windows Handle for receiving data reading message must be registered to hWnd of RFIDLockCmd structure. Result can be obtained using UHF_GetError once WM_MSG_ACCESS message is received.

Access password in reserved memory is used to limit access.

RFIDLockPermission Enum value in RFIDLockCmd structure is used.

PERMISSION_ACCESSIBLE:

Read and write of password is possible. Change permission is also possible.

PERMISSION_ALWAYS_ACCESSIBLE: Read and write of password is possible. But, change permission is not possible.

PERMISSION_SECURED_ACCESSIBLE: Read and write of password is not possible. But, change permission is possible.

PERMISSION_ALWAYS_NOT_ACCESSIBLE: Read and write of password is not possible. Change permission is also not possible.

Read / Write accessibility setting is possible in reserved area. EPC, USER area only allow setting for write, where read is always possible. TID is read only.

See Also

UHF_GetError

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : RFIDUHF.RFID_STATUS Lock(ref RFIDUHF.RFIDLockCmd cmd);

Example

```
RFID_STATUS status;
```

```
RFIDLockCmd lock;
```

```
INT32U accpwd;
```

```

accpwd = wcstoul(m_strLockPwd, 0, 16);

lock.hWnd = this->m_hWnd;
lock.lockkillpwd = (RFIDLockPermission)m_cmbKillPwd.GetCurSel();
lock.lockaccesspwd = (RFIDLockPermission)m_cmbAccPwd.GetCurSel();
lock.lockepc = (RFIDLockPermission)m_cmbEPC.GetCurSel();
lock.locktid = (RFIDLockPermission)m_cmbTID.GetCurSel();
lock.lockuser = (RFIDLockPermission)m_cmbUser.GetCurSel();
lock.accpwd = accpwd;

status = UHF_Lock(&lock);

if(RFID_STATUS_OK != status)
{
    MessageBox(_T("RFID Lock Fail!"));
}

```

4.7.10 UHF_Kill

Description

Kill the tag (disabling the tag)

Syntax

```
RFID_STATUS UHF_Kill(RFIDKillCmd *cmd);
```

Parameters

cmd

RFIDKillCmd structure is used to disable tag

Return Value

Returns RFID_STATUS of enum type

Remarks

Windows Handle for receiving data reading message must be registered to hWnd of RFIDKillCmd structure. Result can be obtained using UHF_GetError once WM_MSG_ACCESS message is received.

See Also

UHF_GetError

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : RFIDUHF.RFID_STATUS Kill(ref RFIDUHF.RFIDKillCmd cmd);

Example

```
RFID_STATUS status;
RFIDKillCmd kill;
INT32U killpwd;
killpwd = wcstoul(m_strKillPwd, 0, 16);
kill.hWnd = this->m_hWnd;
kill.killpwd = killpwd;
status = UHF_Kill(&kill);
if(RFID_STATUS_OK != status)
{
    MessageBox(_T("RFID Kill Fail!"));
}
```

4.7.11 UHF_GetData

Description

Get tag data from Inventory and Read

Syntax

```
int UHF_GetData (INT8U *data);
```

Parameters

**data*

[out] get current data

Return Value

Returns the length of the data

Remarks

Data obtained by UHF_Inventory or UHF_Read can be checked. If return value is 0, error can be checked using UHF_GetError.

See Also

UHF_Inventory, UHF_Read, UHF_GetError

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : int GetData(StringBuilder strTagData);

Example

```
int length = 0;
unsigned char data[128] = {0,};
```

```

RFIDErrorCode err;
CString str;
length = UHF_GetData(data);
if(length == 0)
{
    err = UHF_GetError();

}
else
{
    for(int i=0;i<length;i++)
    {
        str.AppendFormat(_T("%02X"), data[i]);
    }
    m_strReadData = str;
}

```

4.7.12 UHF_SetRegionFrequency

Description

Set RFID frequency to suit regional regulation

Syntax

```
RFID_STATUS UHF_SetRegionFrequency(RFIDRegion region);
```

Parameters

region

Set regional frequency by assigning Enum type variable

Return Value

Returns RFID_STATUS of enum type

Remarks

None

See Also

UHF_GetRegionFrequency

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : RFIDUHF.RFID_STATUS SetRegionFrequency(RFIDUHF.RFIDRegion region);

Example

```
RFIDRegion region = RFID_REGION_EURO_NEW;  
RFID_STATUS status = UHF_SetRegionFrequency(region);  
if(status != RFID_STATUS_OK)  
{  
    MessageBox(_T("UHF_SetRegionFrequency Fail!"));  
    return ;  
}
```

4.7.13 UHF_GetRegionFrequency

Description

Check current regional frequency setting

Syntax

```
RFIDRegion UHF_GetRegionFrequency();
```

Parameters

None

Return Value

Returns RFID_STATUS of enum type

Remarks

None

See Also

UHF_SetRegionFrequency

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : RFIDUHF.RFIDRegion GetRegionFrequency();

Example

```
RFIDRegion region = UHF_GetRegionFrequency();
```

4.7.14 UHF_ReadBattery

Description

Check battery voltage and ADC value

Syntax

```
RFID_STATUS UHF_ReadBattery(INT32U *nADCValue, float *fVolt);
```

Parameters

**nADCValue*

[out] Check ADC value of battery

**fVolt*

. [out] Check current battery voltage

Return Value

Returns RFID_STATUS of enum type

Remarks

None

See Also

UHF_ReadBatteryStatus

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : RFIDUHF.RFID_STATUS ReadBattery(ref uint nADCValue, ref float fVolt);

Example

```
RFID_STATUS status;
```

```
INT32U nADC = 0;
```

```
float fVolt;
```

```
BATTERY_STATUS battstatus;
```

```
UHF_ReadBattery(&nADC, &fVolt);
```

```
status = UHF_ReadBatteryStatus(&battstatus);
```

```
m_strStatus.Format(_T("%.2f(%d)"),fVolt, battstatus);
```

4.7.15 UHF_ReadBatteryStatus

Description

Check battery level (0~4 level)

Syntax

```
RFID_STATUS UHF_ReadBatteryStatus(BATTERY_STATUS *step);
```

Parameters

**step*

[out] Check battery level through Enum type BATTERY_STATUS

Return Value

Returns RFID_STATUS of enum type

Remarks

None

See Also

UHF_ReadBattery

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : RFIDUHF.RFID_STATUS ReadBatteryStatus(ref RFIDUHF.BATTERY_STATUS step);

Example

```
RFID_STATUS status;
INT32U nADC = 0;
float fVolt;
BATTERY_STATUS battstatus;
UHF_ReadBattery(&nADC, &fVolt);
status = UHF_ReadBatteryStatus(&battstatus);
m_strStatus.Format(_T("%.2f(%d)"),fVolt, battstatus);
```

4.7.16 UHF_Version

Description

Check DLL and F/W version

Syntax

```
BOOL UHF_Version(RFID_VERSION *LibVer, RFID_VERSION *MacVer, TCHAR *tzDllVersion);
```

Parameters

**LibVer*

[out] Check version of NRFMCCE.dll

**MacVer*

[out] Check reader firmware version

**tzDllVersion*

[out] Check RFDI_UHF.dll version

Return Value

TRUE = Success

FALSE = Fail

Remarks

None

See Also

None

For .Net

Namespace : RFID_UHF_Net.RFIDUHF

Function : bool Version(ref RFIDUHF.RFID_VERSION LibVer, ref RFIDUHF.RFID_VERSION MacVer, StringBuilder strDllVersion);

Example

```
RFID_VERSION LibVer;
```

```
RFID_VERSION MacVer;
```

```
TCHAR tzDllVersion[260] = {0x00};
```

```
CString strstatus;
```

```
UHF_Version(&LibVer, &MacVer, tzDllVersion);
```

```
strstatus.Format(_T("Firmware: %u.%u.%u APP: %s"), MacVer.major, MacVer.minor, MacVer.maintenance, VER_APP);
```


4.8 WLAN

Status Return value

Please refer to the below table for the status value definition.

Status Value Definition	Code	Meaning
TRUE	1	Success
FALSE	0	General Error

Define
#define MAX_AP 30
Enum
<pre>typedef enum { SUCCESS=0, FAIL, INVALID_NAME, INVALID_CONFIG, INVALID_DELETE, POWERCYCLE_REQUIRED, INVALID_PARAMETER, INVALID_EAP_TYPE, INVALID_WEP_TYPE, INVALID_FILE }WLAN_ERROR_RESULT;</pre>
Structure
<pre>typedef struct _WLAN_STATUS { DWORD channel; int rssi; double bitRate; int txPower; DWORD DTIM; DWORD beaconPeriod; DWORD beaconsReceived; TCHAR SSID[32]; TCHAR CardState[36]; byte client_MAC[6]; byte client_IP[4]; byte AP_MAC[6]; byte AP_IP[4]; }WLAN_STATUS, *P_WLAN_STATUS;</pre>

```
typedef struct _WLAN_SSID
{
    TCHAR    SSID[32];
    BOOL     Privacy;
    int      Rssi;
} WLAN_SSID;

typedef struct _WLAN_SSID_LIST
{
    int      m_NumberOfItems;
    WLAN_SSID*  m_SSID;
} WLAN_SSID_LIST;

typedef struct _WLAN_CONFIG_NAME
{
    TCHAR    ConfigName[32];
} WLAN_CONFIG_NAME;

typedef struct _WLAN_CONFIG_NAME_LIST
{
    int      m_NumberOfItems;
    WLAN_CONFIG_NAME*  m_ConfigName;
}WLAN_CONFIG_NAME_LIST;
```

Functions for WLAN

Name	Description
WLAN_ActivateConfig	Activate the configuration with the given name.
WLAN_Close	Free WLAN.dll.
WLAN_ConnectAP	Connecting to AP.
WLAN_DeleteConfig	This function deletes the configuration matching 'name'.
WLAN_ExportConfig	This function exports configurations.
WLAN_ImportConfig	This function imports configurations.
WLAN_Init	WLAN.dll initiation
WLAN_GetAllConfigName	This function retrieves all of the configurations.
WLAN_GetCurrentAPInfo	Get current AP information
WLAN_GetBssidList	Scans and lists connectable APs.
WLAN_GetPowerStatus	Get power status.
WLAN_PowerOn	Inserts WLAN card.
WLAN_PowerOff	Removes WLAN card.

4.8.1 WLAN_ActivateConfig

Description

Activate the configuration with the given name.

Syntax

```
WLAN_API BOOL WLAN_ActivateConfig(TCHAR* szName);
```

Parameters

szName

Name of the configuration to make the active one.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

This function succeeds even if the card is not present so, when it is inserted, this becomes the active configuration.

See Also

None

For .Net

Namespace : WLANNet.WLANNet

Function : bool ActivateConfig(string ConfigName)

Example

```
if(WLAN_ActivateConfig("Config Name") == FALSE)
{
    AfxMessageBox(_T("Fail To WLAN_ActivateConfig()"));
}
```

4.8.2 WLAN_Close

Description

Free WLAN module.

Syntax

```
WLAN_API BOOL WLAN_Close();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

WLAN_Init

For .Net

Namespace : WLANNet.WLANNet

Function : bool Close()

Example

```
if(WLAN_Close() == FALSE)
{
    AfxMessageBox(_T("Fail To WLAN_Close()");
}
```

4.8.3 WLAN_ConnectAP

Description

Connecting to AP.

Syntax

WLAN_API int WLAN_ConnectAP(TCHAR* szBssid, TCHAR* szPassword, int iEncryptionType);

Parameters

szBssid

Name of the configuration to connect the one.

szPassword

Password.

iEncryptionType

Type	Description
0	Open
1	WEP
2	WPA PSK
3	WPA2 PSK

Return Value

Return WLAN_ERROR_RESULT

Remarks

None

See Also

None

For .Net

Namespace : WLANNet.WLANNet

Function : bool ConnectAP(string szBssid, string password, int encryptionType)

Example

```
int result=WLAN_ConnectAP(_T("SSID NAME"), _T("Password"), EncryptionType);  
if(result==0)  
    AfxMessageBox(_T("Success To WLAN_ConnectAP()"));
```

4.8.4 WLAN_ConnectAPEX

Description

Connecting to AP.

Syntax

WLAN_API int WLAN_ConnectAPEX(TCHAR* szBssid, TCHAR* szPassword, int iEncryptionType , int iWepKeyType);

Parameters

szBssid

Name of the configuration to connect the one.

szPassword

Password.

iEncryptionType

Type	Description
0	Open
1	WEP
2	WPA PSK
3	WPA2 PSK

iWepKeyType

iWepKeyType of default parameter is zero.

Type	Description
0	Not Set
1	40bit
2	128bit

Return Value

Return WLAN_ERROR_RESULT

Remarks

None

See Also

None

For .Net

Namespace : WLANNet.WLANNet

Function : bool ConnectAPEX(string szBssid, string password, int encryptionType, int WepKeyType)

Example

```
int result=WLAN_ConnectAPEX(_T("SSID NAME"), _T("Password"), EncryptionType, 2);  
if(result==0)  
    AfxMessageBox(_T("Success To WLAN_ConnectAPEX()"));
```

4.8.5 WLAN_DeleteConfig

Description

This function deletes the configuration matching 'name'.

Syntax

```
WLAN_API BOOL WLAN_DeleteConfig(TCHAR *szConfigName);
```

Parameters

szConfigName

Name of the configuration to delete the one.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

You are not allowed to delete the active configuration.

See Also

None

For .Net

Namespace : WLANNet.WLANNet

Function : bool ActivateConfig(string ConfigName)

Example

```
If(WLAN_DeleteConfig(_T("SSID NAME"))==0)  
    AfxMessageBox(_T("Fail To WLAN_ DeleteConfig ( )"));
```

4.8.6 WLAN_ExportConfig

Description

This function exports configurations.

Syntax

```
WLAN_API BOOL WLAN_ExportConfig(TCHAR* szExportPath);
```

Parameters

szExportPath

File name and route to be stored.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

WLAN_ImportConfig

For .Net

Namespace : WLANNet.WLANNet

Function : bool ExportConfig (string ExportPath)

Example

```
If(WLAN_ExportConfig(_T("Flash Disk\\WLAN\\Summit.sdc"))==0)
    AfxMessageBox(_T("Fail To WLAN_ExportConfig ("));
```

4.8.7 WLAN_ImportConfig

Description

This function imports configurations.

Syntax

```
WLAN_API BOOL WLAN_ImportConfig(TCHAR* szImportPath);
```

Parameters

szImportPath

File name and route to get.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

WLAN_ExportConfig

For .Net

Namespace : WLANNet.WLANNet

Function : bool ImportConfig (string ImportPath)

Example

```
If(WLAN_ImportConfig(_T("Flash Disk\\WLAN\\Summit.sdc")==0)
    AfxMessageBox(_T("Fail To WLAN_ ImportConfig (")");
```

4.8.8 WLAN_Init

Description

WLAN module initiation.

Syntax

WLAN_API BOOL WLAN_Init();

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

WLAN_Close

For .Net

Namespace : WLANNet.WLANNet

Function : bool Init ()

Example

```
If(WLAN_Init()==0)
    AfxMessageBox(_T("Fail To WLAN_ Init());
```

4.8.9 WLAN_GetAllConfigName

Description

This function retrieves all of the configurations.

Syntax

WLAN_API int WLAN_GetAllConfigName(WLAN_CONFIG_NAME_LIST* pstConfigList);

Parameters

WLAN_CONFIG_NAME_LIST* pstConfigList.

Return Value

Return WLAN_ERROR_RESULT

Remarks

Except ThirdPartyConfig.

See Also

None

For .Net

Namespace : WLANNet.WLANNet

Function : int GetAllConfigName ()

Example

```
WLAN_CONFIG_NAME_LIST* NameList=new WLAN_CONFIG_NAME_LIST();  
int resut=WLAN_GetAllConfigName(NameList);  
if(result==0)  
    AfxMessageBox(_T("Success To WLAN_GetAllConfigName()"));
```

4.8.10 WLAN_GetCurrentAPInfo

Description

Get current AP information except txPower .

Syntax

```
WLAN_API int WLAN_GetCurrentAPInfo(WLAN_STATUS* pstStatus);
```

Parameters

WLAN_STATUS* pstStatus

Pointer to a WLAN_STATUS structure to be filled in with AP info parameters.

Return Value

Return WLAN_ERROR_RESULT.

Remarks

None

See Also

None

For .Net

Namespace : WLANNet.WLANNet

Function : int GetCurrentAPInfo (ref WLAN_STATUS st)

Example

```

WLAN_STATUS st;
memset(&st, 0, sizeof(st));
int result=WLAN_GetCurrentAPInfo(&st);
if(result==0)
    AfxMessageBox(_T("Success To WLAN_GetCurrentAPInfo"));

```

4.8.11 WLAN_GetBssidList

Description

Scans and lists connectable APs.

Syntax

```
WLAN_API BOOL WLAN_GetBssidList(WLAN_SSID_LIST* pstSsidList);
```

Parameters

WLAN_SSID_LIST pstSsidList*

Pointer to a WLAN_SSID_LIST structure to be filled in SSIDs.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : WLANNet.WLANNet

Function : int GetBssidList (ref WLAN_SSID[] ssidlist)

Example

```

WLAN_SSID_LIST* SSIDList=new WLAN_SSID_LIST();
SSIDList->m_SSID=new WLAN_SSID[40];
if(WLAN_GetBssidList(SSIDList)==0)
    AfxMessageBox(_T("Fail To WLAN_GetBssidList()");

```

4.8.12 WLAN_GetPowerStatus

Description

Get power status.

Syntax

```
WLAN_API BOOL WLAN_GetPowerStatus();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : WLANNet.WLANNet

Function : int GetPowerStatus ()

Example

```
if(WLAN_GetPowerStatus())  
    AfxMessageBox(_T("Power On!"));  
else  
    AfxMessageBox(_T("Power On!"));
```

4.8.13 WLAN_PowerOn

Description

Inserts WLAN card.

Syntax

WLAN_API BOOL WLAN_PowerOn();

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

WLAN_PowerOff

For .Net

Namespace : WLANNet.WLANNet

Function : bool PowerOn ()

Example

```
If(WLAN_PowerOn()==0)
    AfxMessageBox(_T("Fail To WLAN_PowerOn()"));
```

4.8.14 WLAN_PowerOff

Description

Removing WLAN card.

Syntax

```
WLAN_API BOOL WLAN_PowerOff();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

WLAN_PowerOn

For .Net

Namespace : WLANNet.WLANNet

Function : bool PowerOff ()

Example

```
If(WLAN_PowerOff()==0)
    AfxMessageBox(_T("Fail To WLAN_PowerOff()"));
```

4.9 BLUETOOTH

Status Return value & API Error Codes

Please refer to the below table for the status value definition.

Status Value Definition	Code	Meaning
TRUE	1	Success
FALSE	0	General Error

Define
<pre>// Message Queue Commands #define PING 0 #define BTP_FIND_FIRST_DEVICE 1 #define BTP_FIND_NEXT_DEVICE 2 #define BTP_FIND_DEVICE_CLOSE 3 #define BTP_FIND_FIRST_SERVICE 4 #define BTP_FIND_NEXT_SERVICE 5 #define BTP_FIND_SERVICE_CLOSE 6 #define BTP_FIND_FIRST_CONNECTION 7 #define BTP_FIND_NEXT_CONNECTION 8 #define BTP_FIND_CONNECTION_CLOSE 9 #define BTP_CREATE_CONNECTION 10 #define BTP_DELETE_CONNECTION 11 #define BTP_CONNECT 12 #define BTP_DISCONNECT 13 #define BTP_SET_AUTHENTICATION_CALLBACK 14 #define BTP_SET_CONNECTION_CALLBACK 15 #define BTP_FIND_LOCAL_DEVICE 16 #define BTP_SET_INCOMING_PIN 17 #define BTP_SET_OUTGOING_PIN 18 #define BTP_PERFORM_ACTION 19 #define BTP_SET_SECURITY_MODE 20 #define BTP_GET_SECURITY_MODE 21 #define BTP_SET_SCO_CONNECTION_STATE 22 #define BTP_GET_SCO_CONNECTION_STATE 23 // Maximum message size allowed by the message queues. #define MAX_MSG_SIZE 1024 #define MAX_NAME_LENGTH 248 // Maximum name length, including delimiter. #define BLUETOOTH_MAX_NAME_SIZE (MAX_NAME_LENGTH + 1) // Error codes. #define BTP_ERROR 0// Indicates the command executed successfully. #define BTP_ERROR_SUCCESS 1// Indicates that the command was unsuccessful; used when no other more specific error code is applicable. #define BTP_ERROR_INVALID_PARAMETER 2// One of the parameters is not valid.</pre>

<pre> #define BTP_ERROR_INVALID_HANDLE 3// A handle is not valid. #define BTP_ERROR_NO_MORE 4// Indicates there are no more elements in the list. #define BTP_ERROR_MSG_SEND 5// Indicates sending of the command to BTE Explorer failed. #define BTP_ERROR_MSG_RECEIVE 6// Indicates BTE Explorer failed to respond to the command. #define BTP_ERROR_NO_COM_PORT 7// Indicates there are no available COM ports with which to connect. #define BTP_ERROR_BTEXP_NOT_RUNNING 8// Indicates BTE Explorer could not be started. #define BTP_ERROR_NO_KEY_AVAILABLE 9// Returned by authentication callbacks, indicating the callback cannot provide the pass key for the specified device. // Device States #define BTP_DEVICE_STATE_OFF 0 #define BTP_DEVICE_STATE_ON 1 // SCO Connection States #define BTP_SCO_STATE_DISCONNECTED 0 #define BTP_SCO_STATE_CONNECTED 1 //BTP_Perform_Action API #define BTP_ACTION_SHOW_SETTINGS 0x00000001 #define BTP_ACTION_DELETE_ALL_DEVICES 0x00000002 // Searching for remote devices #define BTP_DEVICE_NONE 0x0001 #define BTP_DEVICE_AUTHENTICATED 0x0002 #define BTP_DEVICE_REMEMBERED 0x0004 #define BTP_DEVICE_CONNECTED 0x0008 #define BTP_DEVICE_ALL 0x8000 // Flags for the BTP_Connection_Query_t structure. #define BTP_CONNECTION_NONE 0 #define BTP_CONNECTION_REMEMBERED 1 #define BTP_CONNECTION_ACTIVE 2 #define BTP_CONNECTION_ALL 3 #define MESSAGE_TO_BTE_HEADER_SIZE 12 #define MESSAGE_FROM_BTE_HEADER_SIZE 4 //ListType #define DeviceFind 1 #define ServiceFind 2 #define ConnectionFind 3 </pre>	
Enum	
<pre> typedef enum { BTP_PROFILE_SPP, BTP_PROFILE_DUN, BTP_PROFILE_FAX, BTP_PROFILE_LAN, BTP_PROFILE_FILE_TRANSFER, BTP_PROFILE_HEADSET, BTP_PROFILE_HEADSET_AUDIO_GATEWAY, BTP_PROFILE_HANDS_FREE, </pre>	

```

    BTP_PROFILE_HANDS_FREE_AUDIO_GATEWAY,
    BTP_PROFILE_HID_HOST,
    BTP_PROFILE_HID_DEVICE,
    BTP_PROFILE_UNKNOWN = -1
} BTP_Profile_Type;

typedef enum
{
    bdAll, //returns all devices
    bdHID //returns only HID devices
} BTP_DeviceType_t;

typedef enum
{
    deNIL,
    deNULL,
    deUnsignedInteger1Byte,
    deUnsignedInteger2Bytes,
    deUnsignedInteger4Bytes,
    deUnsignedInteger8Bytes,
    deUnsignedInteger16Bytes,
    deSignedInteger1Byte,
    deSignedInteger2Bytes,
    deSignedInteger4Bytes,
    deSignedInteger8Bytes,
    deSignedInteger16Bytes,
    deTextString,
    deBoolean,
    deURL,
    deUUID_16,
    deUUID_32,
    deUUID_128,
    deSequence,
    deAlternative
} SDP_Data_Element_Type_t;

typedef enum
{
    BTP_AUTHENTICATION_CALLBACK,
    BTP_CONNECTION_CALLBACK
} BTP_Callback_Type;

typedef enum
{
    BTP_SECURITYMODE_NONE,
    BTP_SECURITYMODE_AUTHENTICATE,
    BTP_SECURITYMODE_AUTHENTICATE_AND_ENCRYPT
} BTP_Security_Mode_Type;

```

Structure


```

typedef struct
{
    Byte_t BD_ADDR0;
    Byte_t BD_ADDR1;
    Byte_t BD_ADDR2;
    Byte_t BD_ADDR3;
    Byte_t BD_ADDR4;
    Byte_t BD_ADDR5;
} BD_ADDR_t;

typedef struct
{
    SDP_Data_Element_Type_t Data_Element_Type;
    union
    {
        {
            UUID_16_t    UUID_16;
            UUID_32_t    UUID_32;
            UUID_128_t   UUID_128;
        } UUID_Value;
    }
} SDP_UUID_Entry_t;

typedef struct
{
    Byte_t UUID_Byte0;
    Byte_t UUID_Byte1;
} UUID_16_t;

typedef struct
{
    Byte_t UUID_Byte0;
    Byte_t UUID_Byte1;
    Byte_t UUID_Byte2;
    Byte_t UUID_Byte3;
} UUID_32_t;

typedef struct
{
    Byte_t UUID_Byte0;
    Byte_t UUID_Byte1;
    Byte_t UUID_Byte2;
    Byte_t UUID_Byte3;
    Byte_t UUID_Byte4;
    Byte_t UUID_Byte5;
    Byte_t UUID_Byte6;
    Byte_t UUID_Byte7;
    Byte_t UUID_Byte8;
    Byte_t UUID_Byte9;
    Byte_t UUID_Byte10;
    Byte_t UUID_Byte11;
}

```

```

    Byte_t  UUID_Byte12;
    Byte_t  UUID_Byte13;
    Byte_t  UUID_Byte14;
    Byte_t  UUID_Byte15;
} UUID_128_t;

typedef struct
{
    Word_t  DeviceAttributes;
    Byte_t  InquiryTimeout;
} BTP_Device_Query_t;

typedef struct _tagBTP_Device_Query_Ex_t
{
    unsigned int      Size;
    Byte_t            Version;
    Word_t            DeviceAttributes;
    Byte_t            InquiryTimeout;
    BTP_DeviceType_t  DeviceType;
} BTP_Device_Query_Ex_t;

typedef struct
{
    BD_ADDR_t          BD_ADDR;
    Class_of_Device_t  ClassOfDevice;
    Word_t            DeviceAttributes;
    char              Name[BLUETOOTH_MAX_NAME_SIZE];
} BTP_Device_Info_t;

typedef struct
{
    BD_ADDR_t          BD_ADDR;
    Word_t            NumberServiceUUID;
    SDP_UUID_Entry_t*  Service;
} BTP_Service_Query_t;

typedef struct
{
    BTP_Profile_Type    ProfileType;
    unsigned char        MajorVersion;
    unsigned char        MinorVersion;
    char                ServiceName[BLUETOOTH_MAX_NAME_SIZE];
    unsigned int         RFCOMMPort;
} BTP_Service_Info_t;

typedef struct _tagBTP_Service_Info_Ex_t
{
    unsigned int      Size;
    unsigned int      Version;
    BTP_Profile_Type  ProfileType;
    unsigned char      MajorVersion;
    unsigned char      MinorVersion;

```

```

char                ServiceName[BLUETOOTH_MAX_NAME_SIZE];
unsigned int        ListIndex;
union
{
    BTPSerialPortProfileInfo_t    RemoteSerialPortProfileInfo;
    BTPHIDProfileInfo_t          RemoteHIDProfileInfo;
}ProfileInformation;
} BTP_Service_Info_Ex_t;

```

```

typedef struct _tagBTPSerialPortProfileInfo_t

```

```

{
    unsigned int    RFCOMMServerPort;
    bool            UseActiveSync;
} BTPSerialPortProfileInfo_t;

```

```

typedef struct _tagBTPHIDProfileInfo_t

```

```

{
    unsigned int    L2CAPControlChannel;
    unsigned int    L2CAPInterruptChannel;
    Byte_t          DeviceSubclass;
    bool            VirtualCableSupported;
    bool            DeviceAutomaticReconnect;
    bool            DeviceNormallyConnectable;
} BTPHIDProfileInfo_t;

```

```

typedef struct

```

```

{
    Word_t    ConnectionAttributes;
} BTP_Connection_Query_t;

```

```

typedef struct

```

```

{
    BTP_Connection_ID    ConnectionID;
    BD_ADDR_t            BD_ADDR;
    unsigned int          RFCOMMPort;
    int                   LocalCOMPort;
    unsigned char          MajorVersion;
    unsigned char          MinorVersion;
    Word_t                ConnectionAttributes;
    BTP_Profile_Type       ProfileType;
} BTP_Connection_Info_t;

```

```

typedef struct _tagBTP_Connection_Info_Ex_t

```

```

{
    unsigned int        Size;
    unsigned int        Version;
    BTP_Connection_ID    ConnectionID;
    BD_ADDR_t            BD_ADDR;
    int                 LocalCOMPort;
    unsigned char        MajorVersion;
    unsigned char        MinorVersion;

```

```

Word_t          ConnectionAttributes;
BTP_Profile_Type ProfileType;
HLOCAL          ListHandle;
unsigned int     ListIndex;
union
{
    BTPSerialPortProfileInfo_t RemoteSerialPortProfileInfo;
    BTPHIDProfileInfo_t        RemoteHIDProfileInfo;
}ProfileInformation;
} BTP_Connection_Info_Ex_t;
typedef struct _tagBTPSerialPortProfileInfo_t
{
    unsigned int   RFCOMMServerPort;
    bool           UseActiveSync;
} BTPSerialPortProfileInfo_t;
typedef struct _tagBTPHIDProfileInfo_t
{
    unsigned int   L2CAPControlChannel;
    unsigned int   L2CAPInterruptChannel;
    Byte_t         DeviceSubclass;
    bool           VirtualCableSupported;
    bool           DeviceAutomaticReconnect;
    bool           DeviceNormallyConnectable;
} BTPHIDProfileInfo_t;
typedef struct _tagBTP_PIN_t
{
    unsigned int   PINLength;
    BTP_PIN_Code_t PINCode;
} BTP_PIN_t;
typedef struct _tagBTP_PIN_Code_t
{
    Byte_t PIN_Code0;
    Byte_t PIN_Code1;
    Byte_t PIN_Code2;
    Byte_t PIN_Code3;
    Byte_t PIN_Code4;
    Byte_t PIN_Code5;
    Byte_t PIN_Code6;
    Byte_t PIN_Code7;
    Byte_t PIN_Code8;
    Byte_t PIN_Code9;
    Byte_t PIN_Code10;
    Byte_t PIN_Code11;
    Byte_t PIN_Code12;
    Byte_t PIN_Code13;
    Byte_t PIN_Code14;

```

```
    Byte_t PIN_Code15;  
} BTP_PIN_Code_t;
```

Functions for Bluetooth

Name	Description
BLUETOOTH_Close	Performs cleanup after the API is no longer needed.
BLUETOOTH_Connect	Activates the specified connection in the connection list.
BLUETOOTH_CreateConnection	Defines a new connection, adding it to a list of connections.
BLUETOOTH_CreateConnectionEx	Defines a new connection, adding it to a list of connections.
BLUETOOTH_DeleteConnection	Deletes a connection from the list of connections.
BLUETOOTH_Disconnect	Disconnects from the specified connection.
BLUETOOTH_FindConnectionClose	Frees remote resources.
BLUETOOTH_FindDeviceClose	Frees remote resources.
BLUETOOTH_FindFirstConnection	Finds the first connection meeting the specified criteria.
BLUETOOTH_FindFirstConnectionEx	Finds the first connection meeting the specified criteria.
BLUETOOTH_FindFirstDevice	Performs a GAP inquiry to discover devices, returning the first device meeting the specified criteria.
BLUETOOTH_FindFirstDeviceEx	Performs a GAP inquiry to discover devices, returning the first device meeting the specified criteria. This supports searching for a device of specific type.
BLUETOOTH_FindFirstService	Performs an SDP query to discover remote services for a specified device, returning the first service in the list.
BLUETOOTH_FindFirstServiceEx	Performs an SDP query to discover remote services for a specified device, returning the first service in the list.
BLUETOOTH_FindLocalDevice	Returns device information for the local device.
BLUETOOTH_FindNextConnection	Returns the next connection meeting the criteria specified in the call to BLUETOOTH_FindFirstConnection .
BLUETOOTH_FindNextConnectionEx	Returns the next connection meeting the criteria specified in the call to BLUETOOTH_FindFirstConnection .
BLUETOOTH_FindNextDevice	Returns the next device in the list meeting the initial criteria, specified in the call to BLUETOOTH_FindFirstDevice .
BLUETOOTH_FindNextService	Returns the next service in the list.
BLUETOOTH_FindNextServiceEx	Returns the next service in the list.
BLUETOOTH_FindServiceClose	Frees remote resources.
BLUETOOTH_GetBluetoothState	Gets the Bluetooth device state.
BLUETOOTH_GetSCOConnectionState	Gets the SCO connection state.
BLUETOOTH_GetSecurityMode	Gets the current authentication and encryption settings.
BLUETOOTH_Open	Initializes the BTE Explorer API module.
BLUETOOTH_PerformAction	Triggers BTE Explorer to take the requested action.
BLUETOOTH_SetAuthenticationCallback	Sets the authentication callback for the specified device.

BLUETOOTH_SetBluetoothState	Sets the Bluetooth device state to either on or off.
BLUETOOTH_SetConnectionCallback	Sets the connection callback for the specified connection.
BLUETOOTH_SetIncomingPIN	Sets or clears a static PIN to be used for any incoming Connections.
BLUETOOTH_SetOutgoingPIN	Sets or clears a static PIN to be used for any outgoing Connections.
BLUETOOTH_SetSecurityMode	Sets the authentication and encryption settings for future connections.

4.9.1 BLUETOOTH_Close

Description

This function is responsible for cleaning up the module after it is no longer needed by the application.

Syntax

```
BLUETOOTH_API void BLUETOOTH_Close();
```

Parameters

None

Return Value

None

Remarks

None

See Also

BLUETOOTH_Open

For .Net

Namespace : BlueToothNet Bluetooth

Function : void Close()

Example

```
BLUETOOTH_Close();
```

4.9.2 BLUETOOTH_Connect

Description

This function connects to a connection previously defined by a call to BLUETOOTH_CreateConnection.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_Connect(BTP_Connection_ID ConnectionID);
```

Parameters

ConnectionID

Unique identifier for a connection which was previously defined by a call to BLUETOOTH_CreateConnection and BLUETOOTH_CreateConnectionEx.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR , BTP_ERROR_INVALID_PARAMETER

Remarks

None

See Also

BLUETOOTH_Disconnect

For .Net

Namespace : BluetoothNet Bluetooth

Function : Int32 Connect(BT_Connection_ID ConnectionID)

Example

```
hResult = BLUETOOTH_Connect(m_connectionInfo.ConnectionID);
```

4.9.3 BLUETOOTH_CreateConnection

Description

This function defines a temporary or persistent (Favorite) connection, which may later be connected by a call to BLUETOOTH_Connect (Supports SPP only).

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_CreateConnection(BTP_Connection_Info_t *ConnectionInfo);
```

Parameters

ConnectionInfo

Information defining the new connection. Also, the new connection ID is returned in this structure. This version supports SPP connections only.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_INVALID_PARAMETER

Remarks

None

See Also

BLUETOOTH_DeleteConnection

For .Net

Namespace : BluetoothNet Bluetooth

Function : Int32 CreateConnection(ref BT_Connection_Info_t ConnectionInfo)

Example

None

4.9.4 BLUETOOTH_CreateConnectionEx

Description

This function defines a temporary or persistent (Favorite) connection, which may later be connected by a call to BLUETOOTH_Connect (Supports SPP and HID).

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_CreateConnectionEx(BTP_Connection_Info_Ex_t  
*ConnectionInfo);
```

Parameters

ConnectionInfoEx

Information defining the new connection. Also, the new connection ID is returned in this structure. Currently supports connecting to HID and SPP.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR , BTP_ERROR_INVALID_PARAMETER

Remarks

None

See Also

BLUETOOTH_DeleteConnection

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 CreateConnectionEx(ref BT_Connection_Info_Ex_t ConnectionInfo)

Example

```
hResult = Bluetooth.CreateConnectionEx(ref m_ConnectionInfo);
```

4.9.5 BLUETOOTH_DeleteConnection

Description

This function discards a previously defined connection. After deleting a connection, its connection ID is no longer valid.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_DeleteConnection(BTP_Connection_ID ConnectionID);
```

Parameters

ConnectionID

Unique identifier for a connection which was previously defined by a call to BLUETOOTH_CreateConnection and BLUETOOTH_CreateConnectionEx.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR , BTP_ERROR_INVALID_PARAMETER

Remarks

None

See Also

BLUETOOTH_CreateConnection, BLUETOOTH_CreateConnectionEx

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 DeleteConnection(BT_Connection_ID ConnectionID)

Example

```
hResult = BLUETOOTH_DeleteConnection(m_connectionInfo.ConnectionID);
```

4.9.6 BLUETOOTH_Disconnect

Description

This function disconnects from an active connection. If the connection is not persistent, the connection is also deleted, invalidating its connection ID.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_Disconnect(BTP_Connection_ID ConnectionID);
```

Parameters

ConnectionID

Unique identifier for a connection which was previously defined by a call to BLUETOOTH_CreateConnection and BLUETOOTH_CreateConnectionEx.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR , BTP_ERROR_INVALID_ PARAMETER

Remarks

None

See Also

BLUETOOTH_Connect

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 Disconnect(BT_Connection_ID ConnectionID)

Example

```
hResult = BLUETOOTH_Connect(m_connectionInfo.ConnectionID);
```

4.9.7 BLUETOOTH_FindConnectionClose

Description

This function deletes the remote list of connections and performs any additional cleanup.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindConnectionClose(BTP_Connection_Find ConnectionFind);
```

Parameters

ConnectionFind

Handle to the list connections, originally returned by a call to `BLUETOOTH_FindFirstConnection` or `BLUETOOTH_FindFirstConnectionEx`.

Return Value

`BTP_ERROR_SUCCESS` if successful.

Error codes include the following values: `BTP_ERROR` , `BTP_ERROR_INVALID_HANDLE`

Remarks

None

See Also

`BLUETOOTH_FindFirstConnection`, `BLUETOOTH_FindFirstConnectionEx`

For .Net

Namespace : `BluetoothNet Bluetooth`

Function : `Int32 FindConnectionClose(BT_Connection_Find ConnectionFind)`

Example

```
BLUETOOTH_FindConnectionClose(connectFind);
```

4.9.8 BLUETOOTH_FindDeviceClose

Description

This function deletes the remote list of devices and performs any additional cleanup.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindDeviceClose(BTP_Device_Find DeviceFind);
```

Parameters

DeviceFind

Handle to the list of discovered devices, originally returned by a call to `BLUETOOTH_FindFirstDevice`.

Return Value

`BTP_ERROR_SUCCESS` if successful.

Error codes include the following values: `BTP_ERROR` , `BTP_ERROR_INVALID_HANDLE`

Remarks

None

See Also

`BLUETOOTH_FindFirstDevice`

For .Net

Namespace : `BluetoothNet Bluetooth`

Function : Int32 FindDeviceClose(BT_Device_Find DeviceFind)

Example

```
BLUETOOTH_FindDeviceClose(deviceFind);
```

4.9.9 BLUETOOTH_FindFirstConnection

Description

This function creates a list of active connections and Favorites to traverse, returning the first connection from the list.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindFirstConnection( BTP_Connection_Find *ConnectionFind,  
Connection_Info_t *ConnectionInfo, const BTP_Connection_Query_t *ConnectionQuery);
```

Parameters

ConnectionFind

Information defining the new connection. Also, the new connection ID is returned in this structure. This version supports SPP connections only.

ConnectionInfo

Information defining the first connection from the list.

ConnectionQuery

Provides filtering for the types of connection to be retrieved.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR , BTP_ERROR_NO_MORE, BTP_ERROR_INVALID_PARAMETER

Remarks

None

See Also

BLUETOOTH_FindConnectionClose

For .Net

Namespace : BlueToothNet Bluetooth

```
Function : Int32 FindFirstConnection(ref BT_Connection_Find ConnectionFind,ref  
BT_Connection_Info_t ConnectionInfo, ref BT_Connection_Query_t ConnectionQuery)
```

Example

```
hResult = BLUETOOTH_FindFirstConnection(&connectFind, &connectionInfo, &connectQuery);
```

4.9.10 BLUETOOTH_FindFirstConnectionEx

Description

This function creates a list of active connections and Favorites to traverse, returning the first connection from the list.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindFirstConnectionEx(BTP_Connection_Find *ConnectionFind, BTP_Connection_Info_Ex_t *ConnectionInfoEx, const BTP_Connection_Query_t *ConnectionQuery);
```

Parameters

ConnectionFind

Handle to the list of connections, needed for subsequent calls to BLUETOOTH_FindNextConnectionEx and BLUETOOTH_FindConnectionClose.

ConnectionInfoEx

Information defining the first connection from the list. Supports SPP and HID connections.

ConnectionQuery

Provides filtering for the types of connection to be retrieved.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR , BTP_ERROR_NO_MORE, BTP_ERROR_INVALID_PARAMETER

Remarks

None

See Also

BLUETOOTH_FindConnectionClose

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 FindFirstConnectionEx(ref BT_Connection_Find ConnectionFind,ref BT_Connection_Info_Ex_t ConnectionInfoEx, ref BT_Connection_Query_t ConnectionQuery)

Example

None

4.9.11 BLUETOOTH_FindFirstDevice

Description

This function initializes a GAP inquiry, creating a list of discovered Bluetooth devices. The first device is returned by this function.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindFirstDevice(BTP_Device_Find *DeviceFind, BTP_Device_Info_t *DeviceInfo, const BTP_Device_Query_t *DeviceQuery);
```

Parameters

DeviceFind

Handle to the list of discovered devices, needed for subsequent calls to `BLUETOOTH_FindNextDevice` and `BLUETOOTH_FindDeviceClose`.

DeviceInfo

Information defining the first device.

DeviceQuery

Provides parameters for the GAP Inquiry and filtering for the devices to retrieve.

Return Value

`BTP_ERROR_SUCCESS` if successful.

Error codes include the following values: `BTP_ERROR` , `BTP_ERROR_NO_MORE`, `BTP_ERROR_INVALID_PARAMETER`

Remarks

None

See Also

`BLUETOOTH_FindDeviceClose`

For .Net

Namespace : `BlueToothNet Bluetooth`

Function : `Int32 FindFirstDevice(ref BT_Device_Find DeviceFind, ref BT_Device_Info_t DeviceInfo, ref BT_Device_Query_t DeviceQuery)`

Example

None

4.9.12 BLUETOOTH_FindFirstDeviceEx

Description

This function initializes a GAP inquiry, creating a list of discovered Bluetooth devices of a particular device type or all devices. In this version it supports searching for HID devices. The first device that matches the filter is returned by this function.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindFirstDeviceEx(BTP_Device_Find *DeviceFind,  
BTP_Device_Info_t *DeviceInfo, const BTP_Device_Query_Ex_t *DeviceQuery);
```

Parameters

DeviceFind

Handle to the list of discovered devices, needed for subsequent calls to `BLUETOOTH_FindNextDevice` and `BLUETOOTH_FindDeviceClose`.

DeviceInfo

Information defining the first device.

DeviceQueryEx

Provides parameters for the GAP Inquiry and filtering for the devices to retrieve.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR , BTP_ERROR_NO_MORE, BTP_ERROR_INVALID_PARAMETER

Remarks

None

See Also

BLUETOOTH_FindDeviceClose

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 FindFirstDeviceEx(ref BT_Device_Find DeviceFind, ref BT_Device_Info_t DeviceInfo, ref BT_Device_Query_Ex_t DeviceQuery)

Example

```
hResult = BLUETOOTH_FindFirstDeviceEx(&deviceFind, &deviceInfo, &deviceQuery);
```

4.9.13 BLUETOOTH_FindFirstService

Description

This function performs an SDP service discovery on the specified device, return the first service from the resulting list.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindFirstService(BTP_Service_Find *ServiceFind,  
BTP_Service_Info_t *ServiceInfo, const BTP_Service_Query_t *ServiceQuery);
```

Parameters

ServiceFind

Handle to the list of remote services, needed for subsequent calls to BLUETOOTH_FindNextService and BLUETOOTH_FindServiceClose.

ServiceInfo

Information defining the first remote service in the list.

ServiceQuery

Provides parameters for the SDP query.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_NO_MORE, BTP_ERROR_INVALID_HANDLE

Remarks

None

See Also

BLUETOOTH_FindServiceClose

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 FindFirstService(ref BT_Service_Find ServiceFind, ref BT_Service_Info_t ServiceInfo, ref BT_Service_Query_t ServiceQuery)

Example

None

4.9.14 BLUETOOTH_FindFirstServiceEx

Description

This function performs an SDP service discovery on the specified device, return the first service from the resulting list.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindFirstServiceEx(BTP_Service_Find *ServiceFind,  
BTP_Service_Info_Ex_t *ServiceInfo, const BTP_Service_Query_t *ServiceQuery);
```

Parameters

ServiceFind

Handle to the list of remote services, needed for subsequent calls to BLUETOOTH_FindNextServiceEx and BLUETOOTH_FindServiceClose.

ServiceInfoEx

Information defining the first remote service in the list. This currently supports SPP and HID services.

ServiceQuery

Provides parameters for the SDP query. This currently supports SPP and HID searching by specifying their UUIDs.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_NO_MORE, BTP_ERROR_INVALID_HANDLE

Remarks

None

See Also

BLUETOOTH_FindServiceClose

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 FindFirstServiceEx(ref BT_Service_Find ServiceFind, ref BT_Service_Info_Ex_t ServiceInfo, ref BT_Service_Query_t ServiceQuery)

Example

```
hResult = BLUETOOTH_FindFirstServiceEx(&serviceFind, &serviceInfo, &serviceQuery);
```

4.9.15 BLUETOOTH_FindLocalDevice

Description

This function will return information about the local device. This will include the Bluetooth address, the friendly name, and the class of device.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindLocalDevice(BTP_Find_Local_Device_From_BTE_t  
*DeviceInfo);
```

Parameters

DeviceInfo

Will receive information defining the local device.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR_INVALID_PARAMETER, BTP_ERROR_MSG_SEND

Remarks

None

See Also

None

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 FindLocalDevice(ref BT_Find_Local_Device_From_BTE_t DeviceInfo)

Example

```
hResult = BLUETOOTH_FindLocalDevice(&device);
```

4.9.16 BLUETOOTH_FindNextConnection

Description

This function retrieves the next connection from the list originally created by a call to BLUETOOTH_FindFirstConnection.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindNextConnection(BTP_Connection_Find ConnectionFind,  
BTP_Connection_Info_t *ConnectionInfo);
```

Parameters

ConnectionFind

Handle to the list connections, originally returned by a call to BLUETOOTH_FindFirstConnection.

ConnectionInfo

Information defining a connection from the list.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_NO_MORE, BTP_ERROR_INVALID_HANDLE

Remarks

None

See Also

BLUETOOTH_FindConnectionClose

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 FindNextConnection(BT_Connection_Find ConnectionFind, ref BT_Connection_Info_t ConnectionInfo)

Example

```
hResult = BLUETOOTH_FindNextConnection(connectFind, &connectionInfo);
```

4.9.17 BLUETOOTH_FindNextConnectionEx

Description

This function retrieves the next connection from the list originally created by a call to BLUETOOTH_FindFirstConnection.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindNextConnectionEx(BT_Connection_Find ConnectionFind, BTP_Connection_Info_Ex_t *ConnectionInfo);
```

Parameters

ConnectionFind

Handle to the list connections, originally returned by a call to BLUETOOTH_FindFirstConnection.

ConnectionInfoEx

Information defining a connection from the list. Supports SPP and HID.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_NO_MORE, BTP_ERROR_INVALID_HANDLE

Remarks

None

See Also

BLUETOOTH_FindConnectionClose

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 FindNextConnectionEx(ref BT_Connection_Find ConnectionFind, ref BT_Connection_Info_Ex_t ConnectionInfo)

Example

None

4.9.18 BLUETOOTH_FindNextDevice

Description

This function returns the next device in the list of discovered devices created by a previous call to BLUETOOTH_FindFirstDevice.

Syntax

BLUETOOTH_API HRESULT BLUETOOTH_FindNextDevice(BTP_Device_Find DeviceFind, BTP_Device_Info_t *DeviceInfo);

Parameters

DeviceFind

Handle to the list of discovered devices, originally returned by a call to BLUETOOTH_FindFirstDevice.

DeviceInfo

Information defining a remote device.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_NO_MORE, BTP_ERROR_INVALID_PARAMETER, BTP_ERROR_INVALID_HANDLE

Remarks

None

See Also

BLUETOOTH_FindDeviceClose

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 FindNextDevice(BT_Device_Find DeviceFind, ref BT_Device_Info_t DeviceInfo)

Example

hResult = BLUETOOTH_FindNextDevice(deviceFind, &deviceInfo);

4.9.19 BLUETOOTH_FindNextService

Description

This function returns the next service in the list of services created by a previous call to BLUETOOTH_FindFirstService.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindNextService(BTP_Service_Find ServiceFind,  
BTP_Service_Info_t *ServiceInfo);
```

Parameters

ServiceFind

Handle to the list of remote services, originally returned by a call to BLUETOOTH_FindFirstService.

ServiceInfo

Information defining a remote service from the list.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_NO_MORE, BTP_ERROR_INVALID_HANDLE

Remarks

None

See Also

BLUETOOTH_FindServiceClose

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 FindNextService(BT_Service_Find ServiceFind, ref BT_Service_Info_t ServiceInfo)

Example

None

4.9.20 BLUETOOTH_FindNextServiceEx

Description

This function returns the next service in the list of services created by a previous call to BLUETOOTH_FindFirstService.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindNextService(BTP_Service_Find ServiceFind,  
BTP_Service_Info_t *ServiceInfo);
```

Parameters

ServiceFind

Handle to the list of remote services, originally returned by a call to BLUETOOTH_FindFirstService.

ServiceInfoEx

Information defining the first remote service in the list. This currently supports SPP and HID services.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_NO_MORE, BTP_ERROR_INVALID_HANDLE

Remarks

None

See Also

BLUETOOTH_FindServiceClose

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 FindNextServiceEx(BT_Service_Find ServiceFind, ref BT_Service_Info_Ex_t ServiceInfo)

Example

```
hResult = BLUETOOTH_FindNextServiceEx(serviceFind, &serviceInfo);
```

4.9.21 BLUETOOTH_FindServiceClose

Description

This function deletes the remote list of services and performs any additional cleanup.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_FindServiceClose(BTP_Service_Find ServiceFind);
```

Parameters

ServiceFind

Handle to the list of remote services, originally returned by a call to BLUETOOTH_FindFirstService or BLUETOOTH_FindFirstServiceEx.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_INVALID_HANDLE

Remarks

None

See Also

BLUETOOTH_FindFirstService, BLUETOOTH_FindFirstServiceEx, BLUETOOTH_FindNextService, BLUETOOTH_FindNextServiceEx

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 FindServiceClose(BT_Service_Find ServiceFind)

Example

```
BLUETOOTH_FindServiceClose(serviceFind);
```

4.9.22 BLUETOOTH_GetBluetoothState

Description

This function gets the current Bluetooth device state (either BTP_DEVICE_STATE_ON or BTP_DEVICE_STATE_OFF).

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_GetBluetoothState(DWord_t *BluetoothState);
```

Parameters

BluetoothState

Specifies the current Bluetooth state if the function returns BTP_ERROR_SUCCESS

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_MSG_SEND

Remarks

None

See Also

BLUETOOTH_SetBluetoothState

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 GetBluetoothState(ref UInt32 BluetoothState)

Example

None

4.9.23 BLUETOOTH_GetSCOConnectionState

Description

This function is used to get the SCO connection state for any open Hands-Free connection (either BTP_SCO_STATE_CONNECTED or BTP_SCO_STATE_DISCONNECTED). If SCO is connected, then the audio is being transferred to the Hands-Free device. If SCO is disconnected, then the audio is being played on the local device.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_GetSCOConnectionState(BD_ADDR_t *BD_ADDR, DWord_t *ConnectionState);
```

Parameters

BD_ADDR

The Bluetooth address that BTE Explorer has a Hands-Free connection with.

ConnectionState

Specifies the current SCO connection state if the function returns BTP_ERROR_SUCCESS.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_INVALID_PARAMETER, BTP_ERROR_MSG_SEND

Remarks

None

See Also

BLUETOOTH_SetSCOConnectionState

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 GetSCOConnectionState(ref BD_ADDR_t BD_ADDR, ref UInt32 ConnectionState)

Example

None

4.9.24 BLUETOOTH_GetSecurityMode

Description

For Bluetooth 2.0 (or older) devices, this function is used to retrieve the current security mode. The security mode identifies whether Bluetooth 2.0-style "Mode 3 Security" should be used for authentication and encryption.

The current security mode will be one of the following: BTP_SECURITYMODE_NONE, BTP_SECURITYMODE_AUTHENTICATE, BTP_SECURITYMODE_AUTHENTICATE_AND_ENCRYPT

Syntax

BLUETOOTH_API HRESULT BLUETOOTH_GetSecurityMode(BTP_Security_Mode_Type *SecurityMode);

Parameters

SecurityMode

A pointer to a BTP_Security_Mode_Type that will store the current security mode.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_INVALID_PARAMETER, BTP_ERROR_MSG_SEND

Remarks

None

See Also

BLUETOOTH_SetSecurityMode

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 GetSecurityMode(ref BT_Security_Mode_Type SecurityMode)

Example

None

4.9.25 BLUETOOTH_Open

Description

This function initializes the BTExplorer API module, readying it for use by the application.

Syntax

BLUETOOTH_API bool BLUETOOTH_Open();

Parameters

None

Return Value

None

Remarks

None

See Also

BLUETOOTH_Close

For .Net

Namespace : BlueToothNet Bluetooth

Function : bool Open()

Example

```
if(BLUETOOTH_Open())
{
    m_State.SetWindowText(L"Open Success");
}
else
    m_State.SetWindowText(L"Open Fail");
```

4.9.26 BLUETOOTH_PerformAction

Description

This command is used to request that BTE Explorer take some particular action. This command can be used to launch BTE Explorer in some cases, or trigger specific actions within BTE Explorer in other cases. See the list of supported actions to determine the capabilities of this command.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_PerformAction(DWord_t Action, DWord_t Param1, DWord_t Param2, DWord_t *Return1, DWord_t *Return2);
```

Parameters

Action

Determines the action that BTE Explorer is expected to take as a result of this command.

Param1

The first parameter for the selected action. **Parameters** are defined as needed for each action. Currently parameters are unused and should just be set to zero.

Param2

The second parameter for the selected action. **Parameters** are defined as needed for each action. Currently **Parameters** are unused and should just be set to zero.

Return1

The first return value for the selected action. Currently return values are unused and can be set to NULL.

Return2

The second return value for the selected action. Currently return values are unused and can be set to NULL.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_INVALID_PARAMETER

Remarks

None

See Also

None

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 PerformAction(UInt32 Action, UInt32 Param1, UInt32 Param2, UInt32 Return1, UInt32 Return2)

Example

```
hResult = BLUETOOTH_PerformAction(BTP_ACTION_DELETE_ALL_DEVICES, 0, 0, NULL, NULL);
```

4.9.27 BLUETOOTH_SetAuthenticationCallback

Description

This function registers an authentication callback with BTE Explorer. This function may be called multiple times to associate multiple devices with a callback, but each device may only be associated with a single callback. Calling this function also causes BTE Explorer to attempt to use automatic “JustWorks” pairing if both the local and remote device supports Secure Simple Pairing. “JustWorks” pairing results in a trusted relationship but does not require a PIN code or any user interaction. This callback will be called for “JustWorks” pairing, but the PIN provided will be ignored.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_SetAuthenticationCallback(const BD_ADDR_t *BD_ADDR,  
BTP_Authentication_Callback_t AuthenticationCallback, LPVOID CallbackParameter);
```

Parameters

BD_ADDR

Pointer to the unique identifier of the remote device associated with the callback.

AuthenticationCallback

The callback being registered.

CallbackParameter

User-defined parameter associated with the callback.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_INVALID_PARAMETER

Remarks

None

See Also

None

For .Net

Namespace : BluetoothNet Bluetooth

Function : Int32 SetAuthenticationCallback(ref BD_ADDR_t BD_ADDR, ref
BT_Authentication_Callback_t AuthenticationCallback, ref LPVOID CallbackParameter)

Example

None

4.9.28 BLUETOOTH_SetBluetoothState

Description

This function sets the Bluetooth device to either BTP_DEVICE_STATE_ON or BTP_DEVICE_STATE_OFF. Turning the device on and off will start and stop the BTE Explorer process.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_SetBluetoothState(DWord_t BluetoothState);
```

Parameters

BluetoothState

Specifies the desired Bluetooth state.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_MSG_SEND

Remarks

None

See Also

BLUETOOTH_GetBluetoothState

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 SetBLUETOOTHState(ref UInt32 BLUETOOTHState)

Example

None

4.9.29 BLUETOOTH_SetConnectionCallback

Description

This function registers a connection callback with BTE Explorer. This function may be called multiple times to associate multiple connections with a callback, or to associate multiple callbacks with a connection.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_SetConnectionCallback(BTP_Connection_ID ConnectionID,  
BTP_Connection_Callback_t ConnectionCallback, LPVOID CallbackParameter);
```

Parameters

ConnectionID

Unique identifier for a connection which was previously defined by a call to BLUETOOTH_CreateConnection.

ConnectionCallback

The callback being registered.

CallbackParameter

User-defined parameter associated with the callback.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_INVALID_PARAMETER

Remarks

None

See Also

None

For .Net

Namespace : `BluetoothNet Bluetooth`

Function : `Int32 SetConnectionCallback(ref BT_Connection_ID ConnectionID, ref BT_Connection_Callback_t ConnectionCallback, ref LPVOID CallbackParameter)`

Example

None

4.9.30 BLUETOOTH_SetIncomingPIN

Description

This sets or clears a PIN Code to be used for ANY incoming connections that require authentication. Calling this function also causes BTE Explorer to attempt to use automatic "JustWorks" pairing if both the local and remote device supports Secure Simple Pairing. "JustWorks" pairing results in a trusted relationship but does not require a PIN code or any user interaction. The PIN set is not used for "JustWorks" pairing, but can be used to enable automatic handling if desired.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_SetIncomingPIN(BTP_PIN_t *NewPIN, BTP_PIN_t *OldPIN);
```

Parameters

IncomingPIN

A structure that identifies the new PIN and its length. Set the length of the PIN in this structure to 0 to clear the Incoming PIN.

OldPIN

An output structure that will have the length and value of the Old PIN.

Return Value

`BTP_ERROR_SUCCESS` if successful.

Error codes include the following values: `BTP_ERROR`, `BTP_ERROR_INVALID_PARAMETER`

Remarks

None

See Also

`BLUETOOTH_SetOutgoingPIN`

For .Net

Namespace : `BluetoothNet Bluetooth`

Function : `Int32 SetIncomingPIN(ref BT_PIN_t NewPIN, ref BT_PIN_t OldPIN)`

Example

None

4.9.31 BLUETOOTH_SetOutgoingPIN

Description

This sets or clears a PIN Code to be used for ANY outgoing connections that require authentication. This takes precedence over any registered authentication callback. Calling this function also causes BTE Explorer to attempt to use automatic "JustWorks" pairing if both the local and remote device supports Secure Simple Pairing. "JustWorks" pairing results in a trusted relationship but does not require a PIN code or any user interaction. The PIN set is not used for "JustWorks" pairing, but can be used to enable automatic handling if desired.

Syntax

```
BLUETOOTH_API HRESULT BLUETOOTH_SetOutgoingPIN(BTP_PIN_t *NewPIN, BTP_PIN_t *OldPIN);
```

Parameters

OutgoingPIN

A structure that identifies the new PIN and its length. Set the length of the PIN in this structure to 0 to clear the Outgoing PIN.

OldPIN

An output structure that will have the length and value of the Old PIN.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_INVALID_PARAMETER

Remarks

None

See Also

BLUETOOTH_SetIncomingPIN

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 SetOutgoingPIN(ref BT_PIN_t NewPIN, ref BT_PIN_t OldPIN)

Example

None

4.8.32 BLUETOOTH_SetSCOConnectionState

BLUETOOTH_SetSCOConnectionState

Description

This function is used to set the SCO connection state for any open Hands-Free connection. Turning the SCO connection on and off will transfer audio to the Hands-Free device and return audio to the local device. The possible SCO connection states are BTP_SCO_STATE_CONNECTED and BTP_SCO_STATE_DISCONNECTED.

Syntax

BLUETOOTH_API HRESULT BLUETOOTH_SetSCOConnectionState(BD_ADDR_t *BD_ADDR, DWord_t ConnectionState);

Parameters

BD_ADDR

The Bluetooth address that BTE Explorer has a Hands-Free connection with.

ConnectionState

Specifies the desired SCO connection state.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_INVALID_PARAMETER, BTP_ERROR_MSG_SEND

Remarks

None

See Also

BLUETOOTH_GetSCOConnectionState

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 SetSCOConnectionState(ref BD_ADDR_t BD_ADDR, ref UInt32 ConnectionState)

Example

None

4.9.32 BLUETOOTH_SetSecurityMode

Description

For Bluetooth 2.0 (or older) devices, this function is used to set the security mode for subsequent connections. The security mode dictates whether Bluetooth 2.0-style "Mode 3 Security" should be used for authentication and encryption.

One of three possible modes must be selected: None, Authenticate, or Authenticate and Encrypt.

Syntax

BLUETOOTH_API HRESULT BLUETOOTH_SetSecurityMode(BTP_Security_Mode_Type SecurityMode);

Parameters

SecurityMode

Specifies the desired security mode.

Return Value

BTP_ERROR_SUCCESS if successful.

Error codes include the following values: BTP_ERROR, BTP_ERROR_INVALID_PARAMETER, BTP_ERROR_MSG_SEND

Remarks

None

See Also

BLUETOOTH_GetSecurityMode

For .Net

Namespace : BlueToothNet Bluetooth

Function : Int32 SetSecurityMode(ref BT_Security_Mode_Type SecurityMode)

Example

None

4.10 SYSTEM

Status Return value

Please refer to the below table for the status value definition.

Status Value Definition	Code	Meaning
TRUE	1	Success
FALSE	0	General Error

Enum

```
typedef enum {  
    DEVICE_M3SKY = 0,  
    DEVICE_M3SKYSAM,  
    DEVICE_M3ORANGE,  
    DEVICE_M3SMARTCE,  
    DEVICE_M3SMARTWM,  
    DEVICE_M3T,  
    DEVICE_M3ORANGEPLUS,  
    DEVICE_POS,  
    DEVICE_MM3,  
    DEVICE_M3ORANGEPLUS_CE,  
    DEVICE_M3BLACK,  
    DEVICE_M3UL10_CE,  
    DEVICE_M3BLACK_CE  
} DEVICE_INFO;  
  
typedef enum{  
    BATTERY_POWER,  
    EXTERNAL_POWER  
}POWERTYPE;
```

Structruem

```
typedef struct _GSensor_PARAMS  
{  
    int GsensorEnable;  
    bool DisplayRotate;  
    bool MotionDisplayOff;  
    bool MotionDisplayWakeup;  
    bool MotionSleepOn;  
    bool MotionSleepWakeup;  
    bool MotionSleepOnPrevent;  
    bool MotionStateCheck;  
    bool DisplayMenuCheck;  
    bool SuspendMenuCheck;  
}GSensor_PARAMS, *PGSenor_PARAMS;
```

Functions for System

Name	Description
SYSTEM_BacklightOn	Backlight On/Off
SYSTEM_Cleanboot	Cleanboot
SYSTEM_GetBacklightlevel	Gets the Backlight Level
SYSTEM_GetBacklightTimeOut	Gets the BacklightTimeOut
SYSTEM_GetBatteryLifePercent	Gets the Battery Life Percent
SYSTEM_GetBatteryState	Gets the Battery State
SYSTEM_GetBluetooth	Gets the Bluetooth
SYSTEM_GetCpuClock	Gets the CPU Clock
SYSTEM_GetDeviceInfo	Gets the Information of Device
SYSTEM_GetGUID	Gets the GUID
SYSTEM_GetGSensorValue	Gets the Gyro Sensor Value
SYSTEM_GetOSVersionInfo	Gets the Information of OS Version
SYSTEM_GetPhoneVolumeLevel	Gets the Phone Volume Level
SYSTEM_GetPowerTimeOut	Gets the PowerTimeOut
SYSTEM_GetSerialNumber	Gets the Serial Number
SYSTEM_GetVersionInfo	Gets the Information of C++ DLL Version
SYSTEM_GetVolumeLevel	Gets the Volume Level
SYSTEM_GetWlan	Gets the Wlan
SYSTEM_KeypadLock	Kaypad Lock/Unlock
SYSTEM_Reboot	Reboot
SYSTEM_SetBacklightlevel	Sets the Backlight Level
SYSTEM_SetBacklightTimeOut	Sets the BacklightTimeOut
SYSTEM_SetBluetooth	Sets the Bluetooth On/Off
SYSTEM_SetCpuClock	Sets the CPU Clock
SYSTEM_SetGSensorValue	Sets the Gyro Sensor Value
SYSTEM_SetPhoneVolumeLevel	Sets the Phone Volume Level
SYSTEM_SetPowerTimeOut	Sets the PowerTimeOut
SYSTEM_SetSleepMode	Sleep On
SYSTEM_SetVolumeLevel	Sets the Volume Level
SYSTEM_SetWlan	Sets the Wlan On/Off

SYSTEM Vibrate	Vibrate On/Off
SYSTEM VolumeMute	Volume Mute

4.10.1 SYSTEM_BacklightOn

Description

Backlight On/Off.

Syntax

```
SYSTEM_API BOOL    SYSTEM_BacklightOn(BOOL bOn);
```

Parameters

bOn

The state is either FALSE=Backlight Off, TRUE= Backlight On.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

This function cannot be used with M3 SMART CE.

See Also

None

For .Net

Namespace : SystemNet.System

Function : bool BacklightOn(bool bOn)

Example

```
SYSTEM_BacklightOn(TRUE);
```

```
SYSTEM_BacklightOn(FALSE);
```

4.10.2 SYSTEM_Cleanboot

Description

Cleanboot.

Syntax

```
SYSTEM_API BOOL    SYSTEM_Cleanboot(BOOL bOn);
```

Parameters

bOn

The state is either TRUE=Cleanboot, FALSE=Reboot.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : SystemNet.System

Function : bool Cleanboot(bool bOn)

Example

```
BOOL m_bReboot = true;
SYSTEM_Cleanboot(m_bReboot);
SYSTEM_Reboot();
```

4.10.3 SYSTEM_GetBacklightlevel

Description

Gets the Backlight Level.

Syntax

```
SYSTEM_API DWORD SYSTEM_GetBacklightlevel();
```

Parameters

None

Return Value

Backlightlevel Value.

Remarks

Backlight Level of M3T is divided up into 7 M3T_BACKLIGHTLEVEL Segments. The other devices are divided up into 9 BACKLIGHTLEVEL segments.

See Also

SYSTEM_SetBacklightlevel

For .Net

Namespace : SystemNet.System

Function : int GetBacklightlevel()

Example

```
DWORD dwBacklightlevel_Cur;
dwBacklightlevel_Cur = SYSTEM_GetBacklightlevel();
m_ctlBackLightLevel.SetPos(dwBacklightlevel_Cur);
switch(dwBacklightlevel_Cur)
{
    case BACKLIGHT_100:
        m_strBacklightLevel.Format(L"100%%");
        break;
```

```

case BACKLIGHT_90:
    m_strBacklightLevel.Format(L"90%%");
    break;
case BACKLIGHT_80:
    m_strBacklightLevel.Format(L"80%%");
    break;
case BACKLIGHT_70:
    m_strBacklightLevel.Format(L"70%%");
    break;
case BACKLIGHT_60:
    m_strBacklightLevel.Format(L"60%%");
    break;
case BACKLIGHT_40:
    m_strBacklightLevel.Format(L"40%%");
    break;
case BACKLIGHT_20:
    m_strBacklightLevel.Format(L"20%%");
    break;
case BACKLIGHT_10:
    m_strBacklightLevel.Format(L"10%%");
    break;
case BACKLIGHT_05:
    m_strBacklightLevel.Format(L"5%%");
    break;
default:
    m_strBacklightLevel.Format(L"60%%");
    break;
}

```

4.10.4 SYSTEM_GetBacklightTimeOut

Description

Gets the BacklightTimeOut.

Syntax

```
SYSTEM_API DWORD SYSTEM_GetBackLightTimeOut(POWERTYPE type);
```

Parameters

type

The state is either 0=Battery Status, 1= AC Status.

Return Value

BacklightTimeOut Value.

Remarks

None

See Also

SYSTEM_SetBackLightTimeOut

For .Net

Namespace : SystemNet.System

Function : bool GetBacklightTimeOut(int PowerType)

Example

```
typedef enum{  
    BATTERY_POWER,  
    EXTERNAL_POWER  
}POWERTYPE;  
  
DWORD dwBacklightTimeout_Cur;  
  
dwBacklightTimeout_Cur = SYSTEM_GetBackLightTimeOut(BATTERY_POWER);  
  
m_ctlBackLightTimeout.SetCurSel(dwBacklightTimeout_Cur);
```

4.10.5 SYSTEM_GetBatteryLifePercent

Description

Gets the Battery Life Percent.

Syntax

```
SYSTEM_API int SYSTEM_GetBatteryLifePercent();
```

Parameters

None

Return Value

Current BatteryLife Value.

Remarks

None

See Also

None

For .Net

Namespace : SystemNet.System

Function : int GetBatteryLifePercent()

Example

```
int bBatteryLife;  
bBatteryLife = SYSTEM_GetBatteryLifePercent();
```

4.10.6 SYSTEM_GetBatteryState

Description

Gets the Battery State.

Syntax

```
SYSTEM_API BOOL    SYSTEM_GetBatteryState();
```

Parameters

None

Return Value

TRUE indicates AC Power. FALSE indicates Battery Power.

Remarks

None

See Also

None

For .Net

Namespace : SystemNet.System

Function : bool GetBatteryState()

Example

```
if(SYSTEM_GetBatteryState() == TRUE)  
    m_strBatteryStatus.Format(L"Power : AC Power");  
else  
    m_strBatteryStatus.Format(L"Power : Battery Power");
```

4.10.7 SYSTEM_GetBluetooth

Description

Gets the Bluetooth.

Syntax

```
SYSTEM_API BOOL    SYSTEM_GetBluetooth();
```


Parameters

None

Return Value

TRUE indicates Bluetooth ON. FALSE indicates Bluetooth OFF.

Remarks

This function can only be used in M3 Mobile Device installing Windows Mobile OS.

See Also

SYSTEM_SetBluetooth

For .Net

Namespace : SystemNet.System

Function : bool GetCODE93(out CODE93_PARAMS pCode93)

Example

```
if(SYSTEM_GetBluetooth() == TRUE)
    m_strBluetooth.Format(L"Bluetooth : ON");
else
    m_strBluetooth.Format(L"Bluetooth : OFF");
```

4.10.8 SYSTEM_GetCpuClock

Description

Gets the CPU Clock.

Syntax

SYSTEM_API int SYSTEM_GetCpuClock(DWORD* getclock);

Parameters

getclock

Gets the Current CPU Clock Value.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

This function can't be used in M3 Smart, MM3 and M3 OX10.

See Also

SYSTEM_SetCpuClock

For .Net

Namespace : SystemNet.System

Function : int GetCpuClock(out IntPtr getclock);

Example

```
nCurClock = SYSTEM_GetCpuClock(&dwCpuClock);  
switch (nCurClock)  
{  
case M3T_208MHz:  
    m_strCpuClock.Format(L"CPU Clock : 208MHz");  
    break;  
case M3T_416MHz:  
    m_strCpuClock.Format(L"CPU Clock : 416MHz");  
    break;  
case M3T_624MHz:  
    m_strCpuClock.Format(L"CPU Clock : 624MHz");  
    break;  
case M3T_806MHz:  
    m_strCpuClock.Format(L"CPU Clock : 806MHz");  
    break;  
default:  
    m_strCpuClock.Format(L"CPU Clock : 208MHz");  
    break;  
}
```

4.10.9 SYSTEM_GetDeviceInfo

Description

Gets the Information of Device

Syntax

```
SYSTEM_API int SYSTEM_GetDeviceInfo();
```

Parameters

None

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : System.Net.System

Function : int GetDeviceInfo();

Example

```
nDeviceInfo = SYSTEM_GetDeviceInfo();
switch (nDeviceInfo)
{
    case DEVICE_M3SKY:
    case DEVICE_M3SKYSAM:
        M3Sky_GetCpu();
        break;
    case DEVICE_M3ORANGE:
        M3Orange_GetCpu();
        break;
    case DEVICE_M3SMARTWM:
        M3SmartWM_GetCpu();
        break;
    case DEVICE_M3T:
        M3T_GetCpu();
        break;
    case DEVICE_M3ORANGEPLUS_WM:
    case DEVICE_MM3:
        break;
    case DEVICE_Pos:
        Pos_GetCpu();
        break;
}
```

4.10.10 SYSTEM_GetGUID

Description

Gets the GUID

Syntax

```
SYSTEM_API bool SYSTEM_GetGUID(TCHAR* szGUID);
```

Parameters

szGUID

Gets the GUID Value.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SYSTEM_GetGUID

For .Net

Namespace : SystemNet.System

Function : String GetGuid();

Example

```
TCHAR strGUID[1024] = {0,};  
SYSTEM_GetGUID(strGUID);  
m_strGuid.Format(L"%s", strGUID);
```

4.10.11 SYSTEM_GetGSensorValue

Description

Gets the Gyro Sensor Value

Syntax

```
SYSTEM_API bool SYSTEM_GetGSenorValue(PGSensor_PARAMS pGSensor_Params);
```

Parameters

pGSensor_Params

Pointer to a GSensor_PARAMS structure to be filled in with the Gyro Sensor common parameters

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

This function can only be used in M3 SMART WM and M3 Black WM

See Also

SYSTEM_SetGSensorValue

For .Net

Namespace : SystemNet.System

Function : Bool GetGSensorValue(out GSensor_PARAMS pGSensor_PARAMS);

Example

```
PGSensor_PARAMS pgSensor = new GSensor_PARAMS();
SYSTEM_GetGSensorValue(pgSensor)
m_nGsensorMode = pgSensor->GsensorEnable;
m_bBacklight_OFF = pgSensor->MotionDisplayOff;
m_bBacklight_ON = pgSensor->MotionDisplayWakeup;
m_bPOWER_OFF = pgSensor->MotionSleepOn;
m_bMotionSleepOnPrevent = pgSensor->MotionSleepOnPrevent;
m_bDisplay_Rotate = pgSensor->DisplayRotate;
delete pgSensor
```

4.10.12 SYSTEM_GetOSVersionInfo

Description

Gets the Information of OS Version

Syntax

```
SYSTEM_API bool SYSTEM_GetOSVersionInfo(TCHAR* strVersionInfo);
```

Parameters

szGUID

Gets the GUID Value.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : SystemNet.System

Function : String GetGuid();

Example

```
TCHAR strGUID[1024] = {0,};
SYSTEM_GetGUID(strGUID);
m_strGuid.Format(L"%s", strGUID);
```

4.10.13 SYSTEM_GetPhoneVolumeLevel

Description

Gets the Phone Volume Level.

Syntax

```
SYSTEM_API void SYSTEM_GetVolumeLevel ();
```

Parameters

None

Return Value

The return value is PhoneVolumeLevel.

Remarks

None

See Also

SYSTEM_SetVolumeLevel

For .Net

Namespace : SystemNet.System

Function : int GetVolumeLevel()

Example

```
DWORD dwVolumeLevel_Cur;
dwVolumeLevel_Cur = SYSTEM_GetVolumeLevel();
switch (dwVolumeLevel_Cur)
{
case 0:
    dwVolumeLevel_Cur = 5;
    break;
case 1:
    dwVolumeLevel_Cur = 4;
    break;
case 2:
    dwVolumeLevel_Cur = 3;
    break;
case 3:
    dwVolumeLevel_Cur = 2;
    break;
case 4:
    dwVolumeLevel_Cur = 1;
    break;
```

case 5:

```
dwVolumeLevel_Cur = 0;
```

```
break;
```

default:

```
dwVolumeLevel_Cur = 3;
```

```
break;
```

```
}
```

4.10.14 SYSTEM_GetPowerTimeOut

Description

Gets the PowerTimeOut.

Syntax

```
SYSTEM_API DWORD SYSTEM_GetPowerTimeOut(POWERTYPE type);
```

Parameters

type

The state is either 0=Battery Status, 1= AC Status.

Return Value

The return value is PowerTimeout.

Remarks

None

See Also

SYSTEM_SetPowerTimeOut

For .Net

Namespace : SystemNet.System

Function : int GetPowerTimeOut(int PowerType)

Example

```
DWORD dwPowerTimeout_Cur;
```

```
dwPowerTimeout_Cur = SYSTEM_GetPowerTimeOut(BATTERY_POWER);
```

```
m_ctlPowerTimeout.SetCurSel(dwPowerTimeout_Cur);
```

4.10.15 SYSTEM_GetSerialNumber

Description

Gets the Serial Number.

Syntax

```
SYSTEM_API BOOL SYSTEM_GetSerialNumber(TCHAR* strSerial);
```

Parameters

strSerial

Saving SerialNumber

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : SystemNet.System

Function : string GetSerialNumber();

Example

```
TCHAR strSerial[1024] = {0,};
```

```
SYSTEM_GetSerialNumber(strSerial);
```

```
m_strSerial.Format(L"Serial Number : %s", strSerial);
```

4.10.16 SYSTEM_GetVersionInfo

Description

Gets the Information of C++ dll Version

Syntax

```
SYSTEM_API bool SYSTEM_GetVersionInfo(TCHAR* pszVersion);
```

Parameters

pszVersion

Gets the Information of C++ dll Version

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : SystemNet.System

Function : int GetVersionInfo ()

Example

None

4.10.17 SYSTEM_GetVolumeLevel

Description

Gets the Volume Level.

Syntax

```
SYSTEM_API DWORD SYSTEM_GetVolumeLevel();
```

Parameters

None

Return Value

The return value is VolumeLevel.

Remarks

None

See Also

SYSTEM_SetVolumeLevel

For .Net

Namespace : SystemNet.System

Function : int GetVolumeLevel()

Example

```
DWORD dwVolumeLevel_Cur;
dwVolumeLevel_Cur = SYSTEM_GetVolumeLevel();
switch (dwVolumeLevel_Cur)
{
case 0:
    dwVolumeLevel_Cur = 5;
    break;
case 1:
    dwVolumeLevel_Cur = 4;
    break;
case 2:
    dwVolumeLevel_Cur = 3;
    break;
```

```

case 3:
    dwVolumeLevel_Cur = 2;
    break;
case 4:
    dwVolumeLevel_Cur = 1;
    break;
case 5:
    dwVolumeLevel_Cur = 0;
    break;
default:
    dwVolumeLevel_Cur = 3;
    break;
}

```

4.10.18 SYSTEM_GetWlan

Description

Gets the Wlan.

Syntax

```
SYSTEM_API BOOL  SYSTEM_GetWlan();
```

Parameters

None

Return Value

The return value is Wlan Status.

Remarks

None

See Also

SYSTEM_SetWlan

For .Net

Namespace : SystemNet.System

Function : bool GetWlan()

Example

```

if(SYSTEM_GetWlan() == TRUE)
    m_strWlan.Format(L"WLAN : ON");
else

```

```
m_strWlan.Format(L"WLAN : OFF");
```

4.10.19 SYSTEM_KeypadLock

Description

Keypad Lock/Unlock.

Syntax

```
SYSTEM_API BOOL    SYSTEM_KeypadLock(BOOL bOn);
```

Parameters

bOn

The state is either TRUE=Lock, FALSE=Unlock.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : SystemNet.System

Function : bool KeypadLock(bool bOn)

Example

```
SYSTEM_KeypadLock(TRUE);
```

```
SYSTEM_KeypadLock(FALSE);
```

4.10.20 SYSTEM_Reboot

Description

Reboot.

Syntax

```
SYSTEM_API void SYSTEM_Reboot();
```

Parameters

None

Return Value

None

Remarks

None

See Also

None

For .Net

Namespace : SystemNet.System

Function : void Reboot()

Example

```
SYSTEM_Reboot();
```

4.10.21 SYSTEM_SetBacklightlevel

Description

Sets the Backlight Level.

Syntax

```
SYSTEM_API void SYSTEM_SetBacklightlevel(DWORD m_dwBright);
```

Parameters

m_dwBright

Sets the Backlight Level.

Return Value

None

Remarks

Backlight Level of M3T is divided up into 7 M3T_BACKLIGHTLEVEL Segments. The other devices are divided up into 9 BACKLIGHTLEVEL segments.

See Also

SYSTEM_GetBacklightlevel

For .Net

Namespace : SystemNet.System

Function : void SetBacklightlevel(int m_nBright)

Example

```
DWORD dwPos;
```

```
dwPos = m_ctlBackLightLevel.GetPos();
```

```
SYSTEM_SetBacklightlevel(dwPos);
```

4.10.22 SYSTEM_SetBacklightTimeOut

Description

Sets the BacklightTimeOut.

Syntax

SYSTEM_API BOOL SYSTEM_SetBackLightTimeOut(POWERTYPE type, BOOL bCheck, int nTimeOut);

Parameters

type

The state is either 0=Battery Status, 1= AC Status.

bCheck

The state is either FALSE=Not Set Time, TRUE= Set Time.

nTimeOut

Sets the Timeout Value.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SYSTEM_GetBackLightTimeOut

For .Net

Namespace : SystemNet.System

Function : bool SetBackLightTimeOut(int PowerType, bool bCheck, int nTimeOut)

Example

```
DWORD dwBright;
```

```
SYSTEM_SetBackLightTimeOut(BATTERY_POWER, FALSE, dwBright);
```

```
SYSTEM_SetBackLightTimeOut(BATTERY_POWER, TRUE, dwBright);
```

```
SYSTEM_SetBackLightTimeOut(EXTERNAL_POWER, FALSE, dwACBright);
```

```
SYSTEM_SetBackLightTimeOut(EXTERNAL_POWER, TRUE, dwACBright);
```

4.10.23 SYSTEM_SetBluetooth

Description

Sets the Bluetooth On/Off.

Syntax

SYSTEM_API BOOL SYSTEM_SetBluetooth(BOOL bOn);

Parameters

bOn

The state is either 0=Bluetooth Off, 1= Bluetooth On.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

This function can only be used in M3 Mobile Device installing Windows Mobile OS.

See Also

SYSTEM_GetBluetooth

For .Net

Namespace : SystemNet.System

Function : bool SetBluetooth(bool bOn)

Example

```
SYSTEM_SetBluetooth(TRUE);
```

```
SYSTEM_SetBluetooth(FALSE);
```

4.10.24 SYSTEM_SetCpuClock

Description

Sets the CPU Clock.

Syntax

```
SYSTEM_API BOOL    SYSTEM_SetCpuClock(int cpu);
```

Parameters

cpu

Sets the CPU Clock.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

This function can't be used in M3 Smart, MM3, M3 OX10.

See Also

SYSTEM_GetCpuClock

For .Net

Namespace : SystemNet.System

Function : bool SetCpuClock(int cpu)

Example

```
SYSTEM_SetCpuClock(M3T_208MHz);
```

```
SYSTEM_SetCpuClock(M3T_416MHz);
```

```
SYSTEM_SetCpuClock(M3T_624MHz);
```

4.10.25 SYSTEM_SetGSensorValue

Description

Sets the Gyro Sensor Value

Syntax

```
SYSTEM_API BOOL    SYSTEM_SetGSensorValue(PGSensor_PARAMS pGSensor_PARAMS);
```

Parameters

pGSensor_PARAMS

Pointer to a GSensor_PARAMS structure holding the Gyro Sensor common parameters

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

This function can only be used with M3 SMART WM.

See Also

SYSTEM_GetGSensorValue

For .Net

Namespace : SystemNet.System

Function : bool SetGSensorValue(ref m_GSensor_Params)

Example

```
PGSensor_PARAMS pgSensor = new GSensor_PARAMS();
pgSensor->GsensorEnable = m_nGsensorMode;
pgSensor->MotionDisplayOff = m_bBacklight_OFF;
pgSensor->MotionDisplayWakeup = m_bBacklight_ON;
pgSensor->MotionSleepOn = m_bPOWER_OFF;
pgSensor->MotionSleepOnPrevent = m_bMotionSleepOnPrevent;
pgSensor->DisplayRotate = m_bDisplay_Rotate;
SYSTEM_SetGSensorValue(pgSensor);
```

4.10.26 SYSTEM_SetPhoneVolumeLevel

Description

Sets the Phone Volume Level.

Syntax

```
SYSTEM_API void SYSTEM_SetPhoneVolumeLevel(DWORD m_nVolume);
```

Parameters

nVolume

Sets the PhoneVolume Level.

Return Value

None

Remarks

None

See Also

SYSTEM_GetPhoneVolumeLevel

For .Net

Namespace : SystemNet.System

Function : void SetPhoneVolumeLevel(int m_nVolume)

Example

```
SYSTEM_SetPhoneVolumeLevel(5);
```

4.10.27 SYSTEM_SetPowerTimeOut

Description

Sets the PowerTimeOut.

Syntax

```
SYSTEM_API BOOL    SYSTEM_SetPowerTimeOut(POWERTYPE type, int nTimeOut);
```

Parameters

type

The state is either 0=Battery Status, 1= AC Status.

nTimeOut

Sets the Timeout Value.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

Power TimeOut of M3T and M3 POS is divided up into 8 Segments(NONE, 1 Min, 2 Min, 3 Min, 4 Min, 5 Min, 10 Min, 30Min). Power TimeOut of other Dievce is divided up into 5 segments (NONE, 1 Min, 2 Min, 3 Min, 4 Min, 5 Min).

See Also

SYSTEM_GetPowerTimeOut

For .Net

Namespace : SystemNet.System

Function : bool SetPowerTimeOut(int PowerType, int nTimeOut);

Example


```
DWORD dwPower = 0;
```

```
SYSTEM_SetPowerTimeOut(BATTERY_POWER, dwPower);
```

4.10.28 SYSTEM_SetSleepMode

Description

Sleep On.

Syntax

```
SYSTEM_API void SYSTEM_SetSleepMode();
```

Parameters

None

Return Value

None

Remarks

None

See Also

None

For .Net

Namespace : SystemNet.System

Function : void SetSleepMode()

Example

```
SYSTEM_SetSleepMode();
```

4.10.29 SYSTEM_SetVolumeLevel

Description

Sets the Volume Level

Syntax

```
SYSTEM_API void SYSTEM_SetVolumeLevel(DWORD m_nVolume);
```

Parameters

m_nVolume

Sets the Volume Level.

Return Value

None

Remarks

None

See Also

SYSTEM_GetVolumeLevel

For .Net

Namespace : SystemNet.System

Function : void SetVolumeLevel(int m_nVolume)

Example

```
SYSTEM_SetVolumeLevel(1);
```

4.10.30 SYSTEM_SetWlan

Description

Sets the Wlan On/Off.

Syntax

```
SYSTEM_API BOOL SYSTEM_SetWlan(BOOL bOn);
```

Parameters

bOn

The state is either 0=Wlan Off, 1= Wlan On.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

SYSTEM_GetWlan

For .Net

Namespace : SystemNet.System

Function : bool SetWlan(bool bOn)

Example

```
SYSTEM_SetWlan(TRUE);
```

```
SYSTEM_SetWlan(FALSE);
```

4.10.31 SYSTEM_Vibrate

Description

Vibrate On/Off.

Syntax

```
SYSTEM_API BOOL SYSTEM_Vibrate(BOOL bOn);
```

Parameters

bOn

The state is either 0=Vibrate Off, 1= Vibrate On.

Return Value

Nonzero indicates success. Zero indicates failure.

Remarks

None

See Also

None

For .Net

Namespace : SystemNet.System

Function : bool Vibrate(bool bOn)

Example

```
SYSTEM_Vibrate(TRUE);
```

```
SYSTEM_Vibrate(FALSE);
```

4.10.32 SYSTEM_VolumeMute

Description

Volume Mute.

Syntax

```
SYSTEM_API void SYSTEM_VolumeMute();
```

Parameters

None

Return Value

None

Remarks

None

See Also

None

For .Net

Namespace : SystemNet.System

Function : void VolumeMute()

Example

```
SYSTEM_VolumeMute();
```

5 Demo Manual

This chapter is for Demo manuals for WLAN, Bluetooth and System. Please refer to the "Application Manual" for other application's demo manuals.

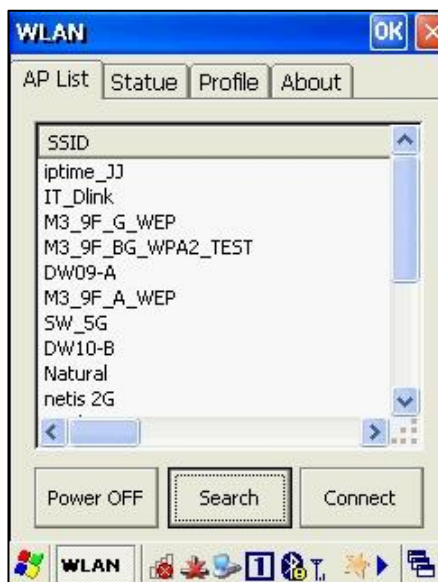
5.1 WLAN

Product	Model	Type	OS	Note
ALL	ALL	Summit	WM6.1/6.5 CE 5.0/6.0	Applicable in Summit module, if 3rd party config is set

Precautions

- WLANTest.exe is usually located at \Flash Disk\WLAN.
- WLANTest.exe can be used on devices with Summit WLAN Module.
To use WlanTray instead of SCU (Summit Utility Client), the Active Profile on SCU must be set to ThirdPartyConfig.

1. First view of the program – AP List View



SSID : Service Set Identifier.

RSSI : Received Signal Strength Indication.

Security

true : Encryption

false : Open.

Power OFF : WLAN tern On/off.

Search : Scan available network.

Connect : Connect to selected network.

2. Connecting View for Encryption Type.



Connect Info : display View current status of WLAN.

Encryption : Access to an AP with stored settings.

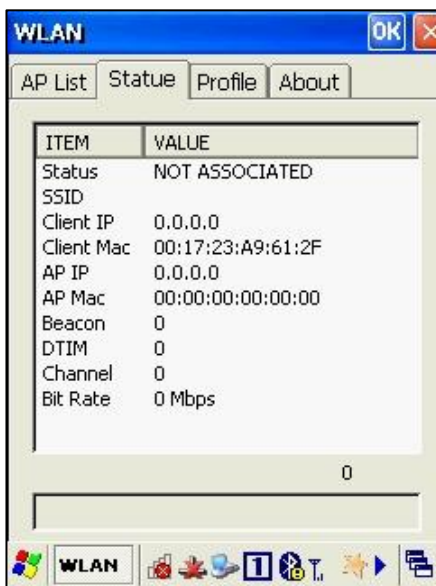
Password : Write password.

Confirm : Confirm password.

Yes : Connect to apply AP info.

No : Cance..

3. State View



Status : Current status of WLAN

SSID : Name of connected SSID

Client IP

Client Mac

AP IP

AP Mac

Beacon : Shows periodic time of broad cast signal(beacon) that received from AP as Ksec.

DTIM : Shows how often the AP includes the DTIM(Delivery Traffic Indication Message) when sending signals. If the DTIM is 3, it means that DTIM is included in every third broadcast signal(beacon).

Channel : Connects to channel that AP sets and mostly channel 9 or 11 is used.

Bit Rate : Number of bits proceeded per 1 second.

TX Power

RSSI : Signal Strength,

4. Profile View



Active Profile : Profile of currently connected WLAN.

Profile List : List of stored profiles. Profile is produced automatically if it is connected at least once.

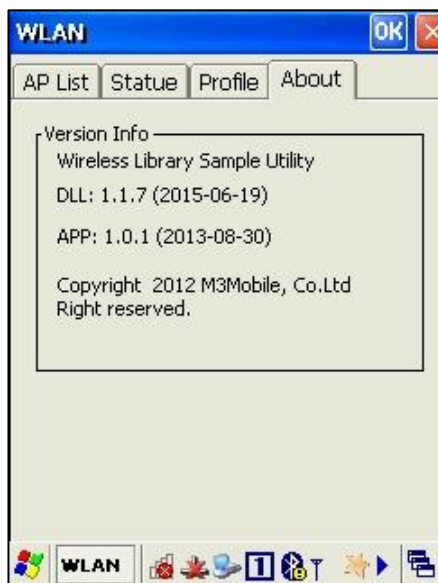
Connect : Connects to selected profile.

Delete : Deletes selected profile

Import : Restores the WLAN connection configuration.

Export : Back up of the WLAN connection configuration.

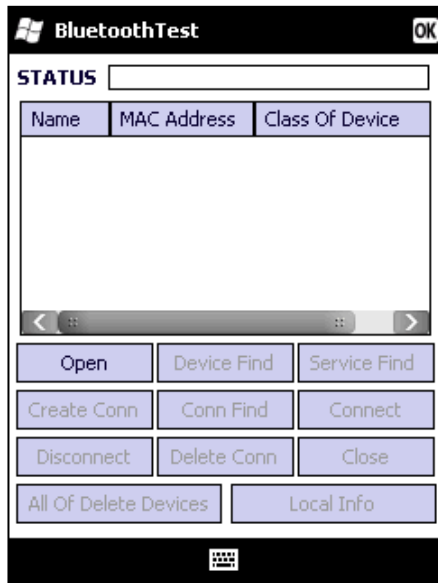
5. About View



Display Version Info

5.2 BLUETOOTH

1. Main Page



STATUS : Status of Bluetooth.

Open : Opens Bluetooth COM Port.

Device Find : Finds Bluetooth-connectable device.

Service Find : Finds service that Bluetooth can be connected.

Create Conn : Makes a Connection to connect.

Conn Find : Finds a Connection.

Connect : Connects Bluetooth devices to each other.

Disconnect : Disconnects the connected device.

Delete Conn : Deletes the Connection.

Close : Closes the Bluetooth COM Port.

All of Delete Devices : Deletes all searched devices.

Local Info : Provides the device information that runs program.



■ This demo program can be applied to upper level from StonestreetOne BTExplorer Version 2.1.1 and Build Version 27460.

■ Performance procedure

1) If there is no Connection...

Open -> Device Find -> Service Find -> Create Connection -> Connection Find -> Connect -> Disconnect -> (Delete Connection) -> Close

2) If there is Connection...

Open -> Connection Find -> Connect -> Disconnect -> (Delete Connection) -> Close

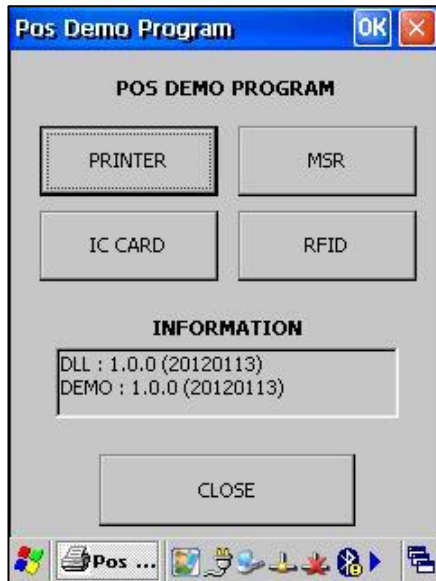
5.3 POS

Product	Model	Type	OS	Note
M3 PS10	M3 PS10	Software	CE 5.0	Program for BT printer in M3 Mobile devices

Precautions

- PosTest.exe is usually located at \Flash Disk\POS.

1. First view of the program



PRINTER : Runs Printer Test Dialog.

MSR : Runs MSR Test Dialog.

IC CARD : Runs IC CARD Test Dialog.

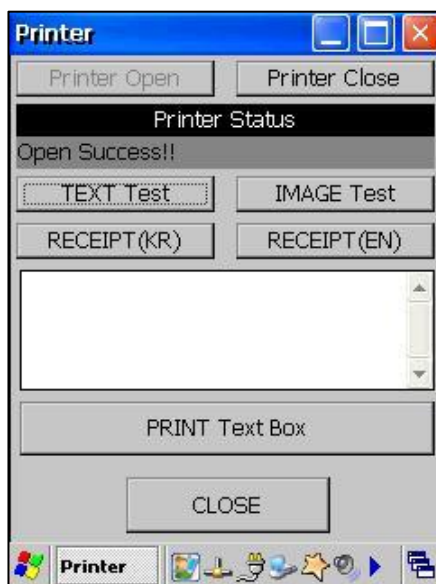
RFID : Runs RFID Test Dialog.

INFORMATION : DLL version info.

Ver : Test program version info

Close : Closes program

2. Printer Dialog.



Printer Open/Close : Printer Module On and Off.

Printer Status : Shows current status of printer.

TEXT Test : Font format can be adjusted but the font cannot.

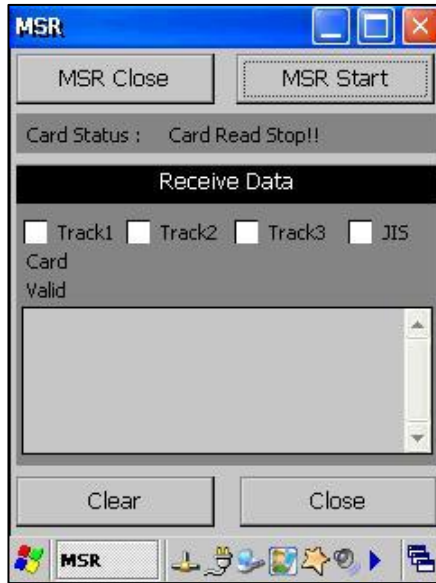
IMAGE Test : Prints text in the text box.

RECEIPT : Prints receipt sample

PRINT Text Box : Prints text in the text box.

CLOSE : Returns to first Dialog.

3. MSR Dialog



MSR Open/Close : MSR module On and Off.

MSR Start/Stop : MSR Reading On and Off.

Card Status : Shows current status of MSR.

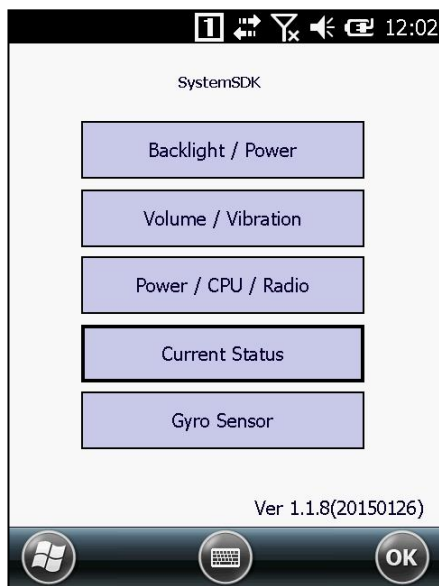
Close : Returns to first Dialog.

Card, Valid : No showing if Track 2 is not read.

5.4 SYSTEM

Product	Model	Type	OS	Note
M3 BK10	M3 BK10		WM 6.5/CE 6.0	Gyro Sensor
M3 MT10	M3 MT10		CE 5.0	
M3 OX10	M3 OX10		WM 6.5/CE 6.0	
M3 PS10	M3 PS10		CE 5.0	
M3 ST10	M3 ST10		WM 6.5/CE 6.0	Gyro Sensor
M3 UL10	M3 UL10		CE 6.0	

1. Main Page



Backlight / Power

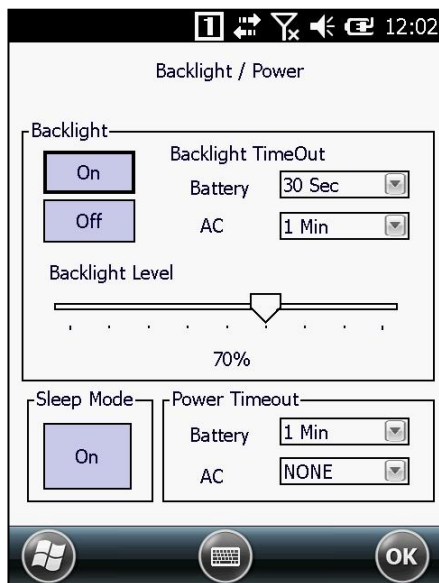
Volume / Vibration

Power / CPU / Radio

Current Status

Gyro Sensor

2. Backlight / Power



Backlight

On : Backlight On

Off : Backlight Off

Backlight TimeOut

Battery : Sets the Battery Backlight Timeout

AC : Sets the AC Backlight Timeout

Backlight Level : Backlight Level Control

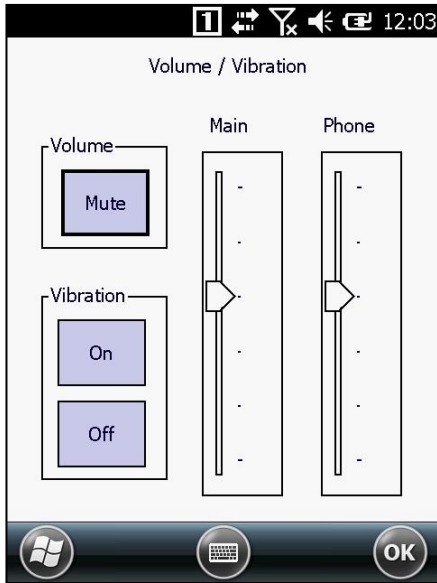
Sleep Mode : Sleep On

Power Timeout

Battery : Sets the Battery Backlight Timeout

AC : Sets the AC Backlight Timeout

3. Volume / Vibration



Volume

Mute : Volume Mute

Vibration

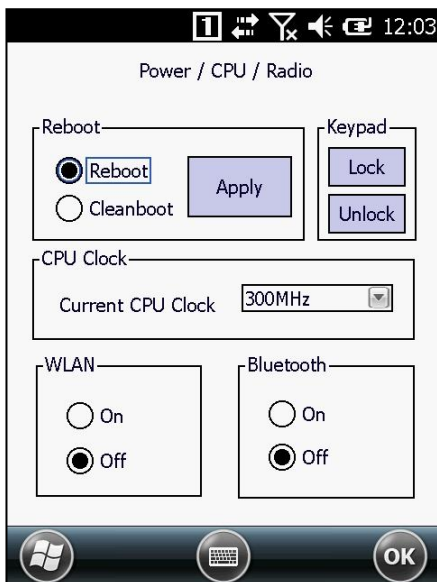
On : Vibration On

Off : Vibration Off

Main Volume : Main Volume Control

Phone Volume : Phone Volume Control

4. Power / CPU / Radio



Reboot

Reboot + Apply : Reboot

Cleanboot + Apply : Cleanboot

Keypad

Lock : Keypad Lock

Unlock : Keypad Unlock

CPU Clock : Sets the Current CPU Clock

WLAN

On : Wlan On

Off : Wlan Off

Bluetooth

On : Bluetooth On

Off : Bluetooth Off

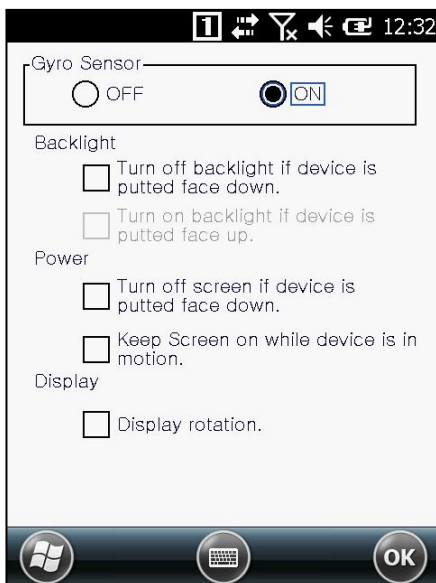
5. Current Status



Status Information

- Serial Number : Gets the Serial Number
- UUID : Gets the UUID
- OS Version : Gets the OS Version
- WLAN : Gets the Wlan Status
- Bluetooth : Gets the Bluetooth Status
- CPU Clock : Gets the Current CPU Clock
- Power : Gets the Power Status(Battery / AC)
- Battery Life : Gets the Current BatteryLife

6. Gyro Sensors



Gyro Sensor

- ON : Gyro Sensor On
- OFF : Gyro Sensor Off

Backlight

- Turn off backlight
- Turn on backlight

Power

- Turn off Screen
- Keep Screen

Display

- Display rotation

Services

If you experience any trouble while using our product, you can visit **M3 Service center** or send enquires to our **online support web page** (<http://itc.m3mobile.net>), we will do our best to solve your trouble as soon as we can.

M3 FAQ document can help you with troubleshooting.

For any enquires about business program, please contact program provider for faster service.

Contact Details

Headquarter

M3 Bldg., 735-45, Yeoksam-Dong, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82 2 574 0037 Fax: +82 2 558 1253

www.m3mobile.net, sales@m3mobile.co.kr

Factory/ Service Center

Chun-ui Techno Park 201-610, 202, Chunui-Dong, WonMi-Gu, Buchoen, Gyeonggi-Do, 420-857, Korea
Tel: +82 32 623 0030, Fax: +82 32 623 0035

Online Support Web page

<http://itc.m3mobile.net>