CipherLab
User Guide

.NET Programming

for 9200 Mobile Computer

Version 1.06

Copyright © 2015 CIPHERLAB CO., LTD.
All rights reserved

The software contains proprietary information of CIPHERLAB CO., LTD.; it is provided under a license agreement containing restrictions on use and disclosure and is also protected by copyright law. Reverse engineering of the software is prohibited.

Due to continued product development this information may change without notice. The information and intellectual property contained herein is confidential between CIPHERLAB and the client and remains the exclusive property of CIPHERLAB CO., LTD. If you find any problems in the documentation, please report them to us in writing. CIPHERLAB does not warrant that this document is error-free.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without the prior written permission of CIPHERLAB CO., LTD.

For product consultancy and technical support, please contact your local sales representative. Also, you may visit our web site for more information.

The CipherLab logo is a registered trademark of CIPHERLAB CO., LTD.

Other product name mentioned in this manual may be trademarks or registered trademarks of their respective companies and are hereby acknowledged.

The editorial use of these names is for identification as well as to the benefit of the owners, with no intention of infringement.

CIPHERLAB CO., LTD.
Website: http://www.cipherlab.com

RELEASE NOTES

Version	Date	Notes
1.06	Jun 15, 2015	▸ Rivise 4.4.1 BSSIDINFO structure adding *public int authentication* ▸ Add 2.7.20 PicklistMode_2D_SE4500 ▸ Add 2.7.21 DisplayMode_2D
1.05	Mar 4, 2015	▸ Revise *timeout* value for ReadBarcodeData in 2.2.3 Barcode Data ▸ Revise *timeout* value for ReadRfidData and WriteRfidData in 2.2.4 RFID Data ▸ Revise *bidirectionalRedundancy* value for UserPreferences in 2.6.1 Preferences
1.04	Mar 30, 2015	▸ Add UserPreferences_Extend in 2.7.1 Preferences for 2D Scan Engine ▸ Add 4.8 User Preferred Settings
1.03	Oct 24, 2014	▸ Revise unit for Wi-Fi Roam Period
1.02	Dec 2, 2013	▸ Add 3.7 Microphone
1.01	Nov 8, 2013	▸ Modify description for DataOutputSettings code length and tag length (does not include prefix or suffix) ▸ Update Reader Configure UI ▸ Add SCU Wi-Fi APIs: 3.7.2 Radio, 3.7.4 Domain, SDK & MAC Information - GetCurrentDomain, GetSDKVersion, 3.7.8 Configuration Profiles, 3.7.9 Edit Configuration Profile, 3.7.10 Summit Config Settings, 3.7.11 Global Config Settings, 3.7.12 Summit Registry Keys, 3.7.13 Configuration File, 3.7.14 WEP Key, 3.7.15 Pre-Shared Key, 3.7.16 LEAP Credentials, 3.7.17 EAP-FAST Credentials, 3.7.18 PEAP-GTC Credentials, 3.7.19 PEAP-MSCHAP Credentials, 3.7.20 EAP-TLS Credentials, 3.7.21 EAP-TTLS Credentials ▸ Add SCU Wi-Fi data structure: 4.4.5 CF10G_STATUS, 4.4.6 CONFIG_FILE_INFO, 4.4.7 SDC_ALL, 4.4.8 SDCConfig, 4.4.9 SDC3rdPartyConfig, 4.4.10 SDCGlobalConfig
1.00	Aug 19, 2013	▸ Initial release

CONTENTS

RELEASE NOTES ..... - 3 -

INTRODUCTION 1

Development Tool 2

CREATE APPLICATIONS 3

1.1. Create Visual C# Applications 4

1.2. Create Visual Basic Applications 11

1.3. Register Decode Message 18

1.3.1. Visual C# 18

READER API 21

2.1. Initialize/Identify Reader 22

2.1.1. Initialization 22

2.1.2. Active Device 23

2.1.3. Reader Type 25

2.2. Obtain Data 26

2.2.1. Data Output Settings 26

2.2.2. RFID Settings 33

2.2.3. Barcode Data 38

2.2.4. RFID Data 43

2.3. Manipulate Status Indication 50

2.3.1. Notification Settings 50

2.3.2. Beeper (Speaker) 51

2.4. Obtain Essential Information 52

2.4.1. Reader DLL Version 52

2.4.2. Decoder Version 53

2.4.3. Error Information 54

2.5. Configure Scan Engine — 1D CCD Scan Engine 55

2.5.1. Preferences 55

2.5.2. Upc_1D_SM1 57

2.5.3. EanJan_1D_SM1 60

2.5.4. Code39_1D_SM1 62

2.5.5. Code93_1D_SM1 64

2.5.6. Interleaved2Of5_1D_SM1 65

2.5.7. Industrial2Of5_1D_SM1 66

2.5.8. Codabar_1D_SM1 67

2.5.9. Msi_1D_SM1 69

2.5.10. GS1_DataBar_1D_SM1 71

2.5.11. Code128_1D_SM1 72

2.6. Configure Scan Engine — 1D Laser Scan Engine 73

2.6.1. Preferences 73

2.6.2. Upc_1D_SE955 77

2.6.3. EanJan_1D_SE955 80

2.6.4. Code39_1D_SE955 82

2.6.5. Code93_1D_SE955 84

2.6.6. Interleaved2Of5_1D_SE955 85
2.6.7. Industrial2Of5_1D_SE955 86
2.6.8. Chinese2Of5_1D_SE955 87
2.6.9. Codabar_1D_SE955 88
2.6.10. Msi_1D_SE955 90
2.6.11. GS1_DataBar_1D_SE955 92
2.6.12. Code128_1D_SE955 93
2.6.13. Code11_1D_SE955 94
2.7. Configure Scan Engine — 2D Scan Engine 95
2.7.1. Preferences 95
2.7.2. Upc_2D_4500 103
2.7.3. EanJan_2D_4500 106
2.7.4. Code39_2D_4500 108
2.7.5. Code93_2D_4500 110
2.7.6. Interleaved2Of5_2D_4500 111
2.7.7. Industrial2Of5_2D_4500 112
2.7.8. Matrix2Of5_2D_4500 113
2.7.9. Chinese2Of5_2D_4500 114
2.7.10. Codabar_2D_4500 115
2.7.11. Msi_2D_4500 116
2.7.12. GS1_DataBar_2D_4500 118
2.7.13. Code128_2D_4500 119
2.7.14. Code11_2D_4500 121
2.7.15. PostalCode_2D_4500 122
2.7.16. Composite_2D_4500 124
2.7.17. Inverse1D_2D_SE4500 125
2.7.18. Korean3Of5_2D_SE4500 126
2.7.19. Symbologies_2D_4500 127
2.7.20. PicklistMode_2D_SE4500 129
2.7.21. DisplayMode_2D 130
2.8. RESET Reader 131
SYSTEM API 133
3.1. System Settings 134
3.1.1. API Version 134
3.1.2. Universally Unique Identifier (UUID) 135
3.1.3. Device Name 136
3.1.4. System Information 137
3.1.5. Start Program 138
3.1.6. Soft Reset 138
3.2. Status Indication 139
3.2.1. LED Light 139
3.2.2. Vibrator 140
3.3. LCD Backlight 141
3.3.1. Backlight Control 141
3.3.2. Backlight Level 143
3.3.3. Reset Backlight to Default 145
3.3.4. Set Backlight to Maximum 146
3.3.5. Set Backlight to Minimum 147
3.4. Keypad Backlight 148

3.5. Touch Panel 149
3.5.1. Touch Panel State 149
3.6. Keypad 151
3.6.1. Sensitivity 151
3.6.2. Keypad Lock 152
3.6.3. Keypad Mode 154
3.6.4. Keypad State 156
3.7. Microphone 158
3.7.1. Device Microphone 158
3.7.2. Headset Microphone 160
3.8. Wireless LAN 162
3.8.1. Wi-Fi Power 162
3.8.2. Radio 163
3.8.3. IP Information 164
3.8.4. Domain, SDK & MAC Information 166
3.8.5. Network List 168
3.8.6. Network Status 171
3.8.7. BSSID 172
3.8.8. Configuration Profiles 173
3.8.9. Edit Configuration Profile 175
3.8.10. Summit Config Settings 180
3.8.11. Global Config Settings 181
3.8.12. Summit Registry Keys 182
3.8.13. Configuration File 183
3.8.14. WEP Key 185
3.8.15. Pre-Shared Key 190
3.8.16. LEAP Credentials 192
3.8.17. EAP-FAST Credentials 194
3.8.18. PEAP-GTC Credentials 196
3.8.19. PEAP-MSCHAP Credentials 199
3.8.20. EAP-TLS Credentials 202
3.8.21. EAP-TTLS Credentials 205
3.9. Bluetooth 208
3.9.1. Start/stop Bluetooth 210
3.9.2. Find Devices 211
3.9.3. Pair Devices 212
3.9.4. Available Services & Device Information 213
3.9.5. Bluetooth Connection & Status 217
3.9.6. FTP Service – View Files 222
3.9.7. FTP Service – File Transfer 224
3.9.8. SPP COM Port 226
3.10. Camera 228
3.10.1. Camera Power 229
3.10.2. Camera Settings 230
3.10.3. Preview 232
3.10.4. Still Capture 236
3.10.5. Torch Mode 239
3.11. GPS 240
3.12. Button Assignment 241

3.13. Signature Capture 247
3.13.1. Initialization/Exit 247
3.13.2. Control of Signature Area 249
3.13.3. Background Color 250
3.13.4. Pen Color and Width 251
3.13.5. Image Saving & Loading 252
3.13.6. Signature DLL Version 254
3.14. Error Information 255
DATA STRUCTURE 257
4.1. System Information Structure 258
4.1.1. SysInfo 258
4.2. Backlight Structure 260
4.2.1. BklCtl 260
4.3. Keypad Structure 262
4.3.1. KeyPadBind 262
4.4. Wi-Fi Structures 263
4.4.1. BssidInfo 263
4.4.2. WlanAdptInfo 264
4.4.3. WlanConnectedSt 265
4.4.4. WlanCtl 266
4.4.5. CF10G_STATUS 268
4.4.6. CONFIG_FILE_INFO 271
4.4.7. SDC_ALL 272
4.4.8. SDCConfig 273
4.4.9. SDC3rdPartyConfig 276
4.4.10. SDCGlobalConfig 278
4.5. Bluetooth Structure 284
4.5.1. FTP_File_Info 284
4.6. Camera Structures 285
4.6.1. PicState 285
4.6.2. Flash 286
4.7. Camera Structures 287
4.7.1. PicState 287
4.8. User Preferred Settings 288
4.8.1. UserPreferencesParameters 288
SCAN ENGINE SETTINGS 289
Symbologies Supported 290
RFID Tags Supported 292
SAMPLE CODE 293
Visual C# 293
Visual Basic 295
INDEX 297

INTRODUCTION

This programming guide contains the necessary info for building applications that control readers, retrieve decoded data, and modulate system resources on CipherLab's 9200 Mobile Computer, a multi-functional device powered by Windows Embedded Handheld 6.5.

.NET Framework makes it easy to create such applications. Simply import the prospective .NET component (.NET DLL Class Library) to your .NET Smart Device application to make your way to build it. Two DLL files, **ReaderDllMobile_NET.dll** and **SystemMobile_Net.dll**, that your application can reference and exploit are provided on the bundled CD-ROM.

We recommend that you read the documents thoroughly before use and keep it at hand for quick reference.

Thank you for choosing CipherLab products!

DEVELOPMENT TOOL

Programmers are assumed to possess Windows programming knowledge. For the information on Windows application programming interface (API), essential online resource is available on Microsoft website –

http://msdn2.microsoft.com/en-us/library/default.aspx

As the mobile computer gets powered by Windows Embedded Handheld 6.5, select one of the following development environment and tools as you see fit:

- Visual Studio 2008
- Visual Studio 2005
- 9200 SDK

Chapter 1

CREATE APPLICATIONS

DLL files **ReaderDllMobile_NET.dll** and **SystemMobile_Net.dll** help your application gain control over the reader(s) and hardware on board the mobile computer. Find them on the bundled CD-ROM.

1. IN THIS CHAPTER

1.1 Create Visual C# Applications..... 4
1.2 Create Visual Basic Applications.....11
1.3 Register Decode Message 18

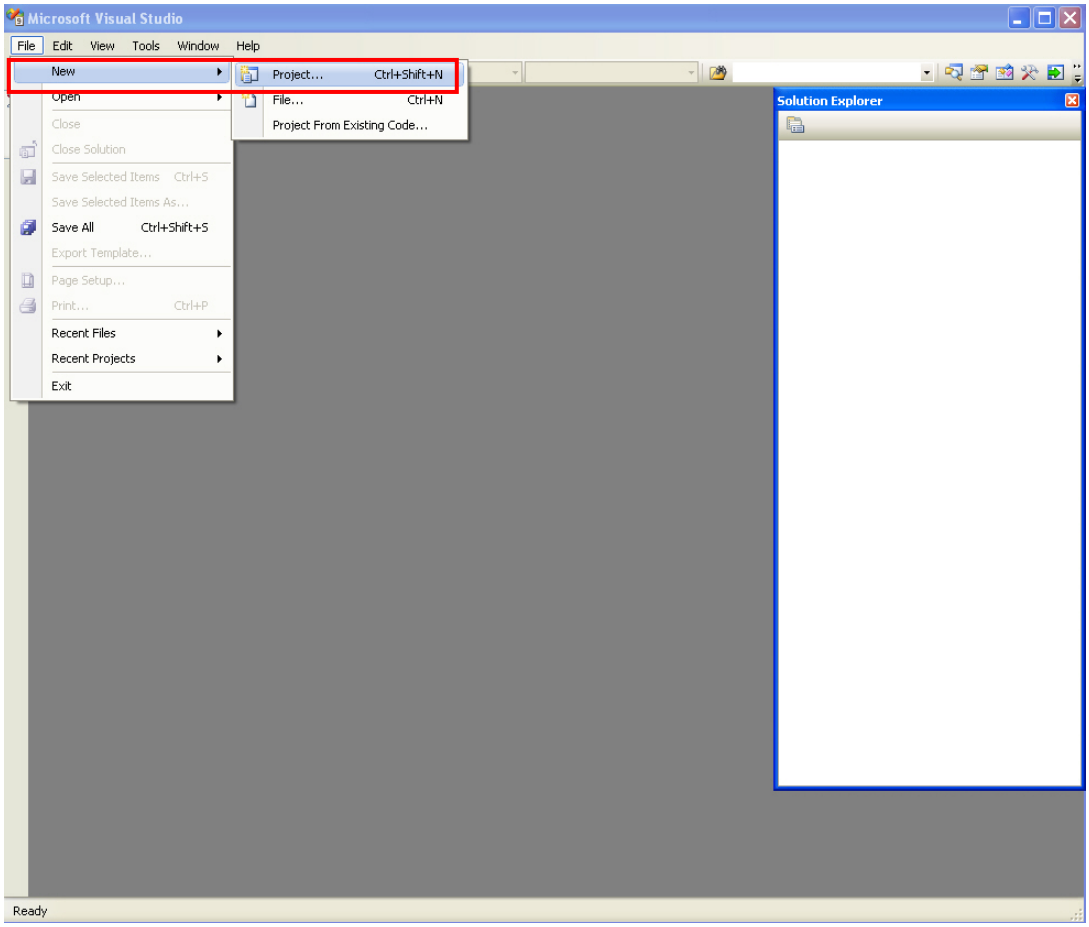
1.1. CREATE VISUAL C# APPLICATIONS

Follow the steps below to create Visual C# applications in Visual Studio 2008. See Appendix II Sample Code.

I) Run Visual Studio on your PC.

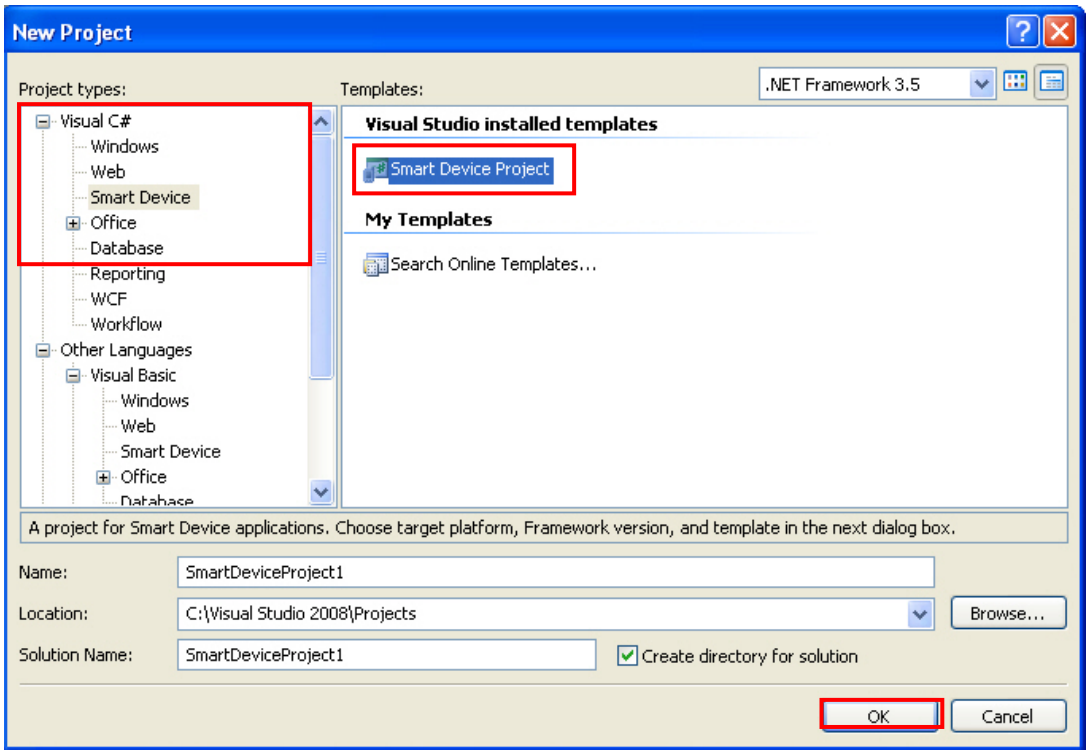
Visual Studio opens.

2) From the menu bar, click **File** | **New** | **Project**.



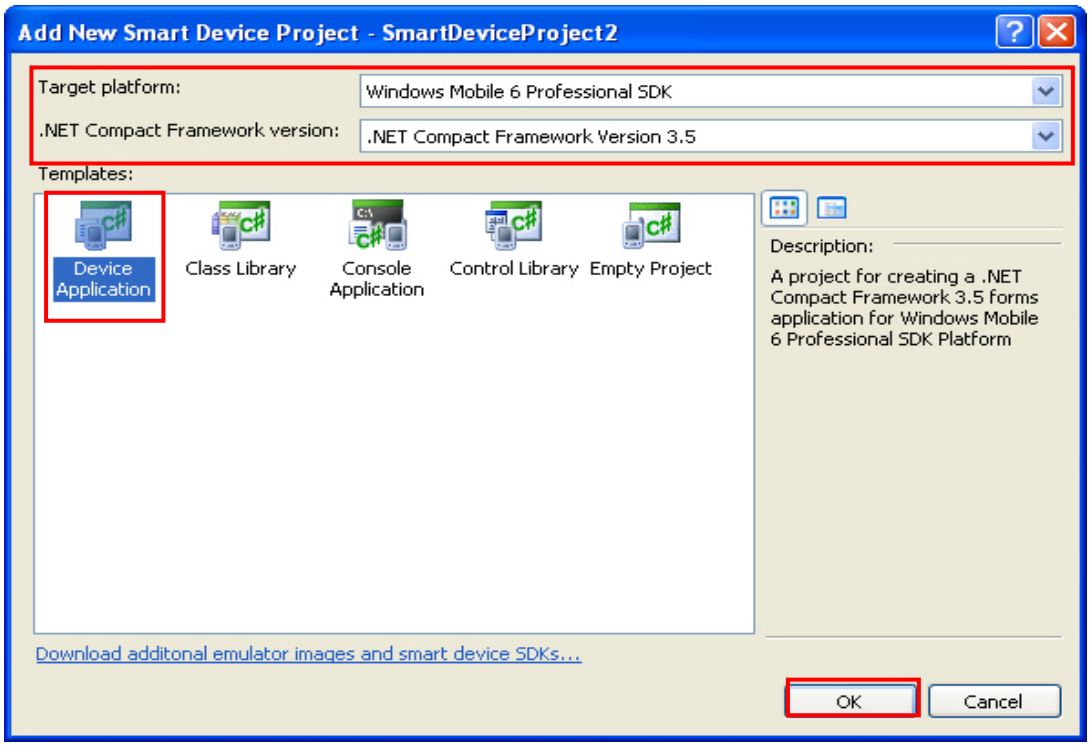
A **New Project** window opens.

3) In the left [Project type] pane, select **Visual C#**|**Smart Device**. In the right [Templates] pane, select **Smart Device Project**. Click **OK** to proceed.



An Add New Smart Device window opens.

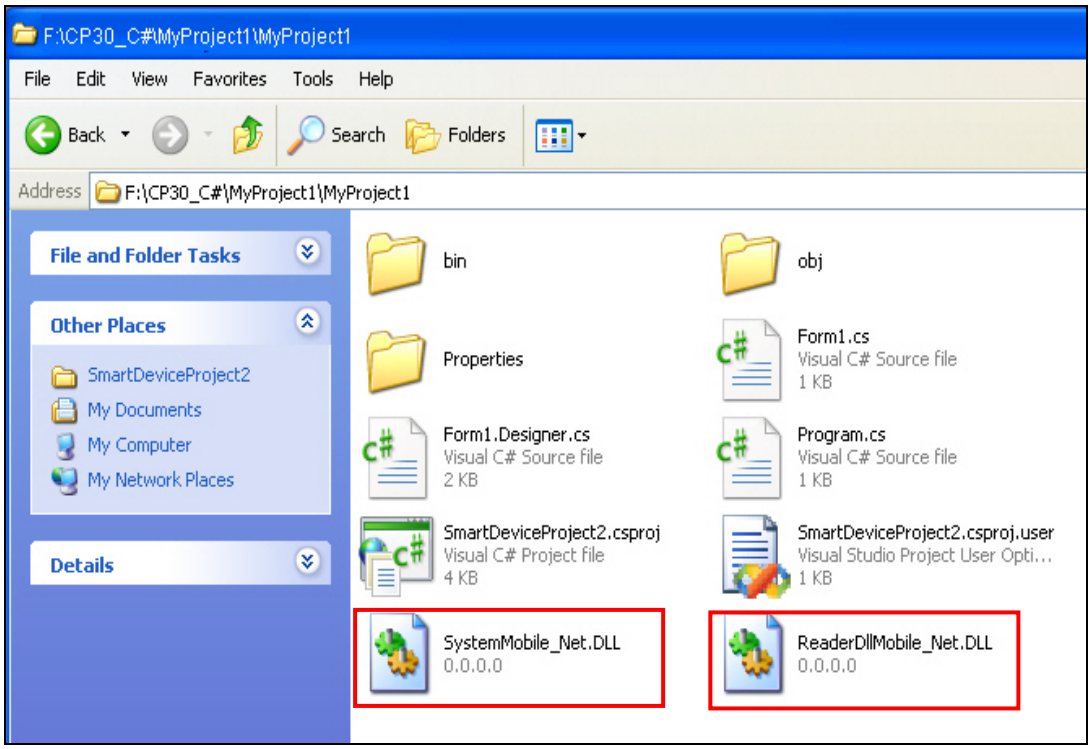
4) Select **Windows Mobile 6 Professional SDK** for target platform and **Device Application** for the template to use. Click **OK**.



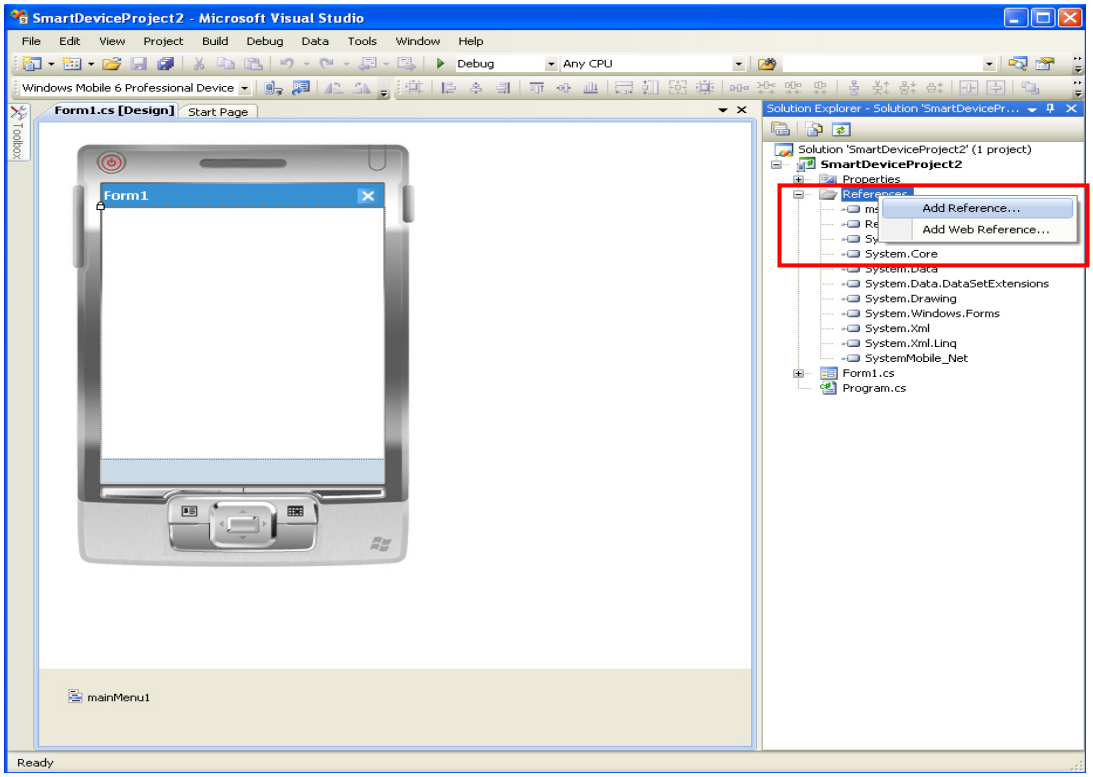
A project folder is created and new project window opens showing a prototype of template.

5) Find the CD-ROM bundled with 9200 mobile computer. Copy the following files to your new project folder:

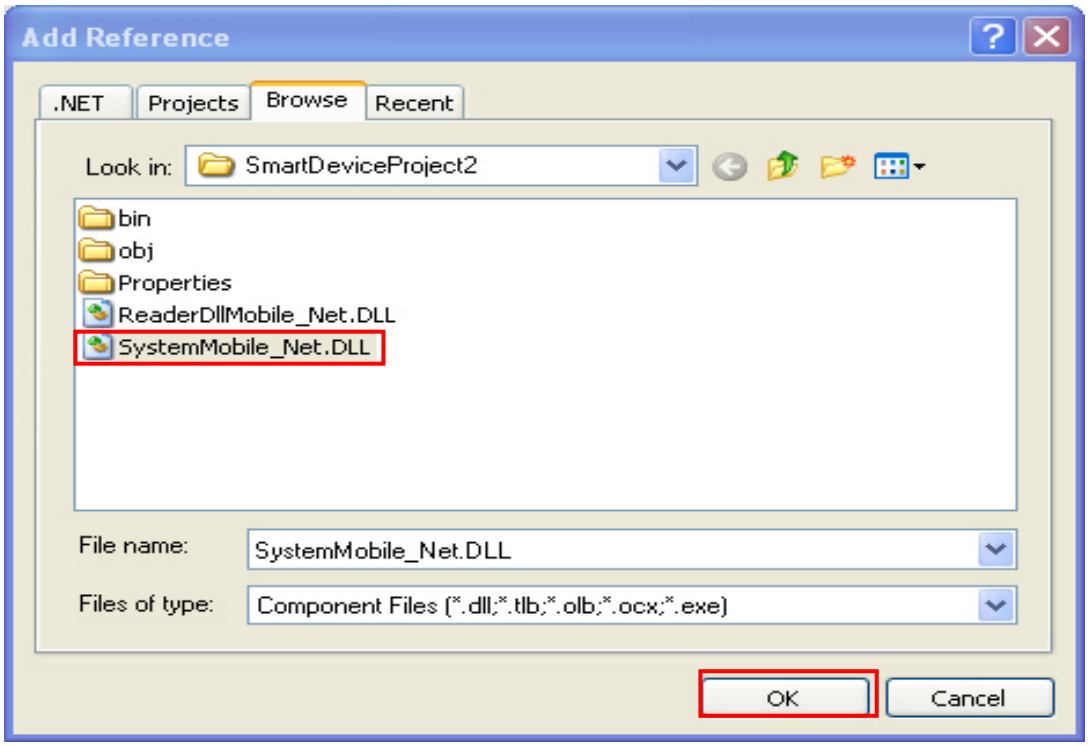
- SystemMobile_Net.dll
- ReaderDllMobile_Net.dll



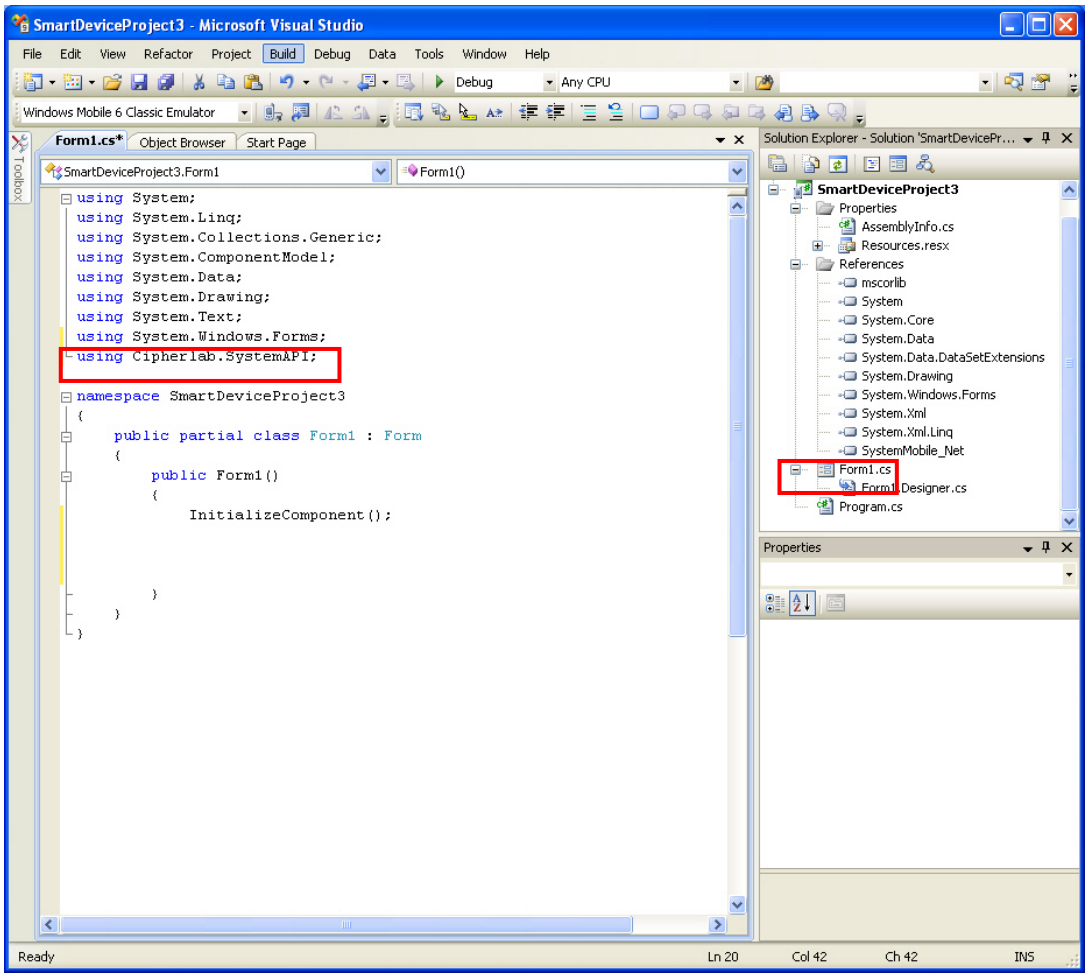
6) Back to Visual Studio. In the right pane, under the name of your new project, right-click **References** and select **Add Reference**.



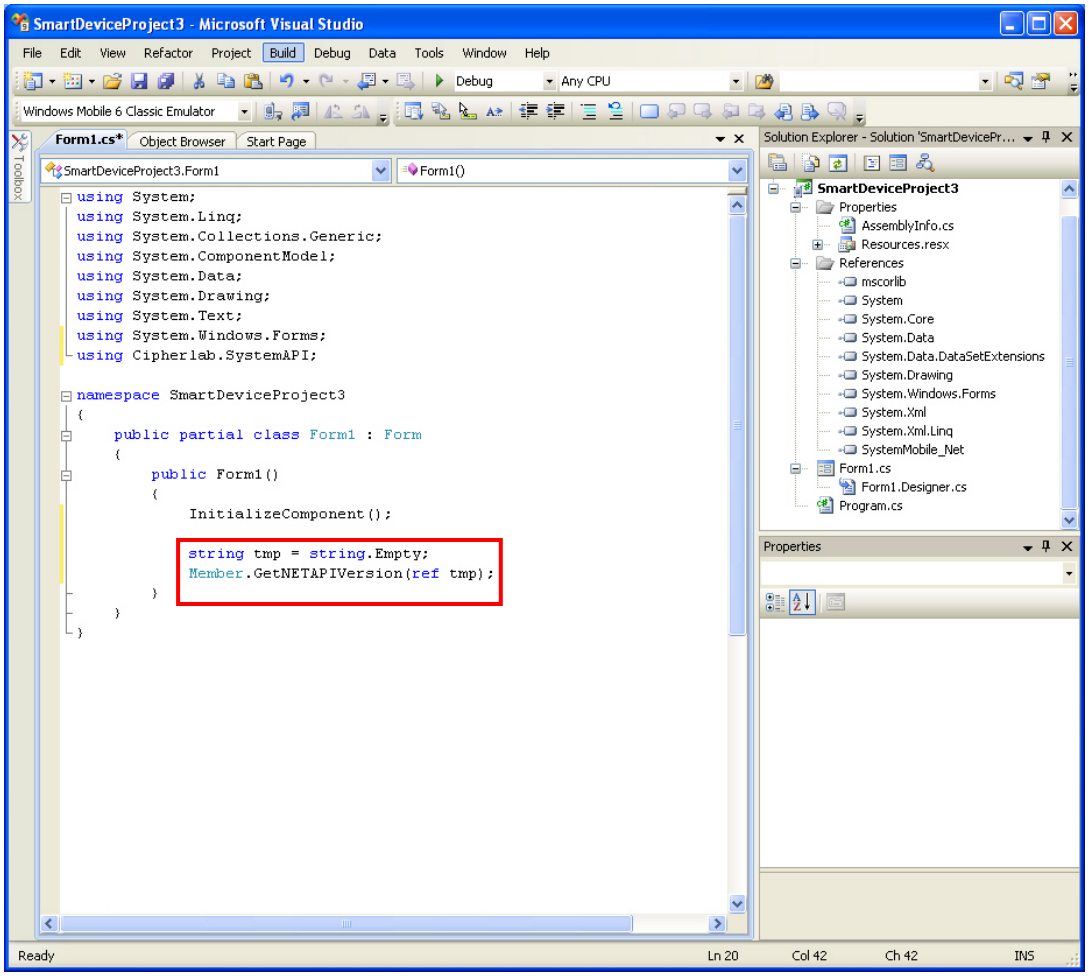
7) Click **Browse** tab. Click the system .dll file and highlight it. Click **OK** to proceed.



8) Include “using Cipherlab.SystemAPI” to use the system .dll file.



9) Start editing code…



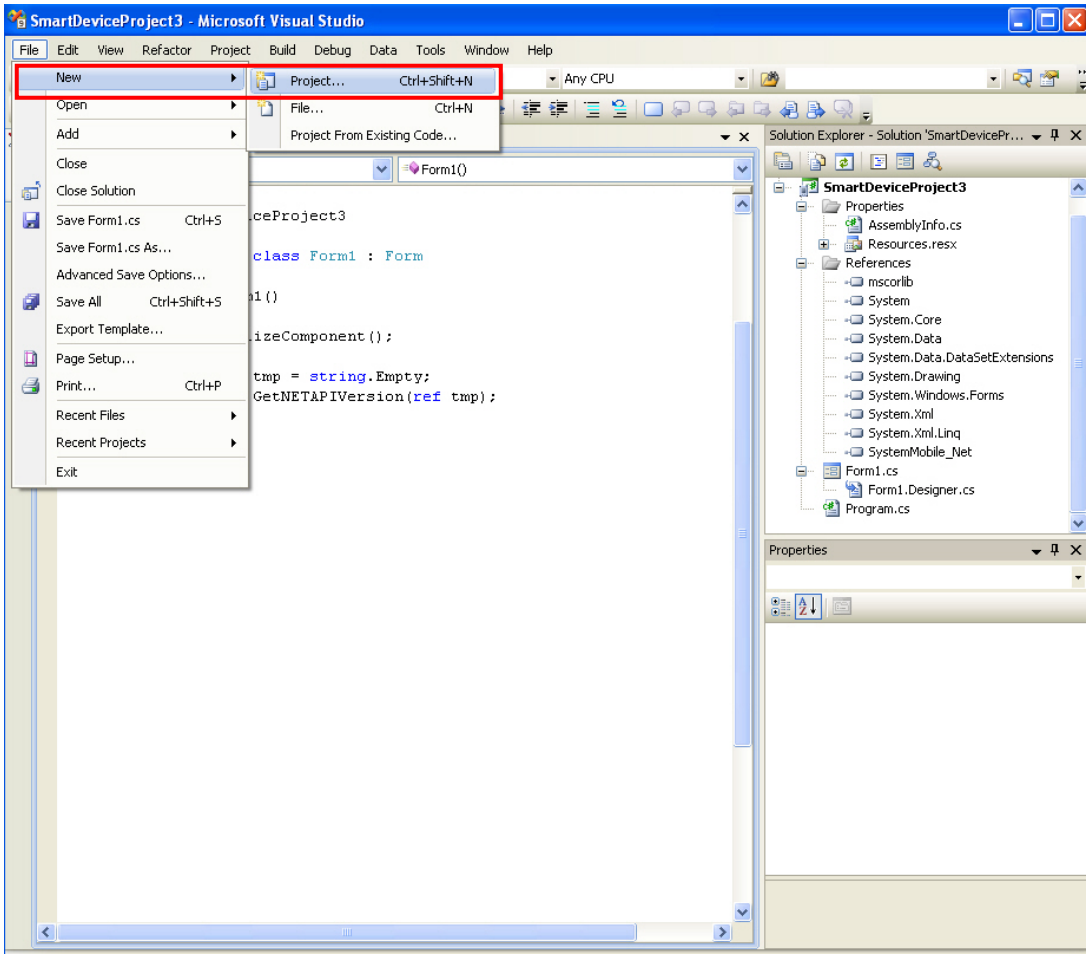
1.2. CREATE VISUAL BASIC APPLICATIONS

Follow the steps below to create a Visual Basic application in Visual Studio 2008. See Appendix II Sample Code.

I) Run Visual Studio on the PC.

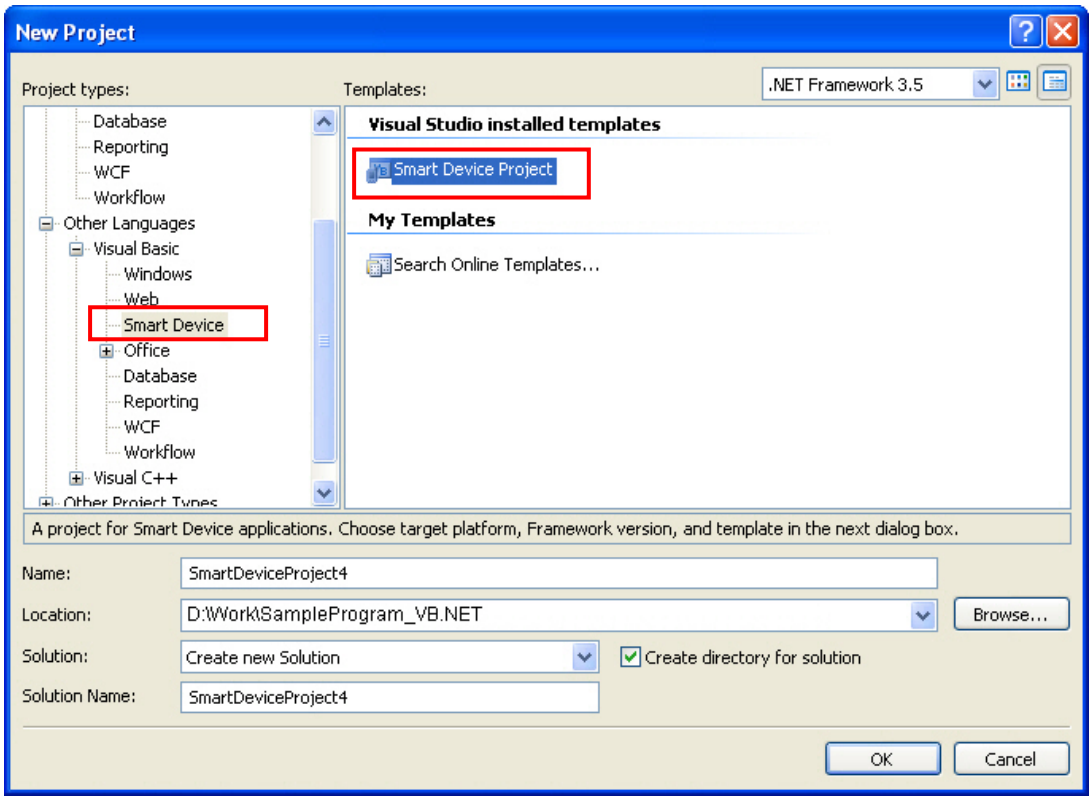
Visual Studio opens.

2) From the menu bar of Visual Studio, click **File** | **New** | **Project**.



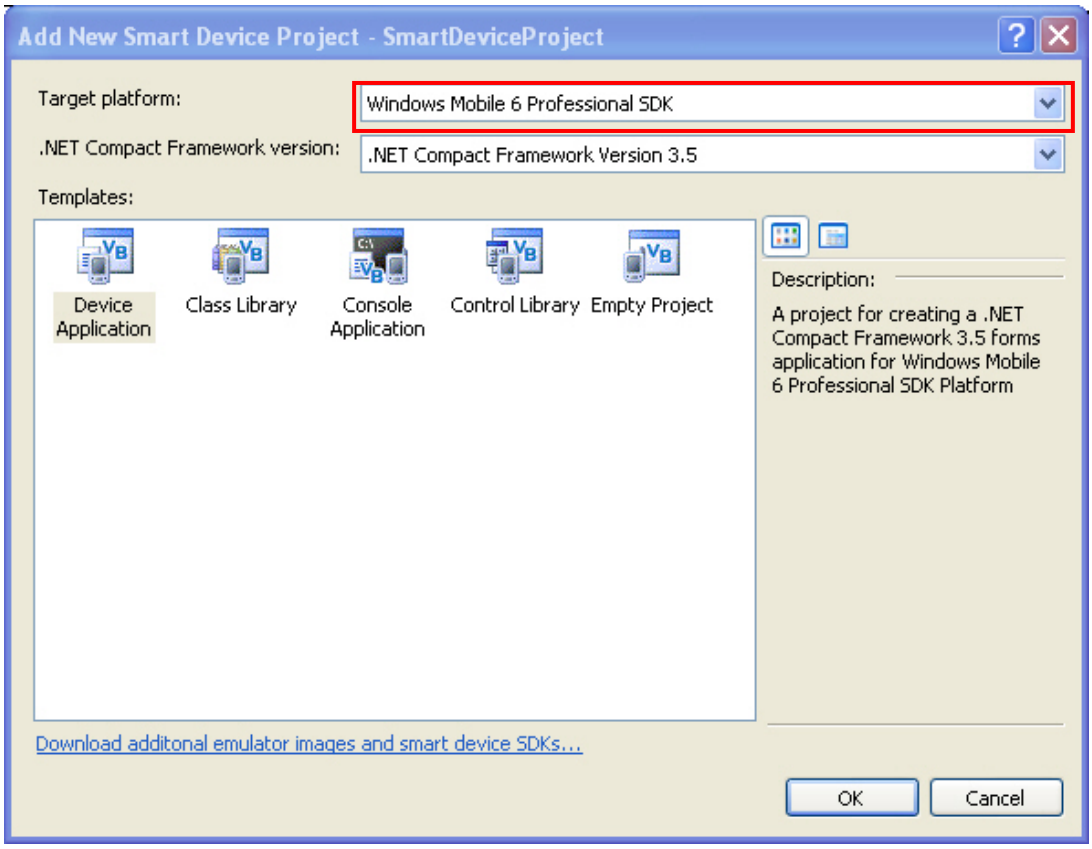
A **New Project** window opens.

3) In the left pane, select **Visual Basic** | **Smart Device**. In the right pane, select **Smart Device Project** template. Click **OK** to proceed.



An Add New Smart Device Project window opens.

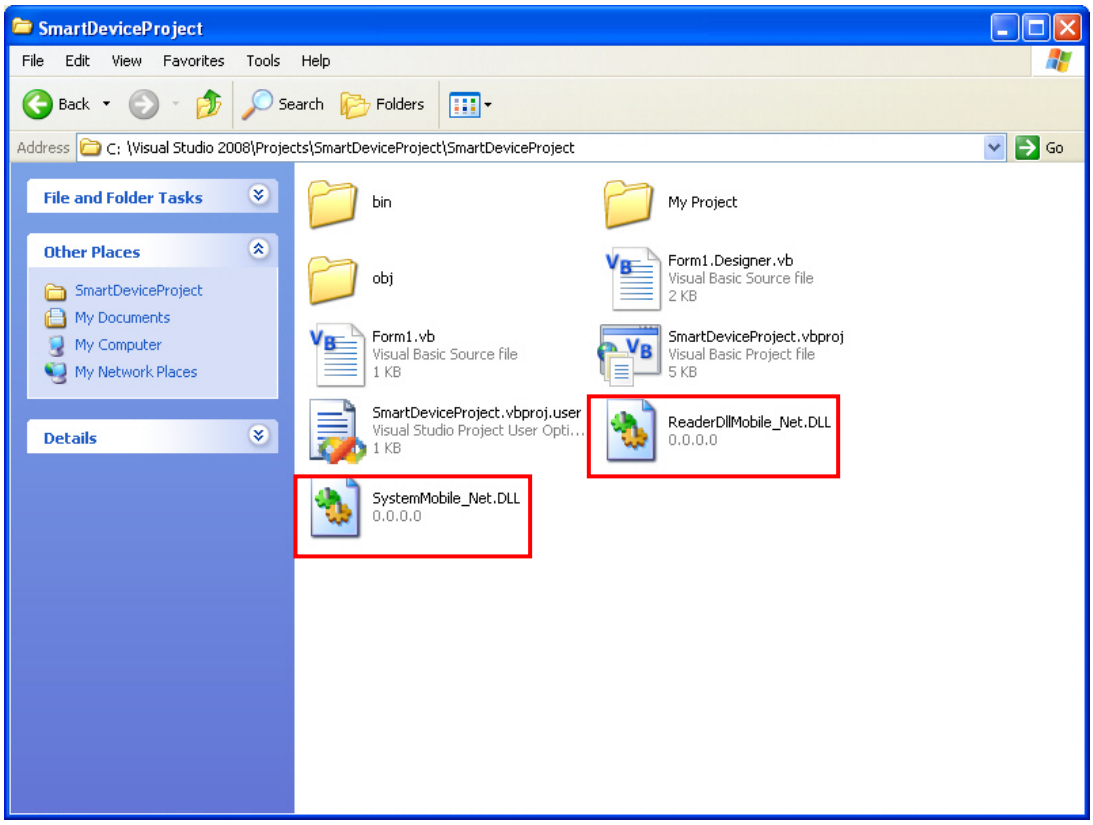
4) Select **Windows Mobile 6 Professional SDK** for target platform and **Device Application** for the template to use. Click **OK**.



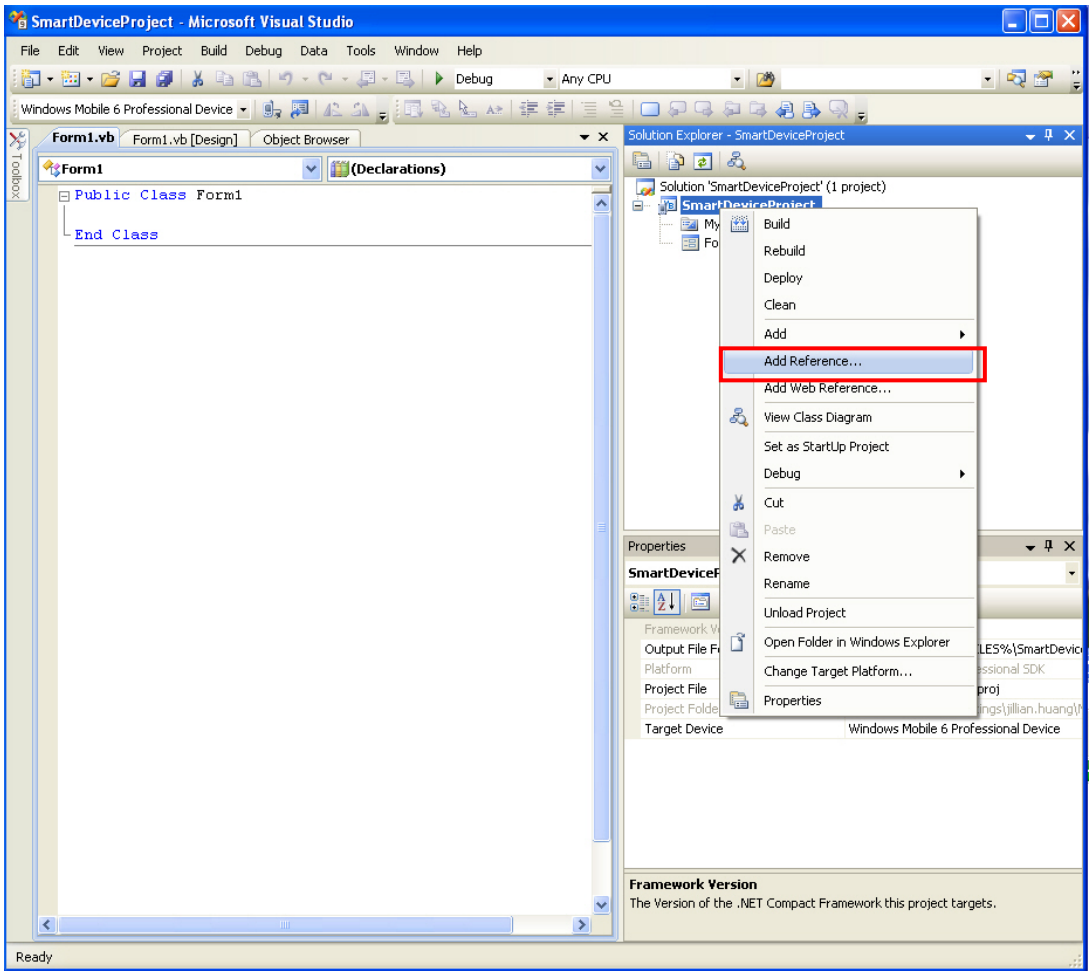
A project folder is created and new project window opens showing a prototype of template.

5) Find the CD-ROM bundled with 9200 mobile computer. Copy the following files to your new project folder:

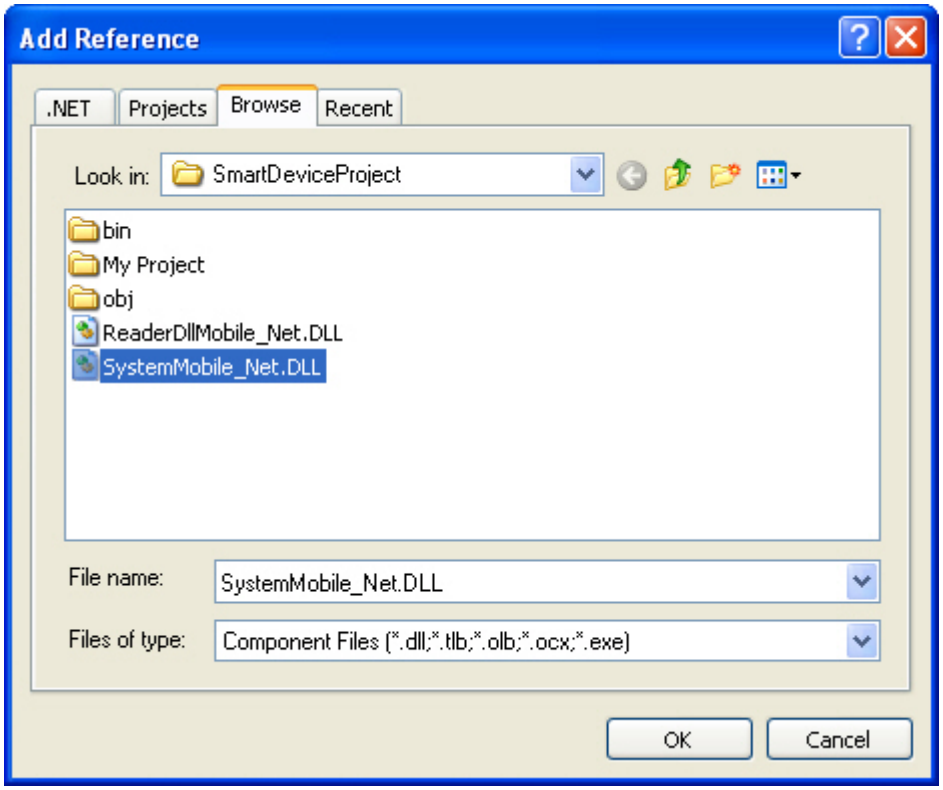
- SystemMobile_Net.dll
- ReaderDllMobile_Net.dll



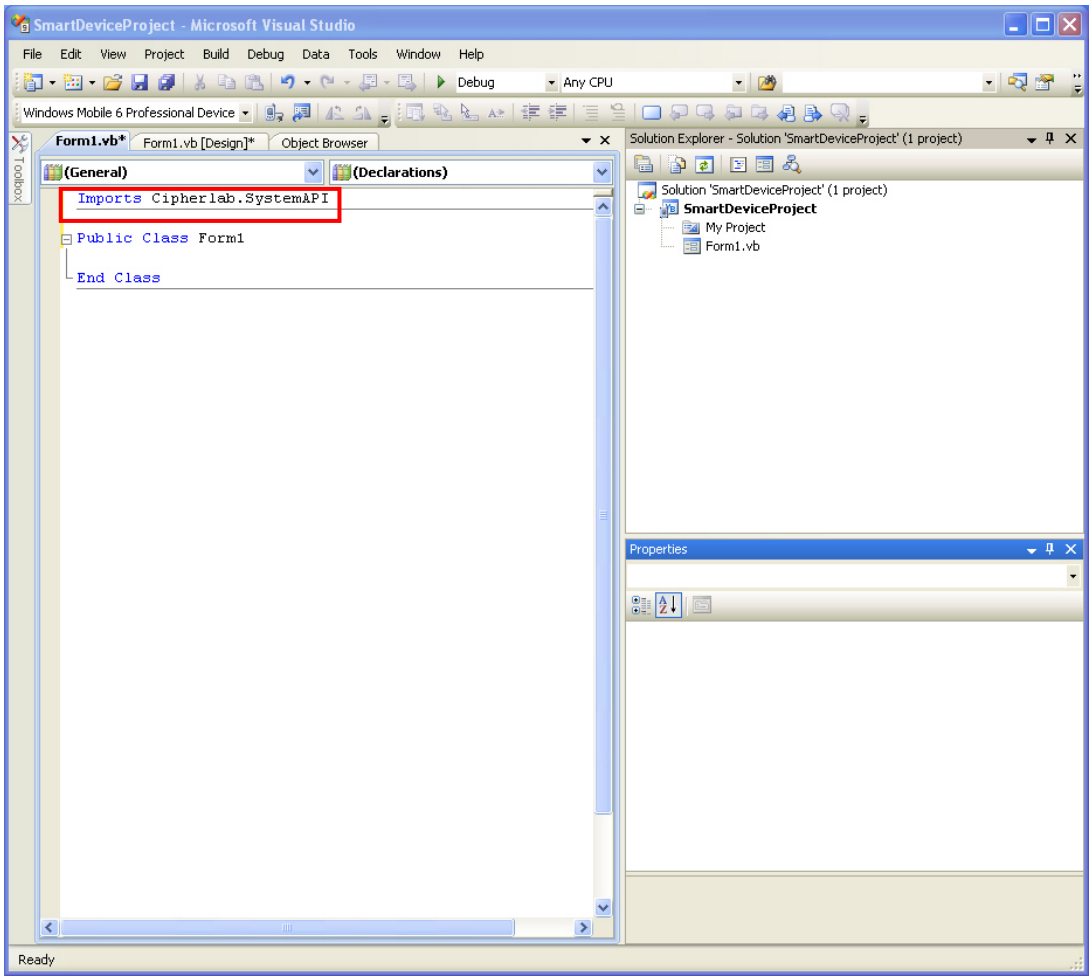
6) Back to Visual Studio. In the right pane, right-click the name of your new project and select **Add Reference**.



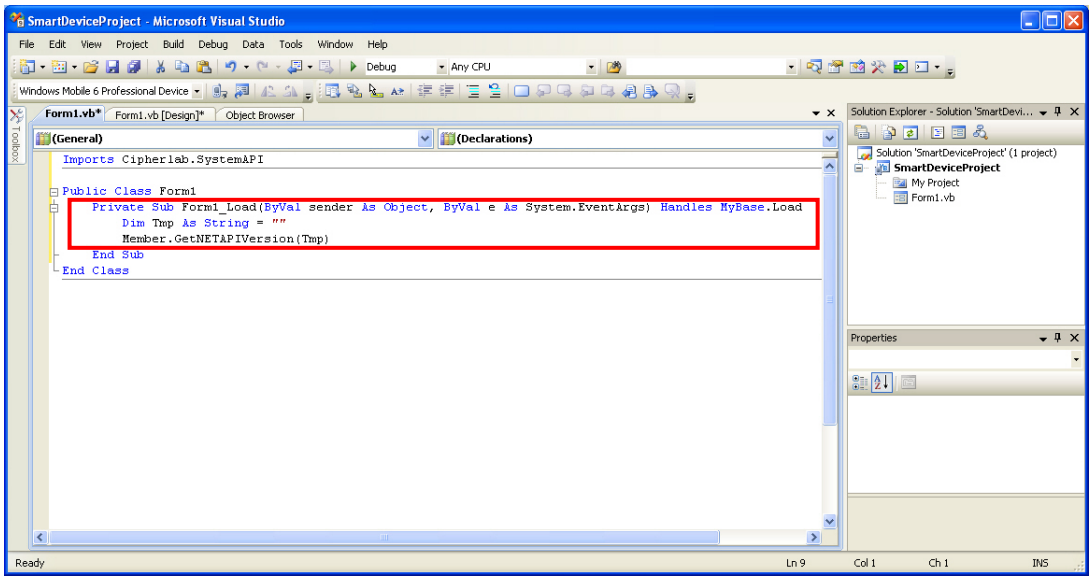
7) Click **Browse** tab. Click the system .dll file and highlight it. Click **OK**.



8) Include “Imports Cipherlab.SystemAPI” to use the system .dll file.



9) Start editing the code.



1.3. REGISTER DECODE MESSAGE

1.3.1. VISUAL C#

To retrieve the decoded data, the application has to “register” message *WM_DECODEDATA*.

For example,

```
decodeMsg = Win32API.RegisterWindowMessage("WM_DECODEDATA");
```

where *decodeMsg* is the message identifier that broadcasts when 1D (laser) reader or RFID reader decodes data.

So the following happen when decoded data comes about,

- a *WM_DECODEDATA* message broadcasts when 1D (laser) reader or RFID reader decodes data.
- your application retrieves message *WM_DECODEDATA* and uses *wParam* to identify the decoded data.

For example,

```
protected override void WndProc(ref Message msg)
          {
              if (msg.Msg == decodeMsg)
             {
                 switch (msg.WParam.ToInt32())
                {
                    case 7:  //Barcode
                           b1 = Reader.ReaderEngineAPI.GetDecodeData(ref tmp);
                           b1 = Reader.ReaderEngineAPI.GetOutputRecord(
                                     DC_READER_BC, ref tmp)
                           break;

                       case 32: //Rfid
                          b1  =  Reader.ReaderEngineAPI.GetDecodeRfidUID(ref
uuidTmp);
                       b1   =   Reader.ReaderEngineAPI.GetDecodeRfidData(ref
databuf);               b1 = Reader.ReaderEngineAPI.GetOutputRecord(
                                 DC_READER_RFID, ref tmp)

                       break;

                   default:
                       break;
                }
             }
             base.WndProc(ref msg);
```

Note: **ReadBarcodeData()** and **ReadRfidData()** scan and retrieve data without intercepting *WM_DECODEDATA*, which is applicable to scan data by tapping a GUI button. All they need to do is to invoke the function in the clicking event handler function.

Chapter 2

READER API

Reader DLL provides selected functions for your self-made application to exploit. Find the necessary files on the bundled CD-ROM before starting out writing your application.

DLL required:
ReaderDllMobile_Net.dll and ReaderDllMobile.dll

2. IN THIS CHAPTER

2.1 Initialize/Identify Reader 22
2.2 Obtain Data 26
2.3 Manipulate Status Indication 50
2.4 Obtain Essential Information 52
2.5 Configure Scan Engine — 1D CCD Scan Engine 55
2.6 Configure Scan Engine — 1D Laser Scan Engine 73
2.7 Configure Scan Engine — 2D Scan Engine 95
2.8 Reset Reader 131

2.1. INITIALIZE/IDENTIFY READER

2.1.1. INITIALIZATION

InitReader

Purpose	Initializes reader module(s).
Syntax	**int InitReader ();**
Example for C#	`int b1 = 0;` `b1 = Reader.ReaderEngineAPI.InitReader();`
Example for VB	`Dim b1 As Integer` `b1 = Reader.ReaderEngineAPI.InitReader()`
Return Value	If successful, it returns 0. Otherwise, it returns 1. ▸ Call GetErrorCode() to find the error condition encountered:

-102	E_READER_INIT_FAILED
-103	E_NO_READER_INSTALLED
-111	E_READER_BC_RESET_FAILED
-112	E_READER_RFID_RESET_FAILED

Remarks	As this is a function that gets reader module(s) ready, it must be called before any other functions. ▸ If this function fails to access reader settings, it auto-resets the reader module(s). ▸ Call GetReaderType to check the reader module integrated on the mobile computer.
See Also	GetActiveDevice, GetReaderType, SetActiveDevice

2.1.2. ACTIVE DEVICE

GetActiveDevice

Purpose	Gets currently active reader type.
Syntax	**int GetActiveDevice ();**
Example for C#	`int dev = 0;` `dev = Reader.ReaderEngineAPI.GetActiveDevice();`
Example for VB	`Dim dev As Integer` `dev = Reader.ReaderEngineAPI.GetActiveDevice()`
Return Value	If successful, it returns the reader type(s) accordingly:

7	DC_READER_BC	Only barcode reader is active.
32	DC_READER_RFID	Only RFID reader is active. (= Barcode reader is inactive.)
255	ALL_DEVICE	Both readers are active.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

1	NO_ACTIVE_DEVICE
-101	E_READER_NOT_INIT

Remarks

Reader activity allows three possibilities:

- Only barcode reader (1D or 2D) is active.
- Only RFID reader is active.
- Both barcode reader and RFID reader are active.

See Also: GetReaderType, InitReader, SetActiveDevice

SetActiveDevice

Purpose	Sets the reader module to use.
Syntax	**int SetActiveDevice (int** *target***);**
Parameters	*target* [in] A value that specifies the active device.

1	NO_ACTIVE_DEVICE	Disables both active devices.
7	DC_READER_BC	Activates only barcode reader. (=RFID reader is inactive.)
32	DC_READER_RFID	Activates only RFID reader. (= Barcode reader is inactive.)
255	ALL_DEVICE	Activates both readers.

Example for C#

```
int b1 = 0;
b1 = Reader.ReaderEngineAPI.SetActiveDevice(7);
```

Example for VB

```
Dim b1 As Integer
b1 = Reader.ReaderEngineAPI.SetActiveDevice(7)
```

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-101	E_READER_NOT_INIT

Remarks

Trigger key's triggering behavior applies only to the active device. No event occurs with the inactive device.

There is no error signal if the reader to set is not found.

See Also

GetReaderType, GetActiveDevice, InitReader

2.1.3. READER TYPE

GetReaderType

Purpose	Gets the available reader type(s)
Syntax	**int GetReaderType ();**
Example for C#	`int type = 0;` `type = Reader.ReaderEngineAPI.GetReaderType();`
Example for VB	`Dim type As Integer` `type = Reader.ReaderEngineAPI.GetReaderType()`
Return Value	If successful, it returns the reader type(s) accordingly:

128	ID_MOD_1D_955
256	ID_MOD_MP_RFID
512	ID_MOD_2D_4500
1024	ID_MOD_1D_SM1
384	ID_MOD_1D_955 + ID_MOD_MP_RFID
768	ID_MOD_2D_4500 + ID_MOD_MP_RFID
1280	ID_MOD_1D_SM1 + ID_MOD_MP_RFID

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

0	Could be (1) reader not initialized, or (2) no reader installed!

Remarks

Reader integration allows three possibilities:

- Only barcode reader (1D or 2D) is integrated.
- Only RFID reader is integrated.
- Both barcode reader and RFID reader are integrated.

See Also

InitReader, GetActiveDevice, SetActiveDevice

2.2. OBTAIN DATA

2.2.1. DATA OUTPUT SETTINGS

Processed Data

DataOutputSettings() sets which info to attach to a decoded barcode data:

Info	Length (Byte)	Description
Code Type	1	Barcode Type. See **DataOutputSettings()** paramenter *showCodeType*.
Code Length	4	Decoded barcode length (excluding prefix and/or suffix codes). See **DataOutputSettings()** paramenter *showCodeLen*.
Prefix Code	0~10	No prefix code if value is 0. See **DataOutputSettings()** parameter *prefixCode* .
Decode Data	1~4096	Decoded barcode data
Suffix Code	0~10	No suffix code if value is 0. See **DataOutputSettings()** parameter *suffixCode*.

- When keyboard emulation is enabled, barcode data is output to the active application locally.
- When keyboard emulation is disabled, use **GetOutputRecord()** to get decoded data upon receiving message *WM_DECODEDATA*. See Register Decode Message.

Note: Data fields by sequence may include –
[Code Type], [Code Length], [Prefix Code], [Decode Data], [Suffix Code]

DataOutputSettings() and **RfidSettings()** set which info to attach to a decoded RFID data:

Info	Length (Byte)	Description
Tag Type	1	RFID Tag type. See **DataOutputSettings()** paramenter *showCodeType*.
Tag Length	4	Decoded data length (excluding prefix and/or suffix codes). See **DataOutputSettings()** parameter *showCodeLen*. Tag length may be - UID only - Data only - UID + Data
Prefix Code	0~10	No prefix code if value is 0. See **DataOutputSettings()** parameter *prefixCode* .
Tag UID	1~20	RFID tag's Universal Identification (UID). See **RfidSettings()** parameter *readUid*.
Suffix Code	0~10	No suffix code if value is 0. See **DataOutputSettings()** parameter *suffixCode*.
Delimiter	1	The character that separates UID and data. See **RfidSettings()** parameter *useDelim*. No delimiter if value is 0x00.
Tag Data	1~4096	Decoded RFID data. See **RfidSettings()** parameter *readUserData*.

- When keyboard emulation is enabled, RFID data is output to the active application locally.
- When keyboard emulation is disabled, use **GetOutputRecord()** to obtain processed data upon receiving message *WM_DECODEDATA*. See Register Decode Message.

Note: Data fields by sequence may include –
[Tag Type], [Tag Length], [Prefix Code], [Tag UID], [Suffix Code][Delimiter][Tag Data]. Note if Tag UID is disabled, output data will not include prefix code, UID, suffix code, and delimiter.

Raw Data

Use the following functions to obtain raw data —

- **GetDecodeData()** and **GetDecodeType()** for printed barcodes.
- **GetDecodeRfidData()**, **GetDecodeRfidUID()** and **GetDecodeTagType()** for RFID tags.

DataOutputSettings

Purpose

Sets where and how to ouptut data.

- "How to output data" means which info to attach to a decoded data such as code/tag type, data length, prefix code, suffix code, etc.

Syntax

```
int DataOutputSettings (int rw,
                        ref int enableKeyboardEmulation,
                        ref int autoEnterWay,
                        ref int autoEnterChar,
                        ref int showCodeType,
                        ref int showCodeLen,
                        ref string prefixCode,
                        ref string suffixCode);
```

Parameters

A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets output settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets output settings

enableKeyboardEmulation

[in][out] A value that specifies whether to emulate data as typed text to subsequently pass it to the active application.

0*	Disable keyboard emulation - No other parameters will take effect unless **GetOutputRecord()** is called.
1	Emulates keyboard input on local machine
2	Emulates keyboard input to remote PC (RDP server via programs like Remote Desktop Connection, Terminal Services Client, etc.)

autoEnterWay

[in][out] A value that specifies whether to auto-affix a character after decoding.

0	Disables
1*	Suffixes the decoded data (= decoded data + Enter-character)
2	Prefixes the decoded data (= Enter-character + decoded data)

autoEnterChar

[in][out] A value that specifies the character to auto-affix.

0	None
1*	Carriage Return (= 0x0d)
2	Tab
3	Space (= 0x20)
4	Comma (= 0x2c)
5	Semicolon (= 0x3b)

showCodeType

[in][out] A value that specifies whether to transmit barcode type or RFID tag type

in data records.

0*	Does not transmit
1	Transmits

showCodeLen

[in][out] A value that specifies whether to transmit code length for a barcode or RFID tag in data records.

- In case only UID is to be read, code length is UID length.

0*	Does not transmit
1	Transmits

prefixCode

[in][out] A string variable that stores prefix code.

suffixCode

[in][out] A string variable that stores suffix code.

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

Remarks

Depending on reader type and associated reader setting, the fields of output record may differ.

When data comes from barcode reader, data fields may include:

`[Code Type][Code Length][Prefix Code][Decode Data][Suffix Code]`

[Code Type]:	This field is output only when *showCodeType* value is non-zero.
[Code Length]:	This field is output only when *showCodeLen* value is non-zero. (Prefix/suffix codes are not included.)
[Prefix Code]:	This field is output only when *prefixCode* is non-zero.
[Decode Data]:	This field is output only when data has been decoded.
[Suffix Code]:	This field is output only when *suffiixCode* value is non-zero.

When data comes from RFID reader, data fields may include:

- If UID is disabled, output will not include prefix code, UID, suffix code, and delimiter.

`[Tag Type][Tag Length][Prefix Code][Tag UID][Suffix Code][Delimiter][Tag Data]`

[Tag Type]:	This field is output only when *showCodeType* value is non-zero.
[Tag Length]:	This field is output only when *ShowCodeLen* value is non-zero. When specified to read UID only, tag length is UID length. (Prefix/suffix codes are not included.)
[Prefix Code]:	This field is output only when *prefixCode* value is non-zero.
[Tag UID]:	This field is output only when **RfidSettings()** *readUID* value is non-zero.
[Suffix Code]:	This field is output only when *suffiixCode* value is non-zero.
[Delimiter]:	This field is output only when **RfidSettings()** *useDelim* value is non-zero.
[Tag Data]:	This field is output only when **RfidSettings()** *readData* value

	is non-zero.

See Also

GetDecodeData, GetDecodeType, GetDecodeRfidData, GetDecodeRfidUID, GetDecodeTagType, GetOutputRecord, RfidSettings

GetOutputRecord

Purpose

Sets how to output data for keyboard emulation

- Means setting which info to attach to decoded data such as code/tag type, code length, prefix code, suffix code, etc.

Syntax

int GetOutputRecord (int *target*,
ref string *buf*,
int *bufSize*);

Parameters

target

[in] A value that specifies where the decoded data comes from.

7	DC_READER_BC	from barcode reader.
32	DC_READER_RFID	from RFID reader.

buf

[out] Pointer to a buffer where decoded data is stored.

bufSize

[in] Maximum number of bytes to receive.

Example for C#

```
int b1 = 0;
string buf = string.Empty;
int bufSize = 5000;
b1 = Reader.ReaderEngineAPI.GetOutputRecord(DC_READER_BC, ref buf, bufSize);
```

Example for VB

```
Dim b1 As Integer
Dim buf As String = ""
Dim bufSize As Integer = 5000
b1 = Reader.ReaderEngineAPI.GetOutputRecord(DC_READER_BC, buf, bufSize)
```

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-101	E_READER_ _INIT
-300	E_BUFF_IS_TOO_SMALL

Remarks

This function is called before another successful decoding to obtain the data just decoded. Depending on reader type and associated reader settings, output may differ.

When data comes from barcode reader, data fields may include:

`[Code Type][Code Length][Prefix Code][Decode Data][Suffix Code]`

[Code Type]:	This field is output only when **DataOutputSettings()** *showCodeType* value is non-zero.
[Code Length]:	This field is output only when **DataOutputSettings()** *showCodeLen* value is non-zero. (Prefix and/or suffix codes are not included.)
[Prefix Code]:	This field is output only when **DataOutputSettings()** *prefixCode* value is non-zero.
[Decode Data]:	This field is output when data gets decoded.
[Suffix Code]:	This field is output only when **DataOutputSettings()**

	suffiixCode value is non-zero.

When data comes from RFID reader, data fields may include:

- Note if UID is disabled, output will not include prefix code, UID, suffix code, and delimiter.

```
[Tag Type][Tag Length][Prefix Code][Tag UID][Suffix Code][Delimiter][Tag Data]
```

[Tag Type]:	This field is output only when **DataOutputSettings()** *showCodeType* value is non-zero.
[Tag Length]:	This field is output only when **DataOutputSettings()** *ShowCodeLen* value is non-zero. When specified to read UID only, tag length is UID length. (Prefix and/or suffix codes are not included.)
[Prefix Code]:	This field is output only when **DataOutputSettings()** *prefixCode* value is non-zero.
[Tag UID]:	This field is output only when **RfidSettings()** *readUID* value is non-zero.
[Suffix Code]:	This field is output only when **DataOutputSettings()** *suffiixCode* value is non-zero.
[Delimiter]:	This field is output only when **RfidSettings()** *useDelim* value is non-zero.
[Tag Data]:	This field is output only when **RfidSettings()** *readData* value is non-zero.

See Also

GetDecodeData, GetDecodeType, GetDecodeRfidData, GetDecodeRfidUID, GetDecodeTagType, DataOutputSettings, RfidSettings

2.2.2. RFID SETTINGS

RfidSettings

Purpose

Sets what to collect, where to start collecting from, and how much data to collect from an RFID tag, and sets how to output RFID data.

- "How to output RFID data" means which info to attach to a decoded RFID data such as tag length, prefix code, tag UID, suffix code, delimiter.

Syntax

int RfidSettings (int *rw*,
ref int *readUid*,
ref int *readUserData*,
ref int *userDataStartByte*,
ref int *readUserDataLen*,
ref int *useDelim*,
ref int *timeout*,
ref string *mifareLoginKey*,
ref int *loginKeyType*);

Parameters

A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets RFID settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets RFID settings

readUID

[in][out] A value that specifies whether to decode UID.

0	Does not decode
1*	Decodes

readUserData

[in][out] A value that specifies whether to decode RFID data.

0*	Does not decode
1	Decodes

userDataStartByte

[in][out] A value that specifies where to start collecting data from.

As different RFID tags may have different capacity, when a value greater than allowed is passed for this parameter, it is auto-adjusted to the maximum value allowed on a tag.

-1* A value that specifies where to start reading from. "-1" means the RFID reader starts reading from the default page of an RFID tags. The following is a table of various RFID tags' default pages:

Tag type	Standard	Default Page
Mifare	(ISO 14443A)	**4**
ICODE SLI	(ISO 15693)	**3**
LRI512	(ISO 15693)	**0**
SRF55VxxP	(ISO 15693)	**3**

	Tag-it	(ISO 15693)	**0**
	Others	(ISO 15693)	**0**
	ICODE	(Phillips)	**5**

readUserDataLen

[in][out] A value that specifies how much data to collect, which is set to 128 bytes by default. As the capacity varies from tag to tag, when the passed value is greater than allowed, it is auto-adjusted to the maximum value allowed on an RFID tag.

userDelim

[in][out] An ASCII value that specifies the delimiter in use.

0*	NO delimiter
1~127	Adds a delimiter between UID and data when both are decoded

timeout

[in][out] A value that specifies the elapsed time before a scanning session* ends.

1~5	3* (second)

mifareLoginKey

[in][out] A string variable that stores the security key in Reader DLL for accessing a specific RFID tag, such as Mifare. It is set to “FFFFFFFFFFFF” by default.

- This key must be a hex string with 12 bytes length.
- This login key is applied to all the sectors covered by the range formulated by “userDataStartByte” and “readUserDataLen”.

LoginKey Example	*Key String*
0	(Keeps the current string)
“F1F2F3F4F5F6”	F1F2F3F4F5F6

loginKeyType

[in][out] A value that specifies login key type.

0*	Key A
1	Key B

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

Remarks

Data fields may include:

`[Tag Type][Tag Length][Prefix Code][Tag UID][Suffix Code][Delimiter][Tag Data]`

[Tag Type]:	This field is output only when **DataOutputSettings()** *showCodeType* value is non-zero.
[Tag Length]:	This field is output only when **DataOutputSettings()** *ShowCodeLen* value is non-zero. When RFID reader is set to read UID only, UID length applies to tag length. (Prefix/suffix codes are not included.)
[Prefix Code]:	This field is output only when **DataOutputSettings()** *prefixCode* value is non-zero.

[Tag UID]:	This field is output only when **RfidSettings()** *readUID* value is non-zero.
[Suffix Code]:	This field is output only when **DataOutputSettings()** *suffiixCode* value is non-zero.
[Delimiter]:	This field is output only when **RfidSettings()** *useDelim* value is non-zero.
[Tag Data]:	This field is output only when **RfidSettings()** *readData* value is non-zero.

- Note if UID is disabled, output will not include prefix code, UID, suffix code, and delimiter.

See Also ChangeMifareKey, GetDecodeRfidData, GetDecodeRfidUID GetDecodeTagType, DataOuputSettings, GetOutputRecord

Note: (1) A scanning session starts when physical trigger key is pressed.
(2) The specified timeout should be greater than the time it takes to press the trigger.
(3) RFID tags may support authentication for security concerns, such as Mifare Standard 1K/4K, and so on.

ChangeMifareKey	
Purpose	Changes Mifare card security key — Key A or Key B of a specific sector.
Syntax	**int ChangeMifareKey (int** *keyType*, **int** *useDefaultKey*, **int** *defaultKeyIndex*, **int** *targetSector*, **byte[]** *loginKey*, **byte[]** *newKeyA*, **byte[]** *newKeyB*, **byte[]** *accessControl*);
Parameters	*loginKeyType* Reserved [in] Pass 0 to ignore this parameter. *useDefaultKey* Reserved [in] Pass 0 to ignore this parameter. *defaultKeyIndex* Reserved [in] Pass 0 to ignore this parameter. *targetSector* [in] The sector to apply change to Mifare security key. *loginKey* [in] Pass Key A that logs in the target sector. *newKeyA* [in] New Key A to log in the target sector. If not chaning key A, pass old Key A instead. This parameter must be a character array of 6 bytes length. *newKeyB* [in] New Key B to log in the target sector. If not chaning key B, pass old Key B instead. This parameter must be a character array of 6 bytes length. *accessControl* Reserved [in] Pass 0 to ignore this parameter.

Example for C#	`int b1 = 0, loginKeyType = 0, useDefaultKey = 0, defaultKeyIndex = 0;`
	`int targetSector = 1, i = 0;`
	`byte loginKey = new byte[6];`
	`byte newKeyA = new byte[6];`
	`byte newKeyB = new byte[6];`
	`byte accessControl = new byte[4];`
	`for (i = 0; i < 6; i++)`
	`{`
	`loginKey[i] = 255;`
	`newKeyA[i] = 255;`
	`newKeyB[i] = 255;`
	`}`
	`newKeyB[5] = 254;`
	`b1 = Reader.ReaderEngineAPI.ChangeMifareKey(`
	`loginKeyType, useDefaultKey, defaultKeyIndex, targetSector, loginKey, newKeyA, newKeyB, accessControl);`

Example for VB

```
Dim loginKeyType As Integer = 0
Dim useDefaultKey As Integer = 0
Dim defaultKeyIndex As Integer = 0
Dim targetSector As Integer = 1
Dim i As Integer = 0
Dim b1 As Integer
Dim loginKey(5) As Byte
Dim newKeyA(5) As Byte
Dim newKeyB(5) As Byte
Dim accessControl(3) As Byte
for i = 0 to 5
loginKey(i) = 255
newKeyA(i) = 255
newKeyB(i) = 255
Next
newKeyB(5) = 254
b1 = Reader.ReaderEngineAPI.ChangeMifareKey(
loginKeyType, useDefaultKey, defaultKeyIndex, targetSector,
loginKeyA, newKeyA, newKeyB, accessControl)
```

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

Remarks

Due to security concerns, security keys cannot be read. Make sure they are presented correctly.

See Also

RfidSettings

2.2.3. BARCODE DATA

Use **GetDecodeData()** and **GetDecodeType()** to get raw data from a barcode.

GetDecodeData	
Purpose	Reads barcode data upon pressing a physical trigger.
Syntax	**int GetDecodeData (ref string** *buf***);** **int** *bufSize***);**
Parameters	*buf* [out] Pointer to a buffer where decoded data is stored. *bufSize* [in] Maximum number of bytes to receive.

Example for C#	`int b1 = 0;`
	`string buf = “”;`
	`int bufSize = 5000;`
	`b1 = Reader.ReaderEngineAPI.GetDecodeData(ref buf, bufSize);`

Example for VB	`Dim b1 As Integer`
	`Dim buf As String = “”`
	`Dim bufSize As Integer = 5000`
	`b1 = Reader.ReaderEngineAPI.GetDecodeData(buf, bufSize)`

Return Value	If successful, it returns 0. Otherwise, it returns 1. ▸ Call GetErrorCode() to find the error condition encountered:
Remarks	This function is called to obtain the just decoded data right before another successful decoding. The decoded data then hooks up Prefix Code or Suffix Code if they are applicable.
See Also	DataOuputSettings, GetDecodeType, GetOutputRecord

-101	E_READER_NOT_INIT
-300	E_BUFF_IS_TOO_SMALL

GetDecodeType

Purpose	Reads barcode type upon pressing a physical trigger.
Syntax	**int GetDecodeType ();**
Example for C#	`int codeType = 0;` `codeType = Reader.ReaderEngineAPI.GetDecodeType();`
Example for VB	`Dim codeType As Integer` `codeType = Reader.ReaderEngineAPI.GetDecodeType()`
Return Value	If successful, it returns the code type. Otherwise, it returns 1. ▸ Call GetErrorCode() to find the error condition encountered:

-101	E_READER_NOT_INIT

Remarks: Before another successful decoding, this function is called to obtain the code type of the currently decoded barcode.

Code Type		Symbology
0x40	64 (@)	ISBT 128
0x41	65 (A)	Code 39
0x42	66 (B)	Italian Pharmacode (Code 32)
0x43	67 (C)	French Pharmacode (CIP 39)
0x44	68 (D)	Industrial 25
0x45	69 (E)	Interleaved 25
0x46	70 (F)	Matrix 25
0x47	71 (G)	Codabar (NW7)
0x48	72 (H)	Code 93
0x49	73 (I)	Code 128
0x4A	74 (J)	UPC-E0
0x4B	75 (K)	UPC-E0 with Addon 2
0x4C	76 (L)	UPC-E0 with Addon 5
0x4D	77 (M)	EAN-8
0x4E	78 (N)	EAN-8 with Addon 2
0x4F	79 (O)	EAN-8 with Addon 5
0x50	80 (P)	EAN-13 (also UPC-A on CCD/Laser scan engine)
0x51	81 (Q)	EAN-13 with Addon 2
0x52	82 (R)	EAN-13 with Addon 5
0x53	83 (S)	MSI
0x54	84 (T)	Plessey
0x55	85 (U)	GS1-128 (EAN-128)
0x56	86 (V)	Undefined
0x57	87 (W)	Undefined

0x58	88 (X)	Undefined
0x59	89 (Y)	Undefined
0x5A	90 (Z)	Telepen
0x5B	91 ([ )	GS1 DataBar Omnidirectional (RSS-14)
0x5C	92 (\)	GS1 DataBar Limited (RSS Limited)
0x5D	93 ( ])	GS1 DataBar Expanded (RSS Expanded)
0x5E	94 (^)	UPC-A
0x5F	95 (_)	UPC-A with Addon 2
0x60	96 (‘)	UPC-A with Addon 5
0x61	97 (a)	UPC-E1
0x62	98 (b)	UPC-E1 with Addon 2
0x63	99 (c)	UPC-E1 with Addon 5
0x64	100 (d)	TLC 39 (TCIF Linked Code 39)
0x65	101 (e)	Trioptic (Code 39)
0x66	102 (f)	Bookland (EAN)
0x67	103 (g)	Code 11
0x68	104 (h)	Code 39 Full ASCII
0x69	105 (i)	IATA[Note] (Code 25 used on flight tickets)
0x6A	106 (j)	Industrial 25 (Discrete 25)
0x6B	107 (k)	PDF417
0x6C	108 (l)	MicroPDF417
0x6D	109 (m)	Data Matrix
0x6E	110 (n)	Maxicode
0x6F	111 (o)	QR Code
0x70	112 (p)	US Postnet
0x71	113 (q)	US Planet
0x72	114 (r)	UK Postal
0x73	115 (s)	Japan Postal
0x74	116 (t)	Australian Postal
0x75	117 (u)	Dutch Postal
0x76	118 (v)	Composite Code
0x77	119 (w)	Macro PDF
0x78	120 (x)	Coupon Code
0x79	121 (y)	Chinese 25
0x7A	122 (z)	Aztec
0x7B	123 ({)	MicroQR

0x7C	124 (|)	USPS 4CB / One Code / Intelligent Mail
0x7D	125 (})	UPU FICS Postal
0x7E	126 (~)	Macro MicroPDF417

See Also DataOuputSettings, GetDecodeData, GetOutputRecord

Note: IATA stands for International Air Transport Association, and this barcode type is used on flight tickets.

ReadBarcodeData

Purpose	Reads barcode data by tapping GUI button (without intercepting *WM_DECODEDATA*).
Syntax	**int ReadBarcodeData (ref int** *codeType*, **ref string** *outBuf*, **int** *bufSize*, **int** *timeout*);
Parameters	*codeType* [out] Pointer to a variable that stores barcode type. *outBuf* [out] Pointer to a buffer where barcode data is stored. *bufSize* [in] Maximum number of bytes to receive. *timeout* [in][out] A value that specifies the elapsed time before a decode session ends.

1~9	3* (second)

Example for C#

```
int b1 = 0;
int codeType = 0;
string outBuf = “”;
int bufSize = 5000;
int timeout = 3;
b1 = Reader.ReaderEngineAPI.ReaderBarcodeData(ref codeType, ref outBuf, bufSize, timeout);
```

Example for VB

```
Dim b1 As Integer
Dim codeType As Integer
Dim outBuf As String = “”
Dim bufSize As Integer = 5000
Dim timeout As Integer = 3
b1 = Reader.ReaderEngineAPI.ReadBaecodeData(codeType, outBuf, bufSize, timeout)
```

Return Value

If successful, it returns the number of bytes obtained (excluding terminating null character).

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

0	E_READER_UNKNOWN_ERROR
-101	E_READER_NOT_INIT

Remarks

This function scans and retrieves data without intercepting *WM_DECODEDATA*. For example, if users want to trigger scanning by tapping a GUI button, all they need to do is to invoke this function in the clicking event handler function.

See Also

DataOutputSettings, GetOutputRecord

2.2.4. RFID DATA

Use **GetDecodeRfidData()**, **GetDecodeRfidUID()** and **GetDecodeTagType()** are the functions to obtain raw data from an RFID tag. Some RFID tags can be read and written on a page-by-page basis. Use **ReadRfidData()** and **WriteRfidData()** to read or write data from/to an RFID tag.

As different RFID tags may have different page length, said functions auto-truncate data to meet page length.

An RFID tag's default page to start reading data from, or writing data to:

Tag Type	Start Page	Standard
Mifare	4	(ISO 14443A)
ICODE SLI	3	(ISO 15693)
LRI512	0	(ISO 15693)
SRF55VxxP	3	(ISO 15693)
Tag-it	0	(ISO 15693)
Others	0	(ISO 15693)
ICODE	5	(Phillips)

Note: Different tag types may have different start page and page capacity. See the spec. of the RFID tag to read/write for its memory organization.

GetDecodeRfidData

Purpose	Reads RFID data upon pressing a physical trigger.
Syntax	**int GetDecodeRfidData (ref string** *buf***);**
Parameters	*buf* [out] Pointer to a buffer where decoded data is stored.

Example for C#	`int b1 = 0;`
	`string szData = string.Empty;`
	`b1 = Reader.ReaderEngineAPI.GetDecodeRfidData(ref szData);`

Example for VB	`Dim b1 As Integer`
	`Dim szData As String = ""`
	`b1 = Reader.ReaderEngineAPI.GetDecodeRfidData(szData)`

Return Value	If successful, it returns 0. Otherwise, it returns 1. ▸ Call GetErrorCode() to find the error condition encountered:

-101	E_READER_NOT_INIT
-300	E_BUFF_IS_TOO_SMALL

Remarks	This function is called to obtain the just decoded data right before another successful decoding. The decoded data then hooks up Prefix Code and/or Suffix Code if they are applicable.
See Also	GetDecodeRfidUID, GetDecodeTagType, DataOutputSettings, GetOutputRecord, RfidSettings

GetDecodeRfidUID

Purpose	Reads UID upon pressing a physical trigger.
Syntax	**int GetDecodeRfidUID (ref string** *buf***);**
Parameters	*buf* [out] Pointer to a buffer where UID is stored.

Example for C#	`int b1 = 0;`
	`string szUID = string.Empty;`
	`b1 = Reader.ReaderEngineAPI.GetDecodeRfidUID(ref szUID);`

Example for VB	`Dim b1 As Integer`
	`Dim szUID As String = ""`
	`b1 = Reader.ReaderEngineAPI.GetDecodeRfidUID(szUID)`

Return Value	If successful, it returns 0. Otherwise, it returns 1. ▸ Call GetErrorCode() to find the error condition encountered:

-101	E_READER_NOT_INIT
-300	E_BUFF_IS_TOO_SMALL

Remarks	This function is called to obtain the just decoded UID right before another successful decoding. The decoded data then hooks up Prefix Code and/or Suffix Code if they are applicable.
See Also	GetDecodeRfidData, GetDecodeTagType, DataOutputSettings, GetOutputRecord, RfidSettings

GetDecodeTagType

Purpose: Collects RFID tag type upon pressing a physical trigger.

Syntax: **int GetDecodeTagType ();**

Example for C#:

```
int tagType = 0;
tagType = Reader.ReaderEngineAPI.GetDecodeTagType();
```

Example for VB:

```
Dim tagType As Integer
tagType = Reader.ReaderEngineAPI.GetDecodeTagType()
```

Return Value: If successful, it returns the tag type.

Return Value	*Defined Value*	*RFID Tag / Standard*
0	0x00	Unknown
80	0x50	ISO15693
81	0x51	I-CODE SLI
82	0x52	I-CODE SLI-L
85	0x55	I-CODE SLI–S
87	0x57	55V10P
88	0x58	55V02P
89	0x59	55V10S
91	0x5B	Tag-it HF-I Plus
92	0x5C	Tag-it HF-I Pro
93	0x5D	MB89R118
94	0x5E	LRI 2K
160	0xA0	Mifare DESFire
161	0xA1	Mifare UL (Ultralight)
162	0xA2	Mifare Mini
163	0xA3	Mifare Classic
164	0xA4	ISO14443-4A
165	0xA5	Mifare S50 1K
166	0xA7	Mifare S50 4K
170	0xAA	Jewel
176	0xB0	ISO14443 B
240	0xF0	FeliCa

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-101	E_READER_NOT_INIT

Remarks: This function is called to obtain the just decoded tag type right before another successful decoding.

See Also: GetDecodeRfidData, GetDecodeRfidUID, DataOutputSettings, GetOutputRecord, RfidSettings

ReadRfidData

Purpose

Reads RFID data by tapping a GUI button (without intercepting *WM_DECODEDATA*).

Syntax

```
int ReadRfidData (int nType,
                  ref string szReadData,
                  int nStartByte,
                  int nReadLen,
                  int nTimeout,
                  string loginKey,
                  int loginKeyType);
```

Parameters

nType

[in] A value that specifies which data to read (UID or encoded data)

1	READ_UID	Reads UID only.
2	READ_DATA	Reads encoded data only.

szReadData

[out] Pointer to a buffer where decoded data is stored.

nStartByte

[in] A value that specifies the byte or page to start reading data from.

-1 Means to start reading from the default page (of a RFID tag).

Tabulated below is a number of RFID tags' default page to start reading data from:

Tag Type	Standard	Default Start Page
Mifare	(ISO 14443A)	**4**
ICODE SLI	(ISO 15693)	**3**
LRI512	(ISO 15693)	**0**
SRF55VxxP	(ISO 15693)	**3**
Tag-it	(ISO 15693)	**0**
Others	(ISO 15693)	**0**
ICODE	(Phillips)	**5**

nReadLen

[in] Number of bytes to read. As different tag types may have different numbers of pages allowed to be read, when you pass a value greater than allowed, it auto-adapts to the maximum value allowed on a tag.

nTimeout

[in][out] A value that specifies the elapsed time before a decode session ends.

1~10	3* (second)

loginKey

[in] A string variable that stores the login key, a security key that accesses an RFID tag such as Mifare Standard 1K/4K and SLE66R35 tags.

- Such key must be a hex string with 12 bytes length.
- This login key is applied to all the sectors covered by the range formulated by "nStartByte" and "nReadLen".

LoginKey Example	*Key String*
0	FFFFFFFFFFFF
"F1F2F3F4F5F6"	F1F2F3F4F5F6

loginKeyType

[in] A value that identifies the login key.

0*	Key A
1	Key B

Example for C#

```
int b1 = 0, loginKeyType = 0;
string szRead = string.Empty;
string loginKey = "FFFFFFFFFFFF";
b1 = Reader.ReaderEngineAPI.ReadRfidData(
Reader.ReaderEngineAPI.READ_DATA, ref szRead, -1, 10, 10, loginKey, loginKeyType);
```

Example for VB

```
Dim b1 As Integer
Dim szRead As String = ""
Dim loginKey As String = "FFFFFFFFFFFF"
Dim loginKeyType As Integer
b1 = Reader.ReaderEngineAPI.ReadRfidData(
Reader.ReaderEngineAPI.READ_DATA, szRead, -1, 10, 10, loginKey, loginKeyType)
```

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-101	E_READER_NOT_INIT
-272	E_TAG_READ_FAILED
-273	E_TAG_VALUE_OVER_RANGE
-274	E_TAG_INVALID_LENGTH
-278	E_TAG_NO_TAG
-280	E_TAG_CANNOT_READ
-281	E_TAG_READ_TIMEOUT

Remarks

This function scans and retrieves data without intercepting *WM_DECODEDATA*. For example the scanning triggered by tapping a GUI button. All that is need is to to invoke this function in the clicking event handler function. The prospective RFID data then can be collected after decoding.

See Also

DataOutputSettings, GetOutputRecord, RfidSettings, WriteRfidData

WriteRfidData

Purpose

Writes data to an RFID tag upon pressing a physical trigger.

Syntax

int WriteRfidData (string *szWriteData*,
int *nStartByte*,
int *nWriteLen*,
ref int *nWritten*,
int *nTimeout*,
int *nMode*,
string *loginKey*,
int *loginKeyType*);

Parameters

szWriteData

[in] A string variable that stores the data to write.

nStartByte

[in] A value that specifies the byte or page (of some tag type) to start writing data to:

-1	Means to start writing from the default page. Tabulated below is a number of RFID tags' default page to start writing data to:

Tag Type	Standard	Default Start Page
Mifare	(ISO 14443A)	**4**
ICODE SLI	(ISO 15693)	**3**
LRI512	(ISO 15693)	**0**
SRF55VxxP	(ISO 15693)	**3**
Tag-it	(ISO 15693)	**0**
Others	(ISO 15693)	**0**
ICODE	(Phillips)	**5**

nWriteLen

[in] Maximum number of bytes to write. As the number of pages allowed to be written varies from tag type to tag type, when you pass a value greater than allowed, it auto-adapts to the maximum value allowed on a tag.

nWritten

[out] Pointer to a buffer where stores the actual number of bytes written.

nTimeout

[in][out] A value that specifies the elapsed time before a decode session ends.

1~10	3* (second)

nMode

[in] A value that specifies writing mode.

0	Starts writing immediately.
Non-zero	Starts writing upon pressing a physical trigger.

loginKey

[in] A string variable that stores the login key, a key to access an RFID tag such as Mifare Standard 1K/4K and SLE66R35 tags.

- Key string must be a hex string with 12 bytes length.
- This login key is applied to all the sectors covered by the range formulated by “nStartByte” and “nWriteLen”.

LoginKey Example	*Key String*
0	FFFFFFFFFFFF
“F1F2F3F4F5F6”	F1F2F3F4F5F6

loginKeyType

[in] A value that identifies the key, key A or B.

0*	Key A
1	Key B

Example for C#

```
int b1 = 0, loginKeyType = 0;
string data = "22222222222222";
string loginKey = "FFFFFFFFFFFF";
b1 = Reader.ReaderEngineAPI.WriteRfidData(
data, -1, data.Length, 0, 10, 0, loginKey, loginKeyType);
```

Example for VB

```
Dim b1 As Integer
Dim data As String = "22222222222222"
Dim loginKey As String = "FFFFFFFFFFFF"
Dim loginKeyType As Integer
b1 = Reader.ReaderEngineAPI.WriteRfidData(
data, -1, data.Length, 0, 10, 0, loginKey, loginKeyType)
```

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-101	E_READER_NOT_INIT
-273	E_TAG_VALUE_OVER_RANGE
-274	E_TAG_INVALID_LENGTH
-275	E_TAG_GET_DATA_FAILED
-276	E_TAG_WRITE_FAILED
-277	E_TAG_CANNOT_WRITE
-278	E_TAG_NO_TAG
-279	E_TAG_WRITE_TIMEOUT

Remarks

This function defines what to write to an RFID tag and enforces writting it.

See Also

DataOutputSettings, GetOutputRecord, ReadRfidData, RfidSettings

2.3. MANIPULATE STATUS INDICATION

ReaderDLL auto-signals the receipt of a successful decoding by a sound or vibration according to **NotificationSettings()** settings. To signal the events other than decoding, call **Beeper()**.

2.3.1. NOTIFICATION SETTINGS

NotificationSettings

Purpose: Configures notification settings.

Syntax:

```
int NotificationSettings (int rw,
                          ref int goodRead,
                          ref int enableVibrator,
                          ref int vibrationTime,
                          ref int ledDuration);
```

Parameters: A default value comes with an asterisk "*".

Rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

goodRead

[in][out] A value that specifies whether to plays a .WAV file to notice successful reading.

0	Mutes
1~9	Plays sound 2*

enableVibrator

[in][out] A value that specifies whether to vibrate for a successful reading.

0*	Disables Vibrator
1	Enables Vibrator

vibrationTime

[in][out] A value that specifies how long to vibrate.

0	No vibration
1~9	2* (second)

ledDuration

[in][out] A value that specifies whether to light green LED for a successful reading.

0*	Disables
Non-zero	Enables and lighting duration (millisecond)

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

See Also Beeper

2.3.2. BEEPER (SPEAKER)

Beeper

Purpose Plays a sound via speaker.

Syntax **int Beeper (int** *nMode*,
string *szPath*);

Parameters *Mode*

[in] A value that specifies the sound to play.

0	Mutes
1~9	Sound 1 ~ Sound 9
-1	User-definable path to the sound to play

path

[in] A string variable that stores the user-defined path.

- Applicable only when **nMode** is -1.

Example for C#

```
int b1 = 0;
string path = string.Empty;
b1 = Reader.ReaderEngineAPI.Beeper(9, path);
b1 = Reader.ReaderEngineAPI.Beeper(-1, "\\window\\test.wav");
```

Example for VB

```
Dim b1 As Integer
Dim path As String = ""
b1 = Reader.ReaderEngineAPI.Beeper(9, path)
b1 = Reader.ReaderEngineAPI.Beeper(-1, "\window\test.wav")
```

Return Value If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-241	E_SOUND_INVALID_INDEX
-242	E_SOUND_INVALID_PATH
0x04	ERROR_PARAMETER

Remarks This function plays a sound or WAVE file in an asynchronous manner. To stop playing, use Beeper(0, "").

See Also NotificationSettings

2.4. OBTAIN ESSENTIAL INFORMATION

2.4.1. READER DLL VERSION

GetDllVer	
Purpose	Obtains current C/C++ library version for Reader DLL.
Syntax	**int GetDllVer (ref string** *buf***);**
Parameters	*buf* [out] Pointer to a buffer where version number is stored.

Example for C#	`int b1 = 0;`
	`string buf = string.Empty;`
	`b1 = Reader.ReaderEngineAPI.GetDllVer(ref buf);`

Example for VB	`Dim b1 As Integer`
	`Dim buf As String = ""`
	`b1 = Reader.ReaderEngineAPI.GetDllVer(buf)`

Return Value	If successful, it returns 0. Otherwise, it returns 1.

GetNETDLLVersion	
Purpose	Obtains current .NET library version for Reader DLL.
Syntax	**int GetNETDLLVersion (ref string** *dotNetVer***);**
Parameters	*dotNetVer* [out] Pointer to a buffer where version information is stored.

Example for C#	`int b1 = 0;`
	`string buf = string.Empty;`
	`b1 = Reader.ReaderEngineAPI.GetNETDLLVer(ref buf);`

Example for VB	`Dim b1 As Integer`
	`Dim buf As String = ""`
	`b1 = Reader.ReaderEngineAPI.GetNETDLLVer(buf)`

Return Value	If successful, it returns 0. Otherwise, it returns 1.

2.4.2. DECODER VERSION

GetDecoderVersion

Purpose	Gets decoder version.
Syntax	**int GetDecoderVersion (int** *target*, **ref string** *buf*);
Parameters	*target* [in] A value that specifies which reader to get decoder version for.

7	DC_READER_BC	Reads barcode reader's decoder version
32	DC_READER_RFID	Reads RFID reader's decoder version.

Buf

[out] Pointer to a buffer where version info is stored.

Example for C#

```
int b1 = 0;
string buf = string.Empty;
b1 = Reader.ReaderEngineAPI.GetDecoderVersion(DC_READER_BC, ref buf);
b1 = Reader.ReaderEngineAPI.GetDecoderVersion(
DC_READER_RFID, ref buf);
```

Example for VB

```
Dim buf As String = ""
Reader.ReaderEngineAPI.GetDecoderVersion(DC_READER_BC, buf)
Reader.ReaderEngineAPI.GetDecoderVersion(DC_READER_RFID, buf)
```

Return Value

If successful, it returns the number of bytes obtained (excluding terminating null character).

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

1	E_READER_UNKNOWN_ERROR
-101	E_READER_NOT_INIT

2.4.3. ERROR INFORMATION

GetErrorCode	
Purpose	Gets current error condition.
Syntax	**int GetErrorCode ();**

Example for C#	`int b1 = 0;`
	`int errCode = 0;`
	`b1 = Reader.ReaderEngineAPI.InitReader();`
	`if(b1 == 0)`
	`{`
	`  errCode = GetErrorCode();`
	`}`

Example for VB	`Dim errCode As Integer`
	`If Reader.ReaderEngineAPI.InitReader() = 0 Then`
	`errCode = Reader.ReaderEngineAPI.GetErrorCode()`
	`End If`

Return Value	If successful, it returns the error code.

2.5. CONFIGURE SCAN ENGINE — 1D CCD SCAN ENGINE

UserPreferences_1D_SM1() configures the 1D CCD barcode reader while others adjust symbology settings.

2.5.1. PREFERENCES

UserPreferences_1D_SM1	**CCD**

Purpose: Configures 1D CCD barcode reader.

Syntax:

```
int UserPreferences_1D_SM1 (int rw,
                            ref int laserOnTime,
                            ref int timeoutBetweenSameBarcode,
                            ref int redundancyLevel,
                            ref int triggerMode);
```

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

laserOnTime

[in][out] A value that specifies the maximum time for decoding a printed barcode during a scan act.

5~99	30* (0.1 second)

timeoutBetweenSameBarcode

[in][out] A value that specifies the minimum time for the scan engine to resurrect the ability to read the same barcode for the 2nd time. This helps escape accidently scanning the same barcode twice.

- This parameter applies to Continuous mode.

0~99	10* (0.1 second)

redundancyLevel

[in][out] A value that specifies decode redundancy. Higher redundancy levels should be selected for deteriorated barcode quality.

1*	The following barcodes must be successfully read twice before being decoded:

Barcode Types	Code Length
Codabar	All
MSI	4 characters or less
Industrial 25 (Discrete 25)	8 characters or less
Interleaved 25	8 characters or less

2	All barcodes must be successfully read twice before being decoded.
3	All barcodes must be successfully read twice before being decoded except for the following that must be read three times:

	<table><tr><th>Barcode Types “Excluded”</th><th>Code Length</th></tr><tr><td>MSI</td><td>4 characters or less</td></tr><tr><td>Industrial 25 (Discrete 25)</td><td>8 characters or less</td></tr><tr><td>Interleaved 25</td><td>8 characters or less</td></tr></table>
4	All barcodes must be successfully read three times before being decoded.

triggerMode

[in][out] A value that specifies the mode to scan.

4	Continuous mode
8*	Laser mode

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.5.2. UPC_1D_SM1

Upc_1D_SM1	**CCD**

Purpose: Configures symbology settings — UPC-A and UPC-E.

Syntax:

```
int Upc_1D_SM1 (int rw,
                ref int enableUpcA,
                ref int enableUpcE,
                ref int enableUpcE1,
                ref int enableAddons,
                ref int addonsRedundancy,
                ref int transmitUpcACheckDigit,
                ref int transmitUpcECheckDigit,
                ref int transmitUpcE1CheckDigit,
                ref int preambleUpcA,
                ref int preambleUpcE,
                ref int preambleUpcE1,
                ref int convertUpcEToA,
                ref int convertUpcE1ToA);
```

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableUpcA

[in][out] A value that specifies whether to enable UPC-A.

0	Disables
1*	Enables

enableUpcE

[in][out] A value that specifies whether to enable UPC-E (=UPC-E0).

0	Disables
1*	Enables

enableUpcE1

[in][out] A value that specifies whether to enable UPC-E1.

0*	Disables
1	Enables

enableAddons

[in][out] A value that specifies whether to enable UPC Addon 2 and Addon 5.

0*	Ignores Addons
1	Decodes only with Addons
2	Decodes with Addons (= Auto-discriminate)

addonsRedundancy

[in][out] A value that specifies decode redundancy when “Decode with Addons (= Auto-discriminate)” is applied.

2~30	7* (times of supplementary decoding)

transmitUpcACheckDigit

[in][out] A value that specifies whether to transmit check digit for UPC-A data.

0	Does not transmit
1*	Transmits

transmitUpcECheckDigit

[in][out] A value that specifies whether to transmit check digit for UPC-E0 data.

0	Does not transmit
1*	Transmits

transmitUpcE1CheckDigit

[in][out] A value that specifies whether to transmit check digit for UPC-E1 data.

0	Does not transmit
1*	Transmits

preambleUpcA

[in][out] A value that specifies whether to verify(transmit) UPC-A preamble.

0	Does not transmit
1*	Transmits System Number only
2	Transmits both System Number & Country Code

preambleUpcE

[in][out] A value that specifies whether to verify(transmit) UPC-E0 preamble.

0	Does not transmit
1*	Transmits System Number only
2	Transmits both System Number & Country Code

preambleUpcE1

[in][out] A value that specifies whether to verify(transmit) UPC-E1 preamble.

0	Does not transmit
1*	Transmits System Number only
2	Transmits both System Number & Country Code

convertUpcEToA

[in][out] A value that specifies whether to convert UPC-E0 to UPC-A.

0*	Does not convert
1	Converts

convertUpcE1ToA

[in][out] A value that specifies whether to convert UPC-E1 to UPC-A.

0*	Does not convert
1	Converts

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.5.3. EANJAN_1D_SM1

EanJan_1D_SM1	**CCD**

Purpose: Configures symbology settings — EAN/JAN-8 and EAN/JAN-13.

Syntax:

```
int EanJan_1D_SM1 (int rw,
                   ref int enableEanJan8,
                   ref int enableEanJan13,
                   ref int enableBooklandEan,
                   ref int enableAddons,
                   ref int addonsRedundancy,
                   ref int enableEanJan8Extended);
```

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableEanJan8

[in][out] A value that specifies whether to enable EAN/JAN-8.

0	Disables
1*	Enables

enableEanJan13

[in][out] A value that specifies whether to enable EAN/JAN-13.

0	Disables
1*	Enables

enableBooklandEan

[in][out] A value that specifies whether to enable Bookland EAN.

- EAN/JAN-13 must be enabled first!

0	Disables
1*	Enables

enableAddons

[in][out] A value that specifies whether to enable EAN/JAN Addon 2 and Addon 5.

0*	Ignored Addons
1	Decodes only with Addons
2	Decodes with Addons (= Auto-discriminate)

addonsRedundancy

[in][out] A value that specifies decode redundancy when "Decode with Addons (= Auto-discriminate)" is applied.

2~30	10* (times of supplementary decoding)

enableEanJan8Extended

[in][out] A value that specifies whether to convert EAN/JAN-8 to EAN/JAN-13.

0*	Does not convert
1	Converts

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.5.4. CODE39_1D_SM1

Code39_1D_SM1	**CCD**

Purpose — Configures symbology settings — Code 39.

Syntax

```
int Code39_1D_SM1 (int rw,
                   ref int enable,
                   ref int convertToCode32,
                   ref int prefixCode32,
                   ref int checkDigitVerification,
                   ref int transmitCheckDigit,
                   ref int fullASCII,
                   ref int length1,
                   ref int length2);
```

Parameters — A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Code 39.

0	Disables
1*	Enables

convertToCode32

[in][out] A value that specifies whether to convert Code 39 to Code 32 (= Italian Pharmacode).

0*	Does not convert
1	Converts

prefixCode32

[in][out] A value that specifies whether to transmit prefix for Code 32 data.

0*	Does not transmit
1	Transmits

checkDigitVerification

[in][out] A value that specifies whether to verify check digit.

0*	Does not verify
1	Verifies

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit.

- Check digit verification must be enabled first.

0*	Does not transmit
1	Transmits

fullASCII

[in][out] A value that specifies whether to support Code 39 Full ASCII.

0*	No (Standard Code 39)
1	Yes (Code 39 Full ASCII)

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

- If “length2” is greater than “length1”, they are applied as Max/Min length; otherwise a fixed length applies.

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.5.5. CODE93_1D_SM1

Code93_1D_SM1	**CCD**

Purpose

Configures symbology settings — Code 93.

Syntax

int Code93_1D_SM1 (int *rw*,
ref int *enable*,
ref int *length1*,
ref int *length2*);

Parameters

A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Code 93.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

- If "length2" is greater than "length1", they are applied as Max/Min length; otherwise a fixed length applies.

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.5.6. INTERLEAVED2OF5_1D_SM1

Interleaved2Of5_1D_SM1	**CCD**

Purpose: Configures symbology settings — Interleaved 25.

Syntax:

```
int Interleaved2Of5_1D_SM1 (int rw,
                            ref int enable,
                            ref int length1,
                            ref int length2,
                            ref int checkDigitVerification,
                            ref int transmitCheckDigit);
```

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Interleaved 25.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

- If "length2" is greater than "length1", they are applied as Max/Min length; otherwise a fixed length applies.

checkDigitVerification

[in][out] A value that specifies whether to verify check digit.

0*	Does not verify
1	Verifies

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit.

0*	Does not transmit
1	Transmits

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.5.7. INDUSTRIAL2OF5_1D_SM1

Industrial2Of5_1D_SM1	**CCD**

Purpose

Configures symbology settings — Industrial 25 (= Discrete 25).

Syntax

int Industrial2Of5_1D_SM1 (int *rw*,
ref int *enable*,
ref int *length1*,
ref int *length2*);

Parameters

A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Industrial 25 (= Discrete 25).

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

- If "length2" is greater than "length1", they are applied as Max/Min length; otherwise a fixed length applies.

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.5.8. CODABAR_1D_SM1

Codabar_1D_SM1	**CCD**

Purpose: Configures symbology settings — Codabar.

Syntax:

```
int Codabar_1D_SM1 (int rw,
                    ref int enable,
                    ref int length1,
                    ref int length2,
                    ref int enableCLSIEditing,
                    ref int enableNOTISEditing);
```

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Codabar.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification. (see above)

0~55	55*

- If "length2" is greater than "length1", they are applied as Max/Min length; otherwise a fixed length applies.

enableCLSIEditing

[in][out] A value that specifies whether to edit CLSI.

0*	Does not edit
1	Edits

- When applied, CLSI editing strips the start/stop characters and inserts a space after the first, fifth, and tenth characters of a 14-character Codabar barcode.
- The 14-character barcode length does not include start/stop characters.

enableNOTISEditing

[in][out] A value that specifies whether to edit NOTIS when decoding barcodes.

0*	Disables
1	Enables

- NOTIS Editing strips the start/stop characters. It is equally efficacious as "Disable Transmit Start/Stop Characters".

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.5.9. MSI_1D_SM1

Msi_1D_SM1	**CCD**

Purpose: Configures symbology settings — MSI.

Syntax:

```
int Msi_1D_SM1 (int rw,
                ref int enable,
                ref int length1,
                ref int length2,
                ref int checkDigitVerification,
                ref int transmitCheckDigit,
                ref int checkDigitAlgorithm);
```

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable MSI.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

- If "length2" is greater than "length1", they are applied as Max/Min length; otherwise a fixed length applies.

checkDigitVerification

[in][out] A value that specifies how to verify check digit.

0*	Verifies one check digit
1	Verifies two check digits

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit.

0*	Does not transmit
1	Transmits

checkDigitAlgorithm

[in][out] A value that specifies which algorithm to apply.

0	Modulo 10 / Modulo 11
1*	Double Modulo 10

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.5.10. GS1_DATABAR_1D_SM1

GS1_DataBar_1D_SM1	**CCD**

Purpose — Configures symbology settings — GS1 DataBar (= RSS family).

Syntax

```
int GS1_DataBar_1D_SM1 (int rw,
                        ref int enableDataBarOmnidirectional,
                        ref int enableGS1_DataBarLimited,
                        ref int enableGS1_DataBarExpanded);
```

Parameters — A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableDataBarOmnidirectional

[in][out] A value that specifies whether to enable GS1 DataBar Omnidirectional (= RSS-14).

0	Disables
1*	Enables

enableGS1_DataBarLimited

[in][out] A value that specifies whether to enable GS1 DataBar Limited (= RSS Limited).

0	Disables
1*	Enables

enableGS1_DataBarExpanded

[in][out] A value that specifies whether to enable GS1 DataBar Expanded (= RSS Expanded).

0	Disables
1*	Enables

Return Value — If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.5.11. CODE128_1D_SM1

Code128_1D_SM1	**CCD**

Purpose — Configures symbology settings — Code 128.

Syntax

```
int Code128_1D_SM1 (int rw,
                    ref int enableCode128,
                    ref int enableGS1_128,
                    ref int enableISBT128);
```

Parameters — A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableCode128

[in][out] A value that specifies whether to enable Code 128.

0	Disables
1*	Enables

enableGS1_128

[in][out] A value that specifies whether to enable GS1-128.

0	Disables
1*	Enables

enableISBT128

[in][out] A value that specifies whether to enable ISBT 128.

0	Disables
1*	Enables

Return Value — If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6. CONFIGURE SCAN ENGINE — 1D LASER SCAN ENGINE

UserPreferences_1D_SE955() and **MiscellaneousOption_1D_SE955()** are the functions to configure 1D (laser) barcode reader while others adjust symbology settings.

2.6.1. PREFERENCES

UserPreferences_1D_SE955	**Laser**

Purpose: Configures 1D (laser) barcode reader.

Syntax:

int UserPreferences_1D_SE955 (int *rw*,
ref int *laserOnTime*,
ref int *scanAngle*,
ref int *timeoutBetweenSameBarcode*,
ref int *redundancyLevel*,
ref int *bidirectionalRedundancy*,
ref int *aimDuration*,
ref int *triggerMode*);

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

laserOnTime

[in][out] A value that specifies the maximum time for decoding a printed barcode during a scan act.

5~99	30* (0.1 second)

scanAngle

[in][out] A value that specifies scan angle.

0x05	Narrow Angle (35°)
0x06*	Wide Angle (47°)

timeoutBetweenSameBarcode

[in][out] A value that specifies the minimum time for the scan engine to resurrect the ability to read the same barcode for the 2nd time. This helps escape accidently scanning the same barcode twice.

- This parameter applies to Continuous mode.

0~99	10* (0.1 second)

redundancyLevel

[in][out] A value that specifies decode redundancy. Higher redundancy levels should be selected for deteriorated barcode quality.

1*	The following barcodes must be successfully read twice before being decoded:

Barcode Types	Code Length

	<table><tr><td>Codabar</td><td>All</td></tr><tr><td>MSI</td><td>4 characters or less</td></tr><tr><td>Industrial 25 (Discrete 25)</td><td>8 characters or less</td></tr><tr><td>Interleaved 25</td><td>8 characters or less</td></tr></table>
2	All barcodes must be successfully read twice before being decoded.
3	All barcodes must be successfully read twice before being decoded except for the following that must be read three times: <table><tr><th>Barcode Types “Excluded”</th><th>Code Length</th></tr><tr><td>MSI</td><td>4 characters or less</td></tr><tr><td>Industrial 25 (Discrete 25)</td><td>8 characters or less</td></tr><tr><td>Interleaved 25</td><td>8 characters or less</td></tr></table>
4	All barcodes must be successfully read three times before being decoded.

bidirectionalRedundancy

[in][out] A value that specifies whether to enable bidirectional redundancy. When enabled, barcodes must be scanned in both forward and reverse directions in order to be decoded.

- This is valid only when redundancy level is enabled for the specific barcode. For instance, when *redundancyLevel* is set at Level 1, only the barcodes that have redundancy enabled will require both forward and reverse direction scanning. When set at Level 2, all barcodes will require both forward and reverse direction scanning.

0*	Disable
1	Enable

aimDuration Reserved

triggerMode

[in][out] A value that specifies the mode to scan.

4	Continuous mode
8*	Laser mode

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

MiscellaneousOption_1D_SE955	**Laser**

Purpose: Configures more reader settings.

Syntax:

int MiscellaneousOption _1D_SE955 (int *rw*,
ref int *transmitCodeIdChar*,
ref int *sendNoReadMessage*);

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

transmitCodeIdChar

[in][out] A value that specifies whether to transmit Code ID characters.

0*	Transmits none
1	Transmits AIM Code ID Character Being included in the beginning of data, each AIM Code ID contains the three-character string "**]cm**" – ▸] = Flag Character (ASCII 93) ▸ c = Code Character (see below) ▸ m = Modifier Character (see below)

sendNoReadMessage Reserved

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

AIM Code ID — Code Characters

Code Character	Code Type
A	Code 39
C	Code 128
E	UPC/EAN
F	Codabar
G	Code 93
H	Code 11
I	Interleaved 25
M	MSI
S	Industrial 25 (Discrete 25), IATA 2 of 5
X	Code 39 Trioptic, Bookland EAN

AIM Code ID — Modifier Characters

Code Type	Option Value	Option
Code 39	0	No check character or Full ASCII processing.
	1	Check digit has been verified.
	3	Check digit has been verified and stripped.
	4	Full ASCII conversion has been performed.
	5	Result of option values 1 and 4.
	7	Result of option values 3 and 4.
Code 128	0	Standard data packet. No Function Code 1“FNC1” in the first character position.
	1	Function Code 1“FNC1” in the first character position.
	2	Function Code 1“FNC1” in the second character position.
Interleaved 25	0	No check digit processing.
	1	Check digit has been verified.
	3	Check digit has been verified and stripped.
Codabar	0	No check digit processing.
Code 93	0	Always transmit 0.
MSI	0	Modulo 10 check digit verified and transmitted.
	1	Modulo 10 check digit verified but not transmitted.
Industrial 25 (Discrete 25)	0	Always transmit 0.
UPC/EAN	0	Standard data packet in full EAN country code format, which is 13 digits for UPC-A and UPC-E (not including supplemental data).
	1	Two-digit supplemental data only.
	2	Five-digit supplemental data only.
	4	EAN-8 data packet.
	A UPC-A with Addon 2 barcode, 012345678905-10, is transmitted to the host as a 21-character string, **]E0**0012345678905**]E1**10.	
Bookland EAN	0	Always transmit 0.
Trioptic Code 39	0	Always transmit 0.

2.6.2. UPC_1D_SE955

Upc_1D_SE955	**Laser**

Purpose: Configures symbology settings — UPC-A and UPC-E.

Syntax:

int Upc_1D_SE955 (int *rw*,
ref int *enableUpcA*,
ref int *enableUpcE*,
ref int *enableUpcE1*,
ref int *enableAddons*,
ref int *addonsRedundancy*,
ref int *transmitUpcACheckDigit*,
ref int *transmitUpcECheckDigit*,
ref int *transmitUpcE1CheckDigit*,
ref int *preambleUpcA*,
ref int *preambleUpcE*,
ref int *preambleUpcE1*,
ref int *convertUpcEtoA*,
ref int *convertUpcE1toA*,
ref int *uccCouponExtendedCode*,
ref int *upcEanSecurityLevel*);

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableUpcA

[in][out] A value that specifies whether to enable UPC-A.

0	Disables
1*	Enables

enableUpcE

[in][out] A value that specifies whether to enable UPC-E (=UPC-E0).

0	Disables
1*	Enables

enableUpcE1

[in][out] A value that specifies whether to enable UPC-E1.

0*	Disables
1	Enables

enableAddons

[in][out] A value that specifies whether to enable UPC Addon 2 and Addon 5.

0*	Ignores Addons
1	Decodes only with Addons

2	Decodes with Addons (= Auto-discriminate)

addonsRedundancy

[in][out] A value that specifies decode redundancy when “Decode with Addons (= Auto-discriminate)” is applied.

2~30	7* (times of supplementary decoding)

transmitUpcACheckDigit

[in][out] A value that specifies whether to transmit check digit for UPC-A data.

0	Does not transmit
1*	Transmits

transmitUpcECheckDigit

[in][out] A value that specifies whether to transmit check digit for UPC-E0 data.

0	Does not transmit
1*	Transmits

transmitUpcE1CheckDigit

[in][out] A value that specifies whether to transmit check digit for UPC-E1 data.

0	Does not transmit
1*	Transmits

preambleUpcA

[in][out] A value that specifies whether to verify(transmit) UPC-A preamble.

0	Does not transmit
1*	Transmits System Number only
2	Transmits both System Number & Country Code

preambleUpcE

[in][out] A value that specifies whether to verify(transmit) UPC-E0 preamble.

0	Does not transmit
1*	Transmits System Number only
2	Transmits both System Number & Country Code

preambleUpcE1

[in][out] A value that specifies whether to verify(transmit) UPC-E1 preamble.

0	Does not transmit
1*	Transmits System Number only
2	Transmits both System Number & Country Code

convertUpcEtoA

[in][out] A value that specifies whether to convert UPC-E0 to UPC-A.

0*	Does not convert
1	Converts

convertUpcE1toA

[in][out] A value that specifies whether to convert UPC-E1 to UPC-A.

0*	Does not convert

1	Converts

uccCouponExtendedCode

[in][out] A value that specifies whether to enable UCC Coupon Code: UPC-A barcodes starting with digit “5” and UPC-A Coupon Codes.

0*	Disables
1	Enables

upcEanSecurityLevel

[in][out] A value that specifies the decode security for UPC/EAN barcodes. Higher security levels should be selected for deteriorated barcode quality. Note that increasing security level decreases the scan engine’s aggressiveness; choose only the security level needed for the application.

0	Level 0 provides sufficient security to decode most “in-spec” UPC/EAN barcodes.
1	Level 1: As barcode quality deteriorates, certain characters become liable to misdecode (i.e. 1, 2, 7, 8). If misdecodes occur due to poor printed quality, and misdecoded character are limited to the said, select this security level.
2*	Level 2: If misdecodes occur due to poor printed quality, and misdecoded characters are not limited to 1, 2, 7 and 8, select this security level.
3	Level 3: If Security Level 2 should fail to prevent misdecodes, select this security level. Note this level is a final act that tackles out-of-spec barcodes by impairing the decoding ability of scan engine. It is therefore recommended to improve barcode’s print quality if this level should be needed.

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6.3. EANJAN_1D_SE955

EanJan_1D_SE955	**Laser**

Purpose — Configures symbology settings — EAN/JAN-8 and EAN/JAN-13.

Syntax

```
int EanJan_1D_SE955 (int rw,
                     ref int enableEanJan8,
                     ref int enableEanJan13,
                     ref int enableBooklandEan,
                     ref int enableAddons,
                     ref int addonsRedundancy,
                     ref int enableEanJan8Extended,
                     ref int uccCouponExtendedCode,
                     ref int upcEanSecurityLevel);
```

Parameters — A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableEanJan8

[in][out] A value that specifies whether to enable EAN/JAN-8.

0	Disables
1*	Enables

enableEanJan13

[in][out] A value that specifies whether to enable EAN/JAN-13.

0	Disables
1*	Enables

enableBooklandEan

[in][out] A value that specifies whether to enable Bookland EAN.

- EAN/JAN-13 must be enabled first!

0	Disables
1*	Enables

enableAddons

[in][out] A value that specifies whether to enable EAN/JAN Addon 2 and Addon 5.

0*	Ignored Addons
1	Decodes only with Addons
2	Decodes with Addons (= Auto-discriminate)

addonsRedundancy

[in][out] A value that specifies decode redundancy when "Decode with Addons (= Auto-discriminate)" is applied.

2~30	10* (times of supplementary decoding)

enableEanJan8Extended

[in][out] A value that specifies whether to convert EAN/JAN-8 to EAN/JAN-13.

0*	Does not convert
1	Converts

uccCouponExtendedCode

[in][out] A value that specifies whether to enable UCC Coupon Code: EAN-13 barcodes starting with digits “99”.

0*	Disables
1	Enables

upcEanSecurityLevel

[in][out] A value that specifies the decode security for UPC/EAN barcodes. Higher redundancy levels should be selected for deteriorated barcode quality. Note that increased security level decreases the scan engine’s aggressiveness; choose only the security level necessary for the application.

0	Level 0 provides sufficient security to decode most “in-spec” UPC/EAN barcodes.
1	Level 1: As barcode quality deteriorates, certain characters become liable to misdecodes (i.e. 1, 2, 7, 8). If misdecodes occur due to poor printed quality, and the misdecoded characters are limited to said characters, select this security level.
2*	Level 2: If misdecodes occur due to poor printed quality, and the misdecoded characters are not limited to 1, 2, 7 and 8, select this security level.
3	Level 3: If Security Level 2 should fail to prevent misdecodes, select this security level. Note this level is a final act that tackles out-of-spec barcodes by impairing the decoding ability of scan engine. It is therefore recommended to improve barcode’s print quality if this level should be needed.

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6.4. CODE39_1D_SE955

Code39_1D_SE955	**Laser**

Purpose — Configures symbology settings — Code 39.

Syntax

```
int Code39_1D_SE955 (int rw,
                     ref int enable,
                     ref int enableTrioptic,
                     ref int convertToCode32,
                     ref int prefixCode32,
                     ref int checkDigitVerification,
                     ref int transmitCheckDigit,
                     ref int fullASCII,
                     ref int length1,
                     ref int length2);
```

Parameters — A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Code 39.

0	Disables
1*	Enables

enableTrioptic

[in][out] A value that specifies whether to enable Trioptic Code 39.

0*	Disables
1	Enables

convertToCode32

[in][out] A value that specifies whether to convert Code 39 to Code 32 (= Italian Pharmacode).

0*	Does not convert
1	Converts

prefixCode32

[in][out] A value that specifies whether to transmit prefix for Code 32 data.

0*	Does not transmit
1	Transmits

checkDigitVerification

[in][out] A value that specifies whether to verify check digit.

0*	Does not verify
1	Verifies

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit.

- Check digit verification must be enabled first.

0*	Does not transmit
1	Transmits

fullASCII

[in][out] A value that specifies whether to support Code 39 Full ASCII.

0*	No (Standard Code 39)
1	Yes (Code 39 Full ASCII)

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

- If “length2” is greater than “length1”, they are applied as Max/Min length; otherwise a fixed length applies.

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6.5. CODE93_1D_SE955

Code93_1D_SE955	**Laser**

Purpose: Configures symbology settings — Code 93.

Syntax:

```
int Code93_1D_SE955 (int rw,
                     ref int enable,
                     ref int length1,
                     ref int length2);
```

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Code 93.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

- If "length2" is greater than "length1", they are applied as Max/Min length; otherwise a fixed length applies.

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6.6. INTERLEAVED2OF5_1D_SE955

Interleaved2Of5_1D_SE955	**Laser**

Purpose: Configures symbology settings — Interleaved 25.

Syntax:

```
int Interleaved2Of5_1D_SE955 (int rw,
                              ref int enable,
                              ref int length1,
                              ref int length2,
                              ref int checkDigitVerification,
                              ref int transmitCheckDigit,
                              ref int convertToEAN13);
```

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Interleaved 25.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

- If "length2" is greater than "length1", they are applied as Max/Min length; otherwise a fixed length applies.

checkDigitVerification

[in][out] A value that specifies whether to verify check digit.

0*	Does not verify
1	Verifies

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit.

0*	Does not transmit
1	Transmits

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6.7. INDUSTRIAL2OF5_1D_SE955

Industrial2Of5_1D_SE955	**Laser**

Purpose: Configures symbology settings — Industrial 25 (= Discrete 25).

Syntax:

```
int Industrial2Of5_1D_SE955 (int rw,
                             ref int enable,
                             ref int length1,
                             ref int length2);
```

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Industrial 25 (= Discrete 25).

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

- If "length2" is greater than "length1", they are applied as Max/Min length; otherwise a fixed length applies.

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6.8. CHINESE2OF5_1D_SE955

Chinese2Of5_1D_SE955	**Laser**

Purpose: Configures symbology settings — Chinese 25.

Syntax:

int Chinese2Of5_1D_SE955 (int *rw*,
ref int *enable*);

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Chinese 25.

0	Disables
1*	Enables

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6.9. CODABAR_1D_SE955

Codabar_1D_SE955	**Laser**

Purpose — Configures symbology settings — Codabar.

Syntax

```
int Codabar_1D_SE955 (int rw,
                      ref int enable,
                      ref int length1,
                      ref int length2,
                      ref int enableCLSI_Editing,
                      ref int enableNOTIS_Editing);
```

Parameters — A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Codabar.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification. (see above)

0~55	55*

- If "length2" is greater than "length1", they are applied as Max/Min length; otherwise a fixed length applies.

enableCLSI_Editing

[in][out] A value that specifies whether to edit CLSI.

0*	Does not edit
1	Edits

- When applied, CLSI editing strips the start/stop characters and inserts a space after the first, fifth, and tenth characters of a 14-character Codabar barcode.
- The 14-character barcode length does not include start/stop characters.

enableNOTIS_Editing

[in][out] A value that specifies whether to edit NOTIS when decoding barcodes.

0*	Disables
1	Enables

- NOTIS Editing strips the start/stop characters. It is equally efficacious as "Disable Transmit Start/Stop Characters".

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6.10. MSI_1D_SE955

Msi_1D_SE955	**Laser**

Purpose: Configures symbology settings — MSI.

Syntax:

```
int Msi_1D_SE955 (int rw,
                  ref int enable,
                  ref int length1,
                  ref int length2,
                  ref int checkDigitVerification,
                  ref int transmitCheckDigit,
                  ref int checkDigitAlgorithm);
```

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable MSI.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

- If "length2" is greater than "length1", they are applied as Max/Min length; otherwise a fixed length applies.

checkDigitVerification

[in][out] A value that specifies how to verify check digit.

0*	Verifies one check digit
1	Verifies two check digits

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit.

0*	Does not transmit
1	Transmits

checkDigitAlgorithm

[in][out] A value that specifies which algorithm to apply.

0	Modulo 10 / Modulo 11
1*	Double Modulo 10

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6.11. GS1_DATABAR_1D_SE955

GS1_DataBar_1D_SE955	**Laser**

Purpose — Configures symbology settings — GS1 DataBar (= RSS family).

Syntax

```
int GS1_DataBar_1D_SE955 (int rw,
                          ref int enableGS1_DataBarOmnidirectional,
                          ref int enableGS1_DataBarLimited,
                          ref int enableGS1_DataBarExpanded,
                          ref int convertToUpcEan);
```

Parameters — A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableGS1_DataBarOmnidirectional

[in][out] A value that specifies whether to enable GS1 DataBar Omnidirectional (= RSS-14).

0	Disables
1*	Enables

enableGS1_DataBarLimited

[in][out] A value that specifies whether to enable GS1 DataBar Limited (= RSS Limited).

0	Disables
1*	Enables

enableGS1_DataBarExpanded

[in][out] A value that specifies whether to enable GS1 DataBar Expanded (= RSS Expanded).

0	Disables
1*	Enables

convertToUpcEan

[in][out] A value that specifies whether to convert RSS to UPC/EAN barcodes.

0*	Does not convert
1	Convert

Return Value — If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6.12. CODE128_1D_SE955

Code128_1D_SE955	**Laser**

Purpose — Configures symbology settings — Code 128.

Syntax

```
int Code128_1D_SE955 (int rw,
                      ref int enableCode128,
                      ref int enableGS1_128,
                      ref int enableISBT128);
```

Parameters — A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableCode128

[in][out] A value that specifies whether to enable Code 128.

0	Disables
1*	Enables

enableGS1_128

[in][out] A value that specifies whether to enable GS1-128.

0	Disables
1*	Enables

enableISBT128

[in][out] A value that specifies whether to enable ISBT 128.

0	Disables
1*	Enables

Return Value — If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.6.13. CODE11_1D_SE955

Code11_1D_SE955	**Laser**

Purpose

Configures symbology settings — Code 11.

Syntax

```
int Code11_1D_SE955 (int rw,
                     ref int enable,
                     ref int numberOfCheckDigits,
                     ref int transmitCheckDigit,
                     ref int length1,
                     ref int length2);
```

Parameters

A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Code 11.

0	Disables
1*	Enables

numberOfCheckDigits

[in][out] A value that specifies whether and how to verify check digit.

0*	None
1	One check digit
2	Two check digits

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit.

0*	Does not transmit
1	Transmits

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

- If "length2" is greater than "length1", they are applied as Max/Min length; otherwise a fixed length applies.

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-253	E_WRONG_READER_TYPE

2.7. CONFIGURE SCAN ENGINE – 2D SCAN ENGINE

UserPreferences_2D_4500() and **MiscellaneousOption_2D_4500** are the functiosn to modulate reader module(s) while others adjust symbology settings.

2.7.1. PREFERENCES

UserPreferences_2D_SE4500	**2D**

Purpose: Configures reader settings.

Syntax:

int UserPreferences_2D_SE4500 (int *rw*,
ref int *laserOnTime*,
ref int *timeoutBetweenSameBarcode*,
ref int *triggerMode*);

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

laserOnTime

[in][out] A value that specifies the maximum time to decode a printed barcode during a scan act.

5~99	30* (0.1 second)

timeoutBetweenSameBarcode Reserved

triggerMode

[in][out] A value that specifies the mode to scan.

0*	Level (default)
7	Presentation mode
9	Auto aim

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

UserPreferences_Extend	**2D**

Purpose

Sets LED illumination level.

Syntax

INT UserPreferences_Extend (INT *rw*,
UserPreferencesParameters **param*);

Parameters

rw

[in] A value that specifies the operation.

'r'	Reader.ReaderEngineAPI.READ_PARAM	Gets settings
'w'	Reader.ReaderEngineAPI.WRITE_PARAM	Sets settings

param

[in] Pointer to the structure UserPreferencesParameters where the information is stored.

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

4	ERROR_PARAMETER

SymbologySecurityLevel_2D_SE4500	**2D**

Purpose: Sets decode redundancy.

Syntax:

int SymbologySecurityLevel _2D_SE4500 (int *rw*,
ref int *redundancyLevel*,
ref int *securityLevel*,
ref int *interCharGapSize*);

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

redundancyLevel

[in][out] A value that specifies decode redundancy. Higher redundancy levels should be selected for deteriorated barcode quality.

1*	The following barcodes must be successfully read twice before being decoded:

Barcode Types	Code Length
Codabar	8 characters or less
MSI	4 characters or less
Industrial 25 (Discrete 25)	8 characters or less
Interleaved 25	8 characters or less

2	All barcodes must be successfully read twice before being decoded.
3	All barcodes must be successfully read twice before being decoded except for the following that must be read three times:

Barcode Types "Excluded"	Code Length
Codabar	8 characters or less
MSI	4 characters or less
Industrial 25 (Discrete 25)	8 characters or less
Interleaved 25	8 characters or less

4	All barcodes must be successfully read three times before being decoded.

securityLevel

[in][out] A value that specifies decode security level, which is useful to fix some printed quality issues when reading delta barcodes such as Code 128, Code 93, UPC/EAN.

0*	Security Level 0 – The default. It allows the scan engine to operate aggressively enough to decode most “in-spec” barcodes.
1	Security Level 1 – Select this option if misdecodes occur. This level should fix most misdecodes.
2	Security Level 2 – Select this option if Security Level 1 fails to fix misdecodes.
3	Security Level 3 – Select this option if Security Level 2 also fails to fix misdecodes. However, selecting this option impairs the decoding ability of the scan engine. If this level of security is necessary, try to improve the barcode quality.

interCharGapSize

[in][out] A value that specifies the intercharacter gap size for Code 39 and Codabar, which is typically quite small. Due to various barcode printing technologies, this gap can grow larger than the maximum size allowed and prevent the scan engine from decoding a barcode. If this problem occurs, set it to “Large Intercharacter Gaps” to tolerate these out-of-specification barcodes.

0x06*	Normal intercharacter gaps
0x0A	Large intercharacter gaps

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

MiscellaneousOption_2D_SE4500	**2D**

Purpose — Configures more reader settings.

Syntax

int MiscellaneousOption _2D_SE4500 (int *rw*,
ref int *transmitCodeIdChar*,
ref int *sendNoReadMessage*);

Parameters — The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

transmitCodeIdChar

[in][out] A value that specifies whether to transmit Code ID characters.

0*	None
1	AIM Code ID Character Being included in the beginning of data, each AIM Code ID contains the three-character string "**]cm**" – ▸] = Flag Character (ASCII 93) ▸ c = Code Character (see below) ▸ m = Modifier Character (see below)

sendNoReadMessage Reserved

Return Value — If successful, it returns 0.

Otherwise, it returns 1.

▸ Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

AIM Code ID — Code Characters

Code Character	Code Type
A	Code 39, Code 39 Full ASCII, Code 32
C	Code 128, Coupon (Code 128 portion)
d	Data Matrix
E	UPC/EAN, Coupon (UPC portion)
e	GS1 DataBar (RSS)
F	Codabar
G	Code 93
H	Code 11
I	Interleaved 25
L	PDF417, Macro PDF417, Micro PDF417
M	MSI
Q	QR Code

S	Industrial 25 (Discrete 25), IATA 2 of 5
U	Maxicode
X	Code 39 Trioptic, Bookland EAN, US Postnet, US Planet, UK Postal, Japan Postal, Australian Postal, Dutch Postal

AIM Code ID — Modifier Characters

Code Type	Option Value	Option
Code 39	0	No check character or Full ASCII processing.
	1	Check digit has been verified.
	3	Check digit has been verified and stripped.
	4	Full ASCII conversion has been performed.
	5	Result of option values 1 and 4.
	7	Result of option values 3 and 4.
Code 128	0	Standard data packet. No Function Code 1“FNC1” in the first character position.
	1	Function Code 1“FNC1” in the first character position.
	2	Function Code 1“FNC1” in the second character position.
Interleaved 25	0	No check digit processing.
	1	Check digit has been verified.
	3	Check digit has been verified and stripped.
Codabar	0	No check digit processing.
Code 93	0	Always transmit 0.
MSI	0	Modulo 10 check digit verified and transmitted.
	1	Modulo 10 check digit verified but not transmitted.
Industrial 25 (Discrete 25)	0	Always transmit 0.
UPC/EAN	0	Standard data packet in full EAN country code format, which is 13 digits for UPC-A and UPC-E (not including supplemental data).
	3	Standard data packet with two-digit or five-digit supplemental data.
	4	EAN-8 data packet.
	A UPC-A with Addon 2 barcode, 012345678905-10, is transmitted to the host as a 18-character string, **]E3**001234567890510.	
Bookland EAN	0	Always transmit 0.
Trioptic Code 39	0	Always transmit 0.
Code 11	0	Single check digit (has been verified.)
	1	Two check digits (has been verified.)

	3	Check digit has been verified but not transmitted.
GS1 DataBar (RSS)	0	Always transmit 0.
	RSS-14 and RSS Limited will be transmitted with an Application Identifier "01". For example, an RSS-14 barcode, 10012345678902, is transmitted as **]e0**0110012345678902.	
EAN.UCC Composites (RSS, GS1-128, 2D portion of UPC composite)	Native mode transmission	
	0	Standard data packet
	1	Data packet containing the data following an encoded symbol separator character.
	2	Data packet containing the data following an escape mechanism character. The data packet does not support the ECI protocol.
	3	Data packet containing the data following an escape mechanism character. The data packet supports the ECI protocol.
	GS1-128 emulation	
	1	Data packet is a GS1-128 barcode (i.e. data is preceded with "]JC1").
PDF417, Micro PDF417	0	Scan engine is set to conform to protocol defined in 1994 PDF417 symbology specifications.
	1	Scan engine is set to follow the ECI protocol (Extended Channel Interpretation). All data characters 92_{DEC} are doubled.
	2	Scan engine is set for Basic Channel operation (no escape character transmission protocol). Data characters 92_{DEC} are not doubled.
	3	The barcode contains a GS1-128 symbol, and the first codeword is 903-907, 912, 914, 915.
	4	The barcode contains a GS1-128 symbol, and the first codeword is in the range 908-909.
	5	The barcode contains a GS1-128 symbol, and the first codeword is in the range 910-911.
	A PDF417 barcode, ABCD, with no transmission protocol enabled, is transmitted as]L2ABCD.	
Data Matrix	0	ECC 000-140, not supported.
	1	ECC 200.
	2	ECC 200, FNC1 in first or fifth position.
	3	ECC 200, FNC1 in second or sixth position.
	4	ECC 200, ECI protocol implemented.
	5	ECC 200, FNC1 in first or fifth position, ECI protocol implemented.
	6	ECC 200, FNC1 in second or sixth position, ECI protocol implemented.
Maxicode	0	Mode 4 or 5

	1	Mode 2 or 3
	2	Mode 4 or 5, ECI protocol implemented.
	3	Mode 2 or 3, ECI protocol implemented in secondary message.
QR Code	0	Model 1
	1	Model 2, ECI protocol not implemented.
	2	Model 2, ECI protocol implemented.
	3	Model 2, ECI protocol not implemented, FNC1 implied in first position.
	4	Model 2, ECI protocol implemented, FNC1 implied in first position.
	5	Model 2, ECI protocol not implemented, FNC1 implied in second position.
	6	Model 2, ECI protocol implemented, FNC1 implied in second position

2.7.2. UPC_2D_4500

Upc_2D_SE4500	**2D**

Purpose: Configures symbology settings — UPC-A and UPC-E.

Syntax:

```
int Upc_2D_SE4500 (int rw,
                   ref int enableUpcA,
                   ref int enableUpcE,
                   ref int enableUpcE1,
                   ref int enableAddons,
                   ref int addonsRedundancy,
                   ref int transmitUpcACheckDigit,
                   ref int transmitUpcECheckDigit,
                   ref int transmitUpcE1CheckDigit,
                   ref int preambleUpcA,
                   ref int preambleUpcE,
                   ref int preambleUpcE1,
                   ref int convertUpcEtoA,
                   ref int convertUpcE1toA,
                   ref int uccCouponExtendedCode);
```

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableUpcA

[in][out] A value that specifies whether to enable UPC-A.

0	Disables
1*	Enables

enableUpcE

[in][out] A value that specifies whether to enable UPC-E (=UPC-E0).

0	Disables
1*	Enables

enableUpcE1

[in][out] A value that specifies whether to enable UPC-E1.

0*	Disables
1	Enables

enableAddons

[in][out] A value that specifies whether to enable UPC/EAN/JAN Addon 2 and Addon 5.

0*	Ignores Addons
1	Decodes only with Addons

2	Decodes with Addons (= Auto-discriminate)

addonsRedundancy

[in][out] A value that specifies decode redundancy when “Decode with Addons (= Auto-discriminate)” is applied.

2~30	10* (times of supplementary decoding)

transmitUpcACheckDigit

[in][out] A value that specifies whether to transmit check digit for UPC-A data.

0	Does not transmit
1*	Transmits

transmitUpcECheckDigit

[in][out] A value that specifies whether to transmit check digit for UPC-E0 data.

0	Does not transmit
1*	Transmits

transmitUpcE1CheckDigit

[in][out] A value that specifies whether to transmit check digit for UPC-E1 data.

0	Does not transmit
1*	Transmits

preambleUpcA

[in][out] A value that specifies whether and how to transmit UPC-A preamble.

0	Does not transmit
1*	Transmits System Number
2	Transmits System Number & Country Code

preambleUpcE

[in][out] A value that specifies whether and how to transmit UPC-E0 preamble.

0	Does not transmit
1*	Transmits System Number
2	Transmits System Number & Country Code

preambleUpcE1

[in][out] A value that specifies whether and how to transmit UPC-E1 preamble.

0	Does not transmit
1*	Transmits System Number
2	Transmits System Number & Country Code

convertUpcEtoA

[in][out] A value that specifies whether to convert UPC-E0 to UPC-A.

0*	Does not convert
1	Converts

convertUpcE1toA

[in][out] A value that specifies whether to convert UPC-E1 to UPC-A.

0*	Does not convert

1	Converts

uccCouponExtendedCode

[in][out] A value that specifies whether to enable UCC Coupon Code: UPC-A barcodes starting with digit “5” and UPC-A Coupon Codes.

0*	Disables
1	Enables

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.3. EANJAN_2D_4500

EanJan_2D_SE4500	**2D**

Purpose: Configures symbology settings — EAN/JAN-8 and EAN/JAN-13.

Syntax:

int EanJan_2D_SE4500 (int *rw*,
ref int *enableEanJan8*,
ref int *enableEanJan13*,
ref int *enableBooklandEan*,
ref int *enableAddons*,
ref int *addonsRedundancy*,
ref int *enableEanJan8Extended*,
ref int *uccCouponExtendedCode*,
ref int *booklandIsbnFormat*,
ref int *EnableIssnEan*);

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableEanJan8

[in][out] A value that specifies whether to enable EAN/JAN-8.

0	Disables
1*	Enables

enableEanJan13

[in][out] A value that specifies whether to enable EAN/JAN-13.

0	Disables
1*	Enables

enableBooklandEan

[in][out] A value that specifies whether to enable Bookland EAN.

0	Disables
1*	Enables

enableAddons

[in][out] A value that specifies whether to enable EAN/JAN Addon 2 and Addon 5.

0*	Ignores Addons
1	Decodes only with Addons
2	Decodes with Addons (= Auto-discriminate)

addonsRedundancy

[in][out] A value that specifies decode redundancy when "Decode with Addons (= Auto-discriminate)" is applied.

2~30	10* (times of supplementary decoding)

enableEanJan8Extended

[in][out] A value that specifies whether to convert EAN/JAN-8 to EAN/JAN-13.

0*	Does not convert
1	Converts

uccCouponExtendedCode

[in][out] A value that specifies whether to enable UCC Coupon Code: EAN-13 barcodes starting with digits “99”.

0*	Disables
1	Enables

booklandIsbnFormat

[in][Out] If you enabled Bookland EAN, select one of the following formats for Bookland data.

0*	Bookland ISBN-10
1	Bookland ISBN-13

EnableIssnEan

[in][Out] A value that specifies the whether to enable Issn Ean.

0*	Disables
1	Enables

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.4. CODE39_2D_4500

Code39_2D_SE4500	**2D**

Purpose — Configures symbology settings — Code 39.

Syntax

```
int Code39_2D_SE4500 (int rw,
                      ref int enable,
                      ref int enableTrioptic,
                      ref int convertToCode32,
                      ref int prefixCode32,
                      ref int checkDigitVerification,
                      ref int transmitCheckDigit,
                      ref int fullASCII,
                      ref int length1,
                      ref int length2);
```

Parameters — The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Code 39.

0	Disables
1*	Enables

enableTrioptic

[in][out] A value that specifies whether to enable Trioptic Code 39.

0*	Disables
1	Enables

convertToCode32

[in][out] A value that specifies whether to convert Code 39 to Code 32 (= Italian Pharmacode).

0*	Does not convert
1	Converts

prefixCode32

[in][out] A value that specifies whether to transmit prefix for Code 32 data.

0*	Does not transmit
1	Transmits

checkDigitVerification

[in][out] A value that specifies whether to verify check digit.

0*	Does not verify
1	Verifies

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit..

0*	Does not transmit
1	Transmits

fullASCII

[in][out] A value that specifies whether to support Code 39 Full ASCII.

0*	No (Standard Code 39)
1	Yes (Code 39 Full ASCII)

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.5. CODE93_2D_4500

Code93_2D_SE4500	**2D**

Purpose: Configures symbology settings — Code 93.

Syntax:

```
int Code93_2D_SE4500 (int rw,
                      ref int enable,
                      ref int length1,
                      ref int length2);
```

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Code 93.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification. (see above)

0~55	55*

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.6. INTERLEAVED2OF5_2D_4500

Interleaved2Of5_2D_SE4500	**2D**

Purpose: Configures symbology settings — Interleaved 25.

Syntax:

```
int Interleaved2Of5_2D_SE4500 (int rw,
                               ref int enable,
                               ref int length1,
                               ref int length2,
                               ref int checkDigitVerification,
                               ref int transmitCheckDigit,
                               ref int convertToEAN13);
```

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Interleaved 25.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

checkDigitVerification

[in][out] A value that specifies whether and how to verify check digit.

0*	Disables
1	USS Check Digit
2	OPCC Check Digit

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit.

0*	Does not transmit
1	Transmits

convertToEAN13

[in][out] A value that specifies whether to convert Interleaved 25 to EAN-13.

0*	Does not convert
1	Converts

Return Value

If successful, it returns 0.
Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.7. INDUSTRIAL2OF5_2D_4500

Industrial2Of5_2D_SE4500 **2D**

Purpose

Configures symbology settings — Industrial 25 (= Discrete 25).

Syntax

int Industrial2Of5_2D_SE4500 (int *rw*,
ref int *enable*,
ref int *length1*,
ref int *length2*);

Parameters

The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Industrial 25 (= Discrete 25).

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

Return Value

If successful, it returns 0.
Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.8. MATRIX2OF5_2D_4500

Matrix2Of5_2D_SE4500	**2D**

Purpose: Configures symbology settings — Matrix 25.

Syntax:

```
int Matrix2Of5_2D_SE4500 (int rw,
                          ref int enable,
                          ref int length1,
                          ref int length2,
                          ref int redundancy,
                          ref int checkDigitVerification,
                          ref int transmitCheckDigit);
```

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Matrix 25.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

redundancy

[in][out] A value that specifies whether to enable decode redundancy.

0*	Disables
1	Enables

checkDigitVerification

[in][out] A value that specifies whether to verify check digit.

0*	Does not verify
1	Verifies

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit.

0*	Does not transmit
1	Transmits

Return Value

If successful, it returns 0.
Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.9. CHINESE2OF5_2D_4500

Chinese2Of5_2D_SE4500 **2D**

Purpose

Configures symbology settings — Chinese 25.

Syntax

int Chinese2Of5_2D_SE4500 (int *rw*,
ref int *enable*);

Parameters

The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Chinese 25.

0	Disables
1*	Enables

Return Value

If successful, it returns 0.
Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.10. CODABAR_2D_4500

Codabar_2D_SE4500	**2D**

Purpose: Configures symbology settings — Codabar.

Syntax:

int Codabar_2D_SE4500 (int *rw*,
ref int *enable*,
ref int *length1*,
ref int *length2*,
ref int *enableCLSI_Editing*,
ref int *enableNOTIS_Editing*);

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Codabar.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

enableCLSI_Editing

[in][out] A value that specifies whether to edit CLSI.

0*	Disables
1	Enables

enableNOTIS_Editing

[in][out] A value that specifies whether to edit NOTIS when decoding barcodes.

0*	Disables
1	Enables

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.11. MSI_2D_4500

Msi_2D_SE4500	**2D**

Purpose: Configures symbology settings — MSI.

Syntax:

```
int Msi_2D_SE4500 (int rw,
                   ref int enable,
                   ref int length1,
                   ref int length2,
                   ref int checkDigitVerification,
                   ref int transmitCheckDigit,
                   ref int checkDigitAlgorithm);
```

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable MSI.

0	Disables
1*	Enables

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification.

0~55	55*

checkDigitVerification

[in][out] A value that specifies how to verify check digit.

0*	Verifies one check digit
1	Verifies two check digits

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit.

0*	Does not transmit
1	Transmits

checkDigitAlgorithm

[in][out] A value that specifies which algorithm to apply.

0	Modulo 10 / Modulo 11
1*	Double Modulo 10

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.12. GS1_DATABAR_2D_4500

GS1_DataBar_2D_SE4500	**2D**

Purpose — Configures symbology settings — GS1 DataBar (= RSS family).

Syntax

```
int GS1_DataBar_2D_SE4500 (int rw,
                           ref int enableGS1_DataBarOmnidirectional,
                           ref int enableGS1_DataBarLimited,
                           ref int enableGS1_DataBarExpanded,
                           ref int convertToUpcEan);
```

Parameters — The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableGS1_DataBarOmnidirectional

[in][out] A value that specifies whether to enable GS1 DataBar Omnidirectional (= RSS-14).

0	Disables
1*	Enables

enableGS1_DataBarLimited

[in][out] A value that specifies whether to enable GS1 DataBar Limited (= RSS Limited).

0	Disables
1*	Enables

enableGS1_DataBarExpanded

[in][out] A value that specifies whether to enable GS1 DataBar Expanded (= RSS Expanded).

0	Disables
1*	Enables

convertToUpcEan

[in][out] A value that specifies whether to convert RSS to UPC/EAN barcodes.

0*	Does not convert
1	Converts

Return Value — If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.13. CODE128_2D_4500

Code128_2D_SE4500	**2D**

Purpose

Configures symbology settings — Code 128.

Syntax

```
int Code128_2D_SE4500 (int rw,
                       ref int enableCode128,
                       ref int enableGS1_128,
                       ref int enableISBT128,
                       ref int enableIsbtConcatenation,
                       ref int isbtConcatenationRedundancy);
```

Parameters

The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableCode128

[in][out] A value that specifies whether to enable Code 128.

0	Disables
1*	Enables

enableGS1_128

[in][out] A value that specifies whether to enable GS1-128.

0	Disables
1*	Enables

enableISBT128

[in][out] A value that specifies whether to enable ISBT 128.

0	Disables
1*	Enables

enableIsbtConcatenation

[in][out] A value that specifies whether to decode and concatenate pairs of ISBT barcodes.

0*	Disables concatenation.
1	Enables concatenation.
2	Enables auto-discrimination

isbtConcatenationRedundancy

[in][out] A value that specifies concatenation redundancy (2~20 times) when auto-discriminate of ISBT concatenation is enabled. By default, it is set to 10 times.

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.14. CODE11_2D_4500

Code11_2D_SE4500	**2D**

Purpose: Configures symbology settings — Code 11.

Syntax:

```
int Code11_2D_SE4500 (int rw,
                      ref int enable,
                      ref int numberOfCheckDigits,
                      ref int transmitCheckDigit,
                      ref int length1,
                      ref int length2);
```

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Code 11.

0	Disables
1*	Enables

numberOfCheckDigits

[in][out] A value that specifies whether and how to verify check digit.

0*	None
1	One check digit
2	Two check digits

transmitCheckDigit

[in][out] A value that specifies whether to transmit check digit.

0*	Does not transmit
1	Transmits

length1

[in][out] A value that specifies length qualification.

0~55	4*

length2

[in][out] A value that specifies length qualification. (see above)

0~55	55*

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.15. POSTALCODE_2D_4500

PostalCode_2D_SE4500	**2D**

Purpose — Configures symbology settings — Postal Codes.

Syntax

```
int PostalCode_2D_SE4500 (int rw,
                          ref int enableUSPostnet,
                          ref int enableUSPlanet,
                          ref int enableUKPostal,
                          ref int transmitUKCheckDigit,
                          ref int enableJapanPostal,
                          ref int enableAustralianPostal,
                          ref int enableDutchPostal,
                          ref int transmitUSCheckDigit);
```

Parameters — The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enableUSPostnet

[in][out] A value that specifies whether to enable US Postnet.

0	Disables
1*	Enables

enableUSPlanet

[in][out] A value that specifies whether to enable US Planet.

0	Disables
1*	Enables

enableUKPostal

[in][out] A value that specifies whether to enable UK Postal.

0	Disables
1*	Enables

transmitUKCheckDigit

[in][out] A value that specifies whether to transmit check digit for UK Postal data.

0	Does not transmit
1*	Transmits

enableJapanPostal

[in][out] A value that specifies whether to enable Japan Postal.

0	Disables
1*	Enables

enableAustralianPostal

[in][out] A value that specifies whether to enable Australian Postal.

0	Disables
1*	Enables

enableDutchPostal

[in][out] A value that specifies whether to enable Dutch Postal.

0	Disables
1*	Enables

transmitUSCheckDigit

[in][out] A value that specifies whether to transmit check digit for US Postal data.

0	Does not transmit
1*	Transmits

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.16. COMPOSITE_2D_4500

Composite_2D_SE4500	**2D**

Purpose — Configures symbology settings — Composite barcodes.

Syntax

```
int Composite_2D_SE4500 (int rw,
                         ref int enableCC_C,
                         ref int enableCC_AB,
                         ref int enableTLC39,
                         ref int enableUpcMode,
                         ref int beepMode,
                         ref int enableEmulationMode);
```

Parameters — The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enableCC_C

[in][out] A value that specifies whether to enable Composite CC-C.

0	Disables
1*	Enables

enableCC_AB

[in][out] A value that specifies whether to enable Composite CC-A/B.

0*	Disables
1	Enables

enableTLC39

[in][out] A value that specifies whether to enable Composite TLC 39 (= TCIF Linked Code 39).

0*	Disables
1	Enables

enableUpcMode

[in][out] A value that specifies whether to enable UPC Composite Mode.

0	Never links UPC (= discards 2D portion if there is any)
1*	Always links UPC (= discards whole barcode if there is no 2D)
2	Auto-discriminates UPC Composites

beepMode Reserved

enableEmulationModel

[in][out] A value that specifies whether to enable GS1-128 Emulation Mode for UCC/EAN Composite Codes.

0*	Disables
1	Enables

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.17. INVERSE1D_2D_SE4500

Inverse1D_2D_SE4500	**2D**

Purpose

Configures symbology settings — Inverse 1D

Syntax

int Inverse1D_2D_SE4500 (int *rw*,

ref int *enable1DInverse*);

Parameters

The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable1DInverse

[in][out] Set the 1D inverse decoder setting.

0*	Regular Only
1	Inverse Only
2	Inverse Autodetect

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.18. KOREAN3OF5_2D_SE4500

Korean3Of5_2D_SE4500	**2D**

Purpose: Configures symbology settings — Korean 3 Of 5

Syntax:

int Korean3Of5_2D_SE4500 (int *rw*,
ref int *enable*);

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enable

[in][out] A value that specifies whether to enable Korean 3 Of 5.

0*	Disables
1	Enables

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.19. SYMBOLOGIES_2D_4500

Symbologies_2D_SE4500	**2D**

Purpose: Configures 2D symbology settings.

Syntax:

```
int Symbologies_2D_SE4500(int rw,
                          ref int enablePDF417,
                          ref int enableMicroPDF417,
                          ref int enableCode128Emulation,
                          ref int enableDataMatrix,
                          ref int enableDataMatrixInverse,
                          ref int decodeMirrorImage,
                          ref int enableMaxicode,
                          ref int enableQRCode,
                          ref int enableQRCodeInverse,
                          ref int enableMicroQR,
                          ref int enableAztec,
                          ref int enableAztecInverse);
```

Parameters: The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	**Reader.ReaderEngineAPI.READ_PARAM**	Gets settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Sets settings

enablePDF417

[in][out] A value that specifies whether to enable PDF417.

0	Disables
1*	Enables

enableMicroPDF417

[in][out] A value that specifies whether to enable MicroPDF417.

0*	Disables
1	Enables

enableCode128Emulation

[in][out] A value that specifies whether to enable Code 128 Emulation for certain MicroPDF417 barcodes.

0*	Disables
1	Enables

- AIM Code Identifier must be enabled first!

When applied, MicroPDF417 barcodes are transmitted as if they were encoded in Code 128 barcodes with one of these prefixes (codewords 906, 907, 912, 914, 915 are not supported):

The first codeword of MicroPDF417 is 903-905:
The original Code ID “]L3” will be changed to “]C1”.
The first codeword of MicroPDF417 is 908 or 909:
The original Code ID “]L4” will be changed to “]C2”.
The first codeword of MicroPDF417 is 910 or 911:
The original Code ID “]L5” will be changed to “]C0”.

enableDataMatrix

[in][out] A value that specifies whether to enable Data Matrix.

0	Disables
1*	Enables

enableDataMatrixInverse

[in][out] A value that specifies whether to enable Data Matrix Inverse.

0*	Regular Only: decodes regular Data Matrix barcodes only
1	Inverse Only: decodes inverse Data Matrix barcodes only
2	Inverse Autodetect: decodes both regular and inverse Data Matrix barcodes

decodeMirrorImage

0*	Never
1	Always
2	Auto

enableMaxicode

[in][out] A value that specifies whether to enable Maxicode.

0	Disables
1*	Enables

enableQRCode

[in][out] A value that specifies whether to enable QR Code.

0	Disables
1*	Enables

enableQRCodeInverse

[in][out] A value that specifies whether to enable QR Code Inverse.

0*	Regular Only: decodes regular QR Code only
1	Inverse Only: decodes inverse QR Code only
2	Inverse Autodetect: decodes both regular and inverse QR Code

enableMicroQR

[in][out] A value that specifies whether to enable MicroQR.

0	Disables
1*	Enables

enableAztec

[in][out] A value that specifies whether to enable Aztec.

0	Disables
1*	Enables

enableAztecInverse

[in][out] A value that specifies whether to enable Aztec Inverse.

0*	Regular Only: decodes regular Aztec barcodes only
1	Inverse Only: decodes inverse Aztec barcodes only
2	Inverse Autodetect: decodes both regular and inverse Aztec barcodes

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.20. PICKLISTMODE_2D_SE4500

PicklistMode_2D_SE4500	**2D**

Purpose

Selects whether to enable picklist mode for decoding accuracy.

Syntax

PicklistMode_2D_SE4500 (int *rw*,

ref int *enable*);

Parameters

The default value (if there is) for each setting is indicated by an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	Reader.ReaderEngineAPI.READ_PARAM	Gets settings
'w'	Reader.ReaderEngineAPI.WRITE_PARAM	Sets settings

enable

[in][out] A value that specifies whether to enable picklist mode.

0*	Disables
2	Enables

Return Value

If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.7.21. DISPLAYMODE_2D

DisplayMode_2D	**2D**

Purpose: Selects whether to enable display mode.

Syntax:

DisplayMode_2D (int *rw*,
ref int *enable*);

Parameters: A default value comes with an asterisk "*".

rw

[in] A value that specifies the operation.

'r'	Reader.ReaderEngineAPI.READ_PARAM	Gets settings
'w'	Reader.ReaderEngineAPI.WRITE_PARAM	Writes settings

enable

[in][out] A value that specifies whether to enable display mode.

0*	Disables
1	Enables

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-201	E_IS_INITIAL_MODE

2.8. RESET READER

ResetReaderToDefault

Purpose: Resets reader module(s).

Syntax: **int ResetReaderToDefault (int** *target***);**

Parameters: *target*

[in] A value that specifies reader type.

7	DC_READER_BC	Resets barcode reader only.
32	DC_READER_RFID	Resets RFID reader only.
255	ALL_DEVICE	Resets both.

Example for C#:

```
int b1 = 0;
b1 = Reader.ReaderEngineAPI.ResetReaderToDefault(
Reader.ReaderEngineAPI.ALL_DEVICE);
```

Example for VB:

```
Dim b1 As Integer
b1 = Reader.ReaderEngineAPI.ResetReaderToDefault(255)
```

Return Value: If successful, it returns 0.

Otherwise, it returns 1.

- Call GetErrorCode() to find the error condition encountered:

-101	E_READER_NOT_INIT
-111	E_READER_BC_RESET_FAILED
-112	E_READER_RFID_RESET_FAILED

Remarks: It takes approximately 2 seconds to reset reader(s) to defaults.

See Also: InitReader

Chapter 3

SYSTEM API

Find the DLL files that are usable by your application on the bundled CD-ROM:

DLL required:
SystemMobile_Net.dll and/or SignatureDotNet.dll

Note: Signature capture doesn't require SystemMobile_Net.dll.

For RS-232 data connection, use COM 0 on the mobile computer.

3. IN THIS CHAPTER

3.1 System Settings 134
3.2 Status Indication..... 139
3.3 LCD Backlight 141
3.4 Keypad Backlight 148
3.5 Touch Panel 149
3.6 Keypad 151
3.7 Microphone 158
3.8 Wireless LAN 162
3.9 Bluetooth 208
3.10 Camera..... 228
3.11 GPS..... 240
3.12 Button Assignment 241
3.13 Signature Capture 247
3.14 Error Information 255

3.1. SYSTEM SETTINGS

3.1.1. API VERSION

GetNETAPIVersion

Purpose	To obtain the current .NET library version for system APIs.
Syntax	**int GetNETAPIVersion (ref string** *dotNetVer***);**
Parameters	*dotNetVer* [out] Pointer to a buffer where the version information is stored.

Example for C#	`int b1 = 0;`
	`string buf = string.Empty;`
	`b1 = Cipherlab.SystemAPI.Member.GetNETAPIVersion(ref buf);`

Example for VB	`Dim b1 As Integer`
	`Dim buf As String = ""`
	`b1 = Cipherlab.SystemAPI.Member.GetNETAPIVersion(buf)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

4	ERROR_PARAMETER (Wrong parameter)

3.1.2. UNIVERSALLY UNIQUE IDENTIFIER (UUID)

GetHALUUID

Purpose	To get information about the Universally Unique Identifier (UUID) that is based on an OEM-defined device hardware identifier.
Syntax	**int GetHALUUID (ref Guid** *guid***);**
Parameters	*guid* [out] Guid that stores the UUID information.

Example for C#	`int b1 = 0;`
	`Guid guid = new Guid();`
	`b1 = Cipherlab.SystemAPI.Member.GetHALUUID(ref guid);`

Example for VB	`Dim b1 As Integer`
	`Dim guid as new Guid`
	`b1 = Cipherlab.SystemAPI.Member.GetHALUUID(guid)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:
Remarks	A Universally Unique Identifier (UUID) is an identifier standard that provides a unique 128-bit (16 byte) value used to identify objects. The Globally Unique Identifier (GUID) is a Microsoft implementation of the UUID standard.

1	UUID inaccessible
4	ERROR_PARAMETER (Wrong parameter)

3.1.3. DEVICE NAME

GetSysDevName

Purpose	To get the device name that is identical to the one displayed in **Start** | **Settings** | **System** – Device Name tab on the mobile computer.
Syntax	**int GetSysDevName (ref string** *deviceName***);**
Parameters	*deviceName* [out] Pointer to a buffer where the device name is stored.

Example for C#	`int b1 = 0;`
	`string deviceName = string.Empty;`
	`b1 = Cipherlab.SystemAPI.Member.GetSysDevName(ref deviceName);`

Example for VB	`Dim b1 As Integer`
	`Dim deviceName As String = ""`
	`b1 = Cipherlab.SystemAPI.Member.GetSysDevName(deviceName)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

2	ERROR_NOSPACE (Wrong buffer size)
4	ERROR_PARAMETER (Wrong parameter)

Remarks	By default, the device name is provided by the manufacturer.

SetSysDevName

Purpose	To change the device name.
Syntax	**int SetSysDevName (string** *deviceName***);**
Parameters	*deviceName* [in] A string variable that stores the device name.

Example for C#	`int b1 = 0;`
	`string setname = "9200";`
	`b1 = Cipherlab.SystemAPI.Member.SetSysDevName(setname);`

Example for VB	`Dim b1 As Integer`
	`Dim setname as String = "9200"`
	`b1 = Cipherlab.SystemAPI.Member.SetSysDevName(setname)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

Remarks	By default, the device name is provided by the manufacturer. ▸ To apply the new name, you must warm boot the system. ▸ The device name may include up to 15 characters from the following ranges: a-z, A-Z, 0-9, or the '-' or '_' character. It must start with 'a-z' or 'A-Z', and cannot end with '-' or '_'.

3.1.4. SYSTEM INFORMATION

GetSysInfo

Purpose	To get the system information, such as serial number, device configuration, manufacturing date, and OS version.
Syntax	**int GetSysInfo (ref SysInfo** *sysInfo***);**
Parameters	*sysInfo* [out] SysInfo structure that stores the system information.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.SysInfo sysinfo =`
	`new Cipherlab.SystemAPI.Member.SysInfo();`
	`b1 = Cipherlab.SystemAPI.Member.GetSysInfo(ref sysinfo);`

Example for VB	`Dim b1 As Integer`
	`Dim sysinfo As New Member.SysInfo`
	`b1 = Cipherlab.SystemAPI.Member.GetSysInfo(sysinfo)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

4	ERROR_PARAMETER (Wrong parameter)
5	ERROR_API_FUNCTION (Fail to call System API)

3.1.5. START PROGRAM

SetInitLoaderBind

Purpose	To bind a program that will be loaded after the system is initialized.
Syntax	**int SetInitLoaderBind (KeyPadBind** *keypadBind*);
Parameters	*keypadBind* [in] KeyPadBind structure that stores the program information.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.KeyPadBind keybind =`
	`new Cipherlab.SystemAPI.Member.KeyPadBind();`
	`keybind.szClass = "\\DisckOnChip\\ImageMaker.exe";`
	`b1 = Cipherlab.SystemAPI.Member.SetInitLoaderBind(keybind);`

Example for VB	`Dim b1 As Integer`
	`Dim keybind As New Member.KeyPadBind`
	`keybind.szClass = "\DisckOnChip\ImageMaker.exe"`
	`b1 = Cipherlab.SystemAPI.Member.SetInitLoaderBind(keybind)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	Fail to set registry
2	Fail to set registry
4	ERROR_PARAMETER (Wrong parameter)

Remarks	The program you bind will be executed before AutoRun.ini and overwrite AutoRun.exe (= AutoRun.exe will never be run).

3.1.6. SOFT RESET

SystemSoftReset

Purpose	To perform software reset (warm boot).
Syntax	**int SystemSoftReset ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.SystemSoftReset();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.SystemSoftReset()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)

3.2. STATUS INDICATION

3.2.1. LED LIGHT

SetLEDLight

Purpose: To manipulate the status light (LED).

Syntax:

int SetLEDLight (byte *color*,
int *onOff*);

Parameters:

color

[in] Byte variable

0x00	Red LED
0x01	Green LED
0x02	Orange LED

onOff

[in] Integer variable

0	Turn off status light
1	Turn on status light

Example for C#:

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetLEDLight(0x01, 1);
```

Example for VB:

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetLEDLight(1, 1)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)
5	ERROR_API_FUNCTION (Fail to call System API)

Remarks: The time interval required between on and off must be greater than 400 milliseconds.

3.2.2. VIBRATOR

GetVibratorPower

Purpose	To find out the current vibrator state.
Syntax	**int GetVibratorPower (ref byte** *onOff***);**
Parameters	*onOff* [out] A byte variable that stores the information.

0	Vibrator OFF
1	Vibrator ON

Example for C#

```
int b1 = 0;
byte bState = new byte();
b1 = Cipherlab.SystemAPI.Member.GetVibratorPower(ref bState);
```

Example for VB

```
Dim b1 As Integer
Dim bState As New Byte
b1 = Cipherlab.SystemAPI.Member.GetVibratorPower(bState)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)
5	ERROR_API_FUNCTION (Fail to call System API)

SetVibratorPower

Purpose	To set the vibrator state.
Syntax	**int SetVibratorPower (byte** *onOff***);**
Parameters	*onOff* [in] Byte variable

0	Vibrator OFF
1	Vibrator ON

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetVibratorPower(0);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetVibratorPower(0)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)
5	ERROR_API_FUNCTION (Fail to call System API)

3.3. LCD BACKLIGHT

3.3.1. BACKLIGHT CONTROL

GetBacklightCTL

Purpose	To get the current settings of backlight control.
Syntax	**int GetBacklightCTL (ref BklCtl** *bklCtl***);**
Parameters	*bklCtl* [out] BklCtl structure that stores the settings.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.BklCtl bkctl =`
	`new Cipherlab.SystemAPI.Member.BklCtl();`
	`b1 = Cipherlab.SystemAPI.Member.GetBacklightCTL(ref bkctl);`

Example for VB	`Dim b1 As Integer`
	`Dim bkctl As New Member.BklCtl`
	`b1 = Cipherlab.SystemAPI.Member.GetBacklightCTL(bkctl)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

SetBacklightCTL

Purpose	To change the settings of backlight control.
Syntax	**int SetBacklightCTL (BklCtl** *bklCtl***);**
Parameters	*bklCtl* [in] BklCtl structure that stores the settings.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.BklCtl bklctl =`
	`new Cipherlab.SystemAPI.Member.BklCtl();`
	`bklctl.batttimeout = 30;`
	`bklctl.actimeout = 60;`
	`bklctl.backlightontap = 1;`
	`bklctl.lightlevel = 3;`
	`bklctl.acbacklightontap = 1;`
	`bklctl.aclightlevel = 6;`
	`bklctl.usebattery = 1;`
	`bklctl.useext = 0;`
	`b1 = Cipherlab.SystemAPI.Member.SetBacklightCTL(bklctl);`

Example for VB	`Dim b1 As Integer`
	`Dim bklctl As New Member.BklCtl`
	`bklctl.batttimeout = 30`
	`bklctl.actimeout = 60`
	`bklctl.backlightontap = 1`
	`bklctl.lightlevel = 3`
	`bklctl.acbacklightontap = 1`
	`bklctl.aclightlevel = 6`
	`bklctl.usebattery = 1`
	`bklctl.useext = 0`
	`b1 = Cipherlab.SystemAPI.Member.SetBacklightCTL(bklctl)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)
16	Out of range, backlight level in battery mode.
32	Out of range, backlight level in AC mode.

3.3.2. BACKLIGHT LEVEL

GetBacklightLV

Purpose

To get the current backlight level.

Syntax

int GetBacklightLV (byte *byFromAC*,
ref byte *byLevel*);

Parameters

byFromAC

[in] Byte variable

0	Backlight level for battery mode
1	Backlight level for AC mode

byLevel

[out] A byte variable that stores the information.

Example for C#

```
int b1 = 0;
byte aclevel = new byte();
b1 = Cipherlab.SystemAPI.Member.GetBacklightLV(1, ref aclevel);
```

Example for VB

```
Dim b1 As Integer
Dim aclevel As New Byte
b1 = Cipherlab.SystemAPI.Member.GetBacklightLV(1, aclevel)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

Remarks

The range of backlight level is from 0 (dark) to 11 (bright).

System defaults are level 6 in battery mode and level 11 in AC mode.

SetBacklightLV

Purpose	To set the backlight level.
Syntax	**int SetBacklightLV (byte** *byFromAC*, **byte** *byLevel*);
Parameters	*byFromAC* [in] Byte variable

0	Backlight level for battery mode
1	Backlight level for AC mode

byLevel
[in] Byte variable – Level 0 (dark) to Level 11 (bright)

Example for C#
```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetBacklightLV(1, 6);
```

Example for VB
```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetBacklightLV(1, 6)
```

Return Value
If successful, it returns 1.
Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

4	ERROR_PARAMETER (Wrong parameter)

3.3.3. RESET BACKLIGHT TO DEFAULT

SetBacklightDefault

Purpose	To reset the settings of backlight control to default.
Syntax	**int SetBacklightDefault ();**
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
16	Out of range, backlight level in battery mode.
32	Out of range, backlight level in AC mode.

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetBacklightDefault();
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetBacklightDefault()
```

Remarks

System defaults are –

```
public uint batttimeout=30;
public uint lightlevel=3;
public uint actimeout=60;
public uint aclightlevel=6;
public uint backlightontap=1;
public uint acbacklightontap=1;
public uint usebattery=1;
public uint useext=0;
```

See Also BklCtl

3.3.4. SET BACKLIGHT TO MAXIMUM

SetBacklightMax

Purpose	To set backlight control for maximum performance.
Syntax	**int SetBacklightMax ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.SetBacklightMax();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.SetBacklightMax()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
16	Out of range, backlight level in battery mode.
32	Out of range, backlight level in AC mode.

Remarks	The values for maximum performance are –
	public uint batttimeout=300;
	public uint lightlevel=10;
	public uint actimeout=600;
	public uint aclightlevel=10;
	public uint backlightontap=1;
	public uint acbacklightontap=1;
	public uint usebattery=1;
	public uint useext=1;

See Also	SetBacklightDefault, BklCtl

3.3.5. SET BACKLIGHT TO MINIMUM

SetBacklightMin	
Purpose	To set backlight control for minimum performance.
Syntax	**int SetBacklightMin ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.SetBacklightMin();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.SetBacklightMin()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
16	Out of range, backlight level in battery mode.
32	Out of range, backlight level in AC mode.

Remarks	The values for minimum performance are –
	public uint batttimeout=30;
	public uint lightlevel=0;
	public uint actimeout=60;
	public uint aclightlevel=0;
	public uint backlightontap=0;
	public uint acbacklightontap=1;
	public uint usebattery=1;
	public uint useext=1;

See Also	SetBacklightDefault, BklCtl

3.4. KEYPAD BACKLIGHT

GetKeylightOnOff

Purpose: To find out the keypad backlight status.

Syntax: **int GetKeylightOnOff (ref byte** *onOff***);**

Parameters: *onOff*

[out] A byte variable that stores the information.

0	Keypad backlight is off
1	Keypad backlight is on

Example for C#

```
int b1 = 0;
byte OnOff = 0;
b1 = Cipherlab.SystemAPI.Member.GetKeylightOnOff(ref OnOff);
```

Example for VB

```
Dim b1 As Integer
Dim OnOff As Byte = 0
b1 = Cipherlab.SystemAPI.Member.GetKeylightOnOff(OnOff)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

SetKeylightOnOff

Purpose: To set the keypad backlight.

Syntax: **int SetKeylightOnOff (byte** *onOff***);**

Parameters: *onOff*

[in] Byte variable

0	Set keypad backlight off
1	Set keypad backlight on

Example for C#

```
int b1 = 0;
byte OnOff = 0;
b1 = Cipherlab.SystemAPI.Member.SetKeylightOnOff(OnOff);
```

Example for VB

```
Dim b1 As Integer
Dim OnOff As Byte = 0
b1 = Cipherlab.SystemAPI.Member.SetKeylightOnOff(OnOff)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

3.5. TOUCH PANEL

3.5.1. TOUCH PANEL STATE

GetTchLock

Purpose	To find out the lock state of touch panel.
Syntax	**int GetTchLock (ref byte** *lockState***);**
Parameters	*lockState* [out] A byte variable that stores the information.

0	Touch panel unlocked
1	Touch panel locked

Example for C#

```
int b1 = 0;
byte bLock = new byte();
b1 = Cipherlab.SystemAPI.Member.GetTchLock(ref bLock);
```

Example for VB

```
Dim b1 As Integer
Dim bLock As New Byte
b1 = Cipherlab.SystemAPI.Member.GetTchLock(bLock)
```

Return Value

If successful, it returns 1.
Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	Fail to get touch panel handle
4	ERROR_PARAMETER (Wrong parameter)
5	ERROR_API_FUNCTION (Fail to call System API)

SetTchLock

Purpose	To set the lock state of touch panel.
Syntax	**int SetTchLock (byte** *lockState***);**
Parameters	*lockState* [in] Byte variable

0	Touch panel unlocked
1	Touch panel loco co ked

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetTchLock(0);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetTchLock(0)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	Fail to get touch panel handle
4	ERROR_PARAMETER (Wrong parameter)
5	ERROR_API_FUNCTION (Fail to call System API)

Note: You cannot lock both the touch panel and keypad at the same time!

3.6. KEYPAD

3.6.1. SENSITIVITY

GetKeypadSensitivity

Purpose	To find out the repeat rate of a key being held down for more than one second.
Syntax	**int GetKeypadSensitivity (ref uint** *value***);**
Parameters	*value* [out] An unsigned integer variable that stores the information.

Example for C#	`int b1 = 0;`
	`uint value = new uint();`
	`b1 = Cipherlab.SystemAPI.Member.GetKeypadSensitivity(ref value);`

Example for VB	`Dim b1 As Integer`
	`Dim value As new Uinteger`
	`b1 = Cipherlab.SystemAPI.Member.GetKeypadSensitivity(value)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	Fail to get key handle
4	ERROR_PARAMETER (Wrong parameter)

Remarks	The greater the value is, the more times the character repeats itself (per second) when you hold down a key for more than one second. ▸ By default, it is set to 6 times per second.

SetKeypadSensitivity

Purpose	To set the repeat rate of a key being held down for more than one second.
Syntax	**int SetKeypadSensitivity (uint** *value***);**
Parameters	*value* [in] Unsigned integer variable. The greater the value is, the more times the character repeats itself (max. 20 times per second) when you hold down a key for more than one second.
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.SetKeypadSensitivity(5);`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.SetKeypadSensitivity(5)`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	Fail to get key handle
4	ERROR_PARAMETER (Wrong parameter)

3.6.2. KEYPAD LOCK

GetKeypadLock

Purpose: To find out the lock state of keypad.

Syntax:

int GetKeypadLock (uint *key*,
ref uint *lockState*);

Parameters:

key

[in] Unsigned integer variable.

This parameter cannot be a combination of the following values:

1	KEY_GENERAL	General key state
2	KEY_ALPHA	[Alpha] key state
4	KEY_FN	[Fn] key state
32	KEY_SCAN	[Scan] key state
64	KEY_POWER	[Power] key state
128	KEY_LSTRIGGER	Left side trigger state
256	KEY_RSTRIGGER	Right side trigger state
512	KEY_SHIFT	[Shift] key state
0xFFFFFFFF	KEY_ALL	Whole keypad state

lockState

[out] An unsigned integer variable that stores the information.

0	Unlocked
1	Locked

Example for C#:

```
int b1 = 0;
uint key = 32, plock = 0;
b1 = Cipherlab.SystemAPI.Member.GetKeypadLock(key, ref plock);
```

Example for VB:

```
Dim b1 As Integer
Dim key As Uinteger = 32
Dim plock As Uinteger = 0
b1 = Cipherlab.SystemAPI.Member.GetKeypadLock(key, plock)
```

Return Value:

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	Fail to get key handle
4	ERROR_PARAMETER (Wrong parameter)

Remarks:

KEY_ALL = KEY_GENERAL + KEY_ALPHA + KEY_FN + KEY_SCAN + KEY_POWER + KEY_LSTRIGGER + KEY_RSTRIGGER + KEY_SHIFT

SetKeypadLock

Purpose: To set the lock state of keypad.

Syntax:

int SetKeypadLock (uint *key*,
uint *lockState*);

Parameters:

key

[in] Unsigned integer variable

This parameter can be a combination of the following values:

1	KEY_GENERAL	General key state
2	KEY_ALPHA	[Alpha] key state
4	KEY_FN	[Fn] key state
32	KEY_SCAN	[Scan] key state
64	KEY_POWER	[Power] key state
128	KEY_LSTRIGGER	Left side trigger state
256	KEY_RSTRIGGER	Right side trigger state
512	KEY_SHIFT	[Shift] key state
0xFFFFFFFF	KEY_ALL	Whole keypad state

lockState

[in] Unsigned integer variable

0	Unlock
1	Lock

Example for C#:

```
int b1 = 0;
uint key = 32, plock = 0;
b1 = Cipherlab.SystemAPI.Member.SetKeypadLock(key, plock);
```

Example for VB:

```
Dim b1 As Integer
Dim key As Uinteger = 32
Dim plock As Uinteger = 0
b1 = Cipherlab.SystemAPI.Member.SetKeypadLock(key, plock)
```

Return Value:

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	Fail to get key handle
2	Touch panel locked
4	ERROR_PARAMETER (Wrong parameter)

Remarks:

KEY_ALL = KEY_GENERAL + KEY_ALPHA + KEY_FN + KEY_SCAN + KEY_POWER + KEY_LSTRIGGER + KEY_RSTRIGGER + KEY_SHIFT

3.6.3. KEYPAD MODE

- Auto Resume mode — The function mode is toggled on by pressing the function key; it is toggled off by pressing the second key of the key combination. A status icon is displayed on the screen to indicate the status.
- Toggle mode — The function mode is toggled on by pressing the function key; it can only be toggled off by pressing the function key again. A status icon is displayed on the screen to indicate the status.
- Multi-Key mode — For any key combination, it requires pressing two keys at the same time, or holding down the function key followed by the second key.

GetKeypadMode

Purpose: To find out the key mode for a special key.

Syntax:

int GetKeypadMode (uint *key*,
ref uint *mode*);

Parameters:

key

[in] Unsigned integer variable

4	KEY_FN	To get [Fn] key mode.

Mode

[out] An unsigned integer variable that stores the information.

0	Toggle mode
1	Auto Resume mode
2	Multi-Key mode

Example for C#:

```
int b1 = 0;
uint key = 4, mode = 0;
b1 = Cipherlab.SystemAPI.Member.GetKeypadMode(key, ref mode);
```

Example for VB:

```
Dim b1 As Integer
Dim key As Uinteger = 4
Dim mode As Uinteger = 0
b1 = Cipherlab.SystemAPI.Member.GetKeypadMode(key, mode)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	Fail to get key handle
4	ERROR_PARAMETER (Wrong parameter)

SetKeypadMode

Purpose

To set the key mode for a special key.

Syntax

int SetKeypadMode (uint *key*,
uint *mode*);

Parameters

key

[in] Unsigned integer variable

4	KEY_FN	To set [Fn] key mode.

Mode

[in] Unsigned integer variable

0	Toggle mode
1	Auto Resume mode
2	Multi-Key mode

Example for C#

```
int b1 = 0;
uint key = 4, mode = 1;
b1 = Cipherlab.SystemAPI.Member.SetKeypadMode(key, mode);
```

Example for VB

```
Dim b1 As Integer
Dim key As Uinteger = 4
Dim mode As Uinteger = 1
b1 = Cipherlab.SystemAPI.Member.SetKeypadMode(key, mode)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	Fail to get key handle
4	ERROR_PARAMETER (Wrong parameter)

3.6.4. KEYPAD STATE

GetKeypadState

Purpose: To find out the key state for a special key.

Syntax:

int GetKeypadState (uint *key*,
ref uint *state*);

Parameters:

key

[in] Unsigned integer variable

2	KEY_ALPHA	To get [Alpha] key state.
4	KEY_FN	To get [Fn] key state.

State

[out] An unsigned integer variable that stores the information.

	Alpha key	FN key
0	Numeric	Inactive
1	N/A	Active
2	Lower-case	N/A

Example for C#:

```
int b1 = 0;
uint key = 4, state = 0;
b1 = Cipherlab.SystemAPI.Member.GetKeypadState(key, ref state);
```

Example for VB:

```
Dim b1 As Integer
Dim key As Uinteger = 4
Dim state As Uinteger = 0
b1 = Cipherlab.SystemAPI.Member.GetKeypadState(key, state)
```

Return Value:

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	Fail to get key handle
4	ERROR_PARAMETER (Wrong parameter)

SetKeypadState

Purpose

To set the key state for a special key.

Syntax

int SetKeypadState (uint *key*,
uint *state*);

Parameters

key

[in] Unsigned integer variable

2	KEY_ALPHA	To set [Alpha] key state.
4	KEY_FN	To set [Fn] key state.

State

[in] Unsigned integer variable

	Alpha key	FN key
0	Numeric	Inactive
1	N/A	Active
2	Lower-case	N/A

Example for C#

```
int b1 = 0;
uint key = 4, state = 1;
b1 = Cipherlab.SystemAPI.Member.SetKeypadState(key, state);
```

Example for VB

```
Dim b1 As Integer
Dim key As Uinteger = 4
Dim state As Uinteger = 1
b1 = Cipherlab.SystemAPI.Member.SetKeypadState(key, state)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	Fail to get key handle
2	Key locked
4	ERROR_PARAMETER (Wrong parameter)

3.7. MICROPHONE

3.7.1. DEVICE MICROPHONE

SetAMicGain

Purpose: To set a customized device microphone gain level.

Syntax: **int SetAMicGain (uint** *level***);**

Parameters: *level*

[in] Unsigned integer variable

A value that specifies the level of device microphone gain.

0	0
1	7281
2	14563
3	21845
4	29126
5	36408
6	43690
7	50971
8	58253
9	65535

Example for C#

```
int b1 = 0, level = 0;
b1 = Cipherlab.SystemAPI.Member.SetAMicGain(level);
```

Example for VB

```
Dim b1 As Integer
Dim level As Integer
b1 = Cipherlab.SystemAPI.Member.SetAMicGain(level)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

GetAMicGain

Purpose

To get device microphone gain level for customization.

Syntax

int GetAMicGain (ref uint *level***);**

Parameters

level

[out] Unsigned integer variable

A value that specifies the level of device microphone gain.

0	0
1	7281
2	14563
3	21845
4	29126
5	36408
6	43690
7	50971
8	58253
9	65535

- Before using this function, set a default level first with SetAMicGain.

Example for C#

```
int b1 = 0, level = 0;
b1 = Cipherlab.SystemAPI.Member.GetAMicGain(ref level);
```

Example for VB

```
Dim b1 As Integer
Dim level As Integer
b1 = Cipherlab.SystemAPI.Member.GetAMicGain(level)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

3.7.2. HEADSET MICROPHONE

SetHSMicGain

Purpose: To set a customized headset microphone gain level.

Syntax: **int SetHSMicGain (uint** *level***);**

Parameters:

level

[in] Unsigned integer variable

A value that specifies the level of headset microphone gain.

0	0
1	7281
2	14563
3	21845
4	29126
5	36408
6	43690
7	50971
8	58253
9	65535

Example for C#

```
int b1 = 0, level = 0;
b1 = Cipherlab.SystemAPI.Member.SetAMicGain(level);
```

Example for VB

```
Dim b1 As Integer
Dim level As Integer
b1 = Cipherlab.SystemAPI.Member.SetAMicGain(level)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

GetHSMicGain

Purpose

To get the customized headset microphone gain level.

Syntax

int GetHSMicGain (ref uint *level***);**

Parameters

level

[out] Unsigned integer variable

A value that specifies the level of headset microphone gain.

0	0
1	7281
2	14563
3	21845
4	29126
5	36408
6	43690
7	50971
8	58253
9	65535

- Before using this function, set a default level first with SetHSMicGain.

Example for C#

```
int b1 = 0, level = 0;
b1 = Cipherlab.SystemAPI.Member.GetHSMicGain(ref level);
```

Example for VB

```
Dim b1 As Integer
Dim level As Integer
b1 = Cipherlab.SystemAPI.Member.GetHSMicGain(level)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

3.8. WIRELESS LAN

3.8.1. WI-FI POWER

GetWiFiPower

Purpose	To find out the current power state of the Wi-Fi module.
Syntax	**int GetWiFiPower (ref byte** *onOff***);**
Parameters	*onOff* [out] A byte variable that stores the information.

0	Turn off the power to the Wi-Fi module
1	Turn on the power to the Wi-Fi module

Example for C#

```
int b1 = 0;
byte poweron = new byte();
b1 = Cipherlab.SystemAPI.Member.GetWiFiPower(ref poweron);
```

Example for VB

```
Dim b1 As Integer
Dim poweron As New Byte
b1 = Cipherlab.SystemAPI.Member.GetWiFiPower(poweron)
```

Return Value

If successful, it returns 1.
Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	Fail to get Wi-Fi power state
4	ERROR_PARAMETER (Wrong parameter)

SetWiFiPower

Purpose	To set the power state of the Wi-Fi module.
Syntax	**int SetWiFiPower (byte** *onOff***);**
Parameters	*onOff* [in] Byte variable

0	Turn off the power to the Wi-Fi module
1	Turn on the power to the Wi-Fi module

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetWiFiPower(1);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetWiFiPower(1)
```

Return Value

If successful, it returns 1.
Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	Fail to open Wi-Fi device handle
2	Fail to set Wi-Fi power
4	ERROR_PARAMETER (Wrong parameter)

3.8.2. RADIO

RadioDisable	
Purpose	To disable the radio.
Syntax	**int RadioDisable ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.RadioDisable();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.RadioDisable()`
Return Value	If successful, it returns 1. Otherwise, it returns 0.

RadioEnable	
Purpose	To enable the radio.
Syntax	**int RadioEnable ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.RadioEnable();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.RadioEnable()`
Return Value	If successful, it returns 1. Otherwise, it returns 0.

3.8.3. IP INFORMATION

GetWlanIpInfo	
Purpose	To get the IP information for wireless networking.
Syntax	**int GetWlanIpInfo (ref WlanAdptInfo** *adpt***);**
Parameters	*adpt* [out] WlanAdptInfo structure that stores the information.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.WlanAdptInfo wlanAdptInfo =`
	`new Cipherlab.SystemAPI.Member.WlanAdptInfo();`
	`b1 = Cipherlab.SystemAPI.Member.GetWlanIpInfo(ref wlanAdptInfo);`

Example for VB	`Dim b1 As Integer`
	`Dim wlanAdptInfo As New Member.WlanAdptInfo`
	`b1 = Cipherlab.SystemAPI.Member.GetWlanIpInfo(wlanAdptInfo)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered: 1 — ERROR_NORESOURCE (Fail to get resource) 4 — ERROR_PARAMETER (Wrong parameter)
Remarks	When DHCP is enabled, this function will get DHCP status only.

SetWlanIpInfo	
Purpose	To set the IP information for wireless networking.
Syntax	**int SetWlanIpInfo (WlanAdptInfo** *adpt***);**
Parameters	*adpt* [in] WlanAdptInfo structure that stores the information.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.WlanAdptInfo setwlinfo =`
	`new Cipherlab.SystemAPI.Member.WlanAdptInfo();`
	`setwlinfo.fUseDHCP = 0;`
	`setwlinfo.IPAddr = “192.168.6.153”;`
	`setwlinfo.SubnetMask = “255.255.255.0”;`
	`setwlinfo.Gateway = “192.168.6.253”;`
	`b1 = Cipherlab.SystemAPI.Member.SetWlanIpInfo(setwlinfo);`

Example for VB	`Dim b1 As Integer`
	`Dim setwlinfo As New Member.WlanAdptInfo`
	`setwlinfo.fUseDHCP = 0`
	`setwlinfo.IPAddr = “192.168.6.153”`
	`setwlinfo.SubnetMask = “255.255.255.0”`
	`setwlinfo.Gateway = “192.168.6.253”`
	`b1 = Cipherlab.SystemAPI.Member.SetWlanIpInfo(setwlinfo)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

3.8.4. DOMAIN, SDK & MAC INFORMATION

GetCurrentDomain

Purpose	To find out the regulatory domain or domains for which the radio is configured.
Syntax	**int GetCurrentDomain ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.GetCurrentDomain();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.GetCurrentDomain()`
Return Value	If successful, it returns the regulatory domain or domains:

0	REG_FCC	(Americas)
1	REG_ETSI	(Europe)
2	REG_TELEC	(Japan)
3	REG_WW	(Worldwide)
4	REG_KCC	(Korea)

Remarks	This function reads the values from SROM and the execution time is significant. Therefore, it must not be called frequently.

GetSDKVersion

Purpose	To find out Summit radio SDK version.
Syntax	**int GetSDKVersion (ref uint** *version***);**
Parameters	*version* [out] An unsigned integer variable that stores the information.

Example for C#	`int b1 = 0;`
	`uint version = 0;`
	`b1 = Cipherlab.SystemAPI.Member.GetSDKVersion(ref version);`

Example for VB	`Dim b1 As Integer`
	`Dim version AS UInteger`
	`b1 = Cipherlab.SystemAPI.Member.GetSDKVersion(version)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

GetWiFiMac	
Purpose	To get the MAC address of Summit radio.
Syntax	**int GetWiFiMac (ref ulong** *wifiMac***);**
Parameters	*wifiMac* [out] Pointer to a buffer where the MAC address is stored.

Example for C#	`int b1 = 0;`
	`ulong wifiMac = 0;`
	`b1 = Cipherlab.SystemAPI.Member.GetWiFiMac(ref wifiMac);`

Example for VB	`Dim b1 As Integer`
	`Dim wifiMac As ULong`
	`b1 = Cipherlab.SystemAPI.Member.GetWiFiMac(wifiMac)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
2	ERROR_API_FUNCTION (Fail to call System API)
3	Wi-Fi power not ready

3.8.5. NETWORK LIST

AddWlanSsidToPreferredList

Purpose	To add a wireless network (via an access point) to the preferred list.
Syntax	**int AddWlanSsidToPreferredList (WlanCtl** *wlanCtl***);**
Parameters	*wlanCtl* [in] WlanCtl structure that stores the information.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.WlanCtl wlanCtl =`
	`new Cipherlab.SystemAPI.Member.WlanCtl();`
	`wlanCtl.SSID = “Netgear”;`
	`wlanCtl.authentication = 0;`
	`wlanCtl.encryption = 1;`
	`wlanCtl.adhoc = 0;`
	`wlanCtl.eap = 0;`
	`wlanCtl.key = “”;`
	`b1 = Cipherlab.SystemAPI.Member.AddWlanSsidToPreferredList(wlanCtl);`

Example for VB	`Dim wlanCtl As New Member.WlanCtl`
	`wlanCtl.SSID = “Netgear”`
	`wlanCtl.authentication = 0`
	`wlanCtl.encryption = 1`
	`wlanCtl.adhoc = 0`
	`wlanCtl.eap = 0`
	`wlanCtl.key = “”`
	`Cipherlab.SystemAPI.Member.AddWlanSsidToPreferredList(wlanCtl)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)
8	ERROR_WLAN_QUIRY_INTERFACE (Fail to inquiry interface)
16	ERROR_WLAN_SSID_REPEAT

GetWlanAvailableList

Purpose	To get the list of available wireless networks (via access points) about detail information like MacAddress、SSID、Rssi….
Syntax	**int GetWlanAvailableList (BSSIDINFO[]** *bssidInfo*, **ref int** *number***);**
Parameters	*bssidInfo* [in][out] Pointer to the structure BSSIDINFO where the list is stored. The maximum arrary size is 50. *number* [in][out] number is quantity of AP.

Example for C#	`int b1 = 0;`
	`int n = 0;`
	`Cipherlab.SystemAPI.Member.BSSIDINFO[] bs =`
	`new Cipherlab.SystemAPI.Member.BSSIDINFO[50];`
	`b1 = Cipherlab.SystemAPI.Member.GetWlanAvailableList(bs,ref n);`

Example for VB	`Dim n As Integer`
	`Dim bs() As Member.BSSIDINFO(49){}`
	`Cipherlab.SystemAPI.Member.GetWlanAvailableList(bs, ByRef n As Integer)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

2	ERROR_NOSPACE (Buffer is too small)
4	ERROR_PARAMETER (Wrong parameter)
8	ERROR_WLAN_QUIRY_INTERFACE (Fail to inquiry interface)
19	ERROR_WLAN_NO_AVAILABLE_LIST

Remarks	Refer to System API's sample code.
See Also	GetWlanPreferredList

GetWlanPreferredList

Purpose	To get the list of preferred wireless networks (via access points) about detail information like MacAddress、SSID、Rssi….
Syntax	**int GetWlanPreferredList (BSSIDINFO[]** *bssidInfo*, **ref int** *number*);
Parameters	*bssidInfo* [in][out] Pointer to the structure BSSIDINFO where the list is stored. The maximum arrary size is 50. *number* [in][out] number is quantity of AP.

Example for C#	`int b1 = 0;`
	`int n = 0;`
	`Cipherlab.SystemAPI.Member.BSSIDINFO[] bs =`
	`new Cipherlab.SystemAPI.Member.BSSIDINFO[50];`
	`b1 = Cipherlab.SystemAPI.Member.GetWlanPreferredList(bs, ref n);`

Example for VB	`Dim n AS Integer`
	`Dim bs() As Member.BSSIDIFNO(49){}`
	`Cipherlab.SystemAPI.Member.GetWlanPreferredList(bs, ByRef n As Integer)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

2	ERROR_NOSPACE (Buffer is too small)
4	ERROR_PARAMETER (Wrong parameter)
8	ERROR_WLAN_QUIRY_INTERFACE (Fail to inquiry interface)
17	ERROR_WLAN_NO_PRSSID_LIST

Remarks	Refer to System API's sample code.
See Also	AddWlanSsidToPreferredList, GetWlanAvailableList, ReconnectWlanPreferredList, ResetWlanPreferredList

ResetWlanPreferredList

Purpose	To clear the list of preferred wireless networks (via access points).
Syntax	**int ResetWlanPreferredList ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.ResetWlanPreferredList();`
Example for VB	`Cipherlab.SystemAPI.Member.ResetWlanPreferredList()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

8	ERROR_WLAN_QUIRY_INTERFACE (Fail to inquiry interface)

Remarks	It will terminate the current connection.

3.8.6. NETWORK STATUS

GetCurrentStatus

Purpose	To find out the connection status, such as information of IP, MAC, AP association, etc.
Syntax	**int GetCurrentStatus (ref CF10G_STATUS** *status***);**
Parameters	*status* [out] CF10G_STATUS structure that stores the information.

Example for C#	`int b1 = 0;`
	`Member.CF10G_Status cfs = new Member.CF10G_Status();`
	`b1 = Cipherlab.SystemAPI.Member.GetCurrentStatus(ref cfs);`

Example for VB	`Dim b1 As Integer`
	`Dim cfs As New Member.CF10G_Status()`
	`b1 = Cipherlab.SystemAPI.Member.GetCurrentStatus(cfs)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

GetWlanConnectedStatus

Purpose	To get the status of current connection.
Syntax	**int GetWlanConnectedStatus (ref WlanConnectedSt** *wlanConnSt***);**
Parameters	*wlanConnSt* [in][out] WlanConnectedSt structure that stores the information.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.WlanConnectedSt wlanConnSt =`
	`new Cipherlab.SystemAPI.Member.WlanConnectedSt();`
	`b1 =`
	`Cipherlab.SystemAPI.Member.GetWlanConnectedStatus(ref wlanConnSt);`

Example for VB	`Dim wlanConnSt As New Member.WlanConnectedSt` `Cipherlab.SystemAPI.Member.GetWlanConnectedStatus(wlanConnSt)`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

4	ERROR_PARAMETER (Wrong parameter)
8	ERROR_WLAN_QUIRY_INTERFACE (Fail to inquiry interface)
18	ERROR_WLAN_NO_ASSOCIATED
20	ERROR_WLAN_NO_CONNECTED

ReconnectWlanPreferredList

Purpose	To reconnect to an access point on the list of preferred wireless networks.
Syntax	**int ReconnectWlanPreferredList ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.ReconnectWlanPreferredList();`
Example for VB	`Cipherlab.SystemAPI.Member.ReconnectWlanPreferredList()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

8	ERROR_WLAN_QUIRY_INTERFACE (Fail to inquiry interface)

3.8.7. BSSID

ScanWiFiBSSID

Purpose	To scan available access points.
Syntax	**int ScanWiFiBSSID ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.ScanWiFiBSSID();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.ScanWiFiBSSID()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

4	ERROR_PARAMETER (Wrong parameter)

GetWiFiBSSIDList

Purpose	To obtain the BSSID list. (The maximum arrary size is 50.)
Syntax	**int GetWiFiBSSIDList (byte[]** *buffer***);**
Parameters	*buffer* [out] A byte array that stores the information.

Example for C#	`int b1 = 0;`
	`byte[] buf = new byte[6136];`
	`b1 = Cipherlab.SystemAPI.Member.GetWifiBSSIDList(buf);`

Example for VB	`Dim b1 As Integer`
	`Dim buf(6136) As Byte`
	`b1 = Cipherlab.SystemAPI.Member.GetWifiBSSIDList(buf)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

4	ERROR_PARAMETER (Wrong parameter)

3.8.8. CONFIGURATION PROFILES

ActivateConfig

Purpose: To activate a configuration profile.

Syntax: **int ActivateConfig (string** *name***);**

Parameters: *name*
[in] A string variable that stores the profile name.

Example for C#:
```
int b1 = 0;
string name = “AP”;
b1 = Cipherlab.SystemAPI.Member.ActivateConfig(name);
```

Example for VB:
```
Dim b1 As Integer
Dim name As String = “AP”
b1 = Cipherlab.SystemAPI.Member.ActivateConfig(name)
```

Return Value: If successful, it returns 1.
Otherwise, it returns 0.

GetCurrentConfig

Purpose: To find out the configuration currently in use.

Syntax: **int GetCurrentConfig (ref int** *num***,**
ref string *name***);**

Parameters: *num*
[out] An integer variable that stores the information.

0	ThirdPartyConfig is active **(Reserved)**
Greater than 0	Identifier of the active configuration profile

name
[out] Pointer to a buffer where the name of configuration profile is stored. Buffer size must be greater than CONFIG_NAME_SZ.

Example for C#:
```
int b1 = 0, num = 0;
string name = “”;
b1 = Cipherlab.SystemAPI.Member.GetCurrentConfig(ref num, ref name);
```

Example for VB:
```
Dim b1 As Integer
Dim num As Integer
Dim name As String = “”
b1 = Cipherlab.SystemAPI.Member.GetCurrentConfig(num, name)
```

Return Value: If successful, it returns 1.
Otherwise, it returns 0.

GetNumConfigs	
Purpose	To get the total number of available configuration profiles.
Syntax	**int GetNumConfigs (ref int** *num***);**
Parameters	*num* [out] An integer variable that stores the information.
Example for C#	`int b1 = 0, num = 0;` `b1 = Cipherlab.SystemAPI.Member.GetNumConfigs(ref num);`

Example for VB	`Dim b1 As Integer`
	`Dim num As Integer`
	`b1 = Cipherlab.SystemAPI.Member.GetNumConfigs(num)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

3.8.9. EDIT CONFIGURATION PROFILE

AddConfig	
Purpose	To add a configuration profile.
Syntax	**int AddConfig (SDCConfig** *config***);**
Parameters	*config* [in] SDCConfig structure that stores the information.

Example for C#	`int b1 = 0;`
	`Member.SDCConfig sdccfg = new Member.SDCConfig(1);`
	`sdccfg.configName = "AP";`
	`sdccfg.SSID = "AP";`
	`sdccfg.clientName = "";`
	`sdccfg.txPower = 0;`
	`sdccfg.authType = Member.AUTH.AUTH_OPEN;`
	`sdccfg.eapType = Member.EAPTYPE.EAP_NONE;`
	`sdccfg.powerSave = Member.POWERSAVE.POWERSAVE_OFF;`
	`sdccfg.wepType = Member.WEPTYPE.WEP_OFF;`
	`sdccfg.bitRate = Member.BITRATE.BITRATE_AUTO;`
	`sdccfg.radioMode = Member.RADIOMODE.RADIOMODE_B_ONLY;`
	`sdccfg.userName.size = 0;`
	`sdccfg.userName.buffer = "";`
	`sdccfg.userName.offset = 0`
	`sdccfg.userPwd.size = 0;`
	`sdccfg.userPwd.buffer = "";`
	`sdccfg.userPwd.offset = 0;`
	`sdccfg.PSK.size = 0;`
	`sdccfg.PSK.buffer = "";`
	`sdccfg.PSK.offset = 0;`
	`sdccfg.WEPKeys.size = 0;`
	`sdccfg.WEPKeys.buffer = "";`
	`sdccfg.WEPKeys.offset = 0;`
	`b1 = Cipherlab.SystemAPI.Member.AddConfig(sdccfg);`

Example for VB	`Dim b1 As Integer`
	`Dim sdccfg As New Member.SDCConfig(1)`
	`sdccfg.configName = "AP";`
	`sdccfg.SSID = "AP";`
	`sdccfg.clientName = "";`
	`sdccfg.txPower = 0;`
	`sdccfg.authType = Member.AUTH.AUTH_OPEN;`
	`sdccfg.eapType = Member.EAPTYPE.EAP_NONE;`
	`sdccfg.powerSave = Member.POWERSAVE.POWERSAVE_OFF;`
	`sdccfg.wepType = Member.WEPTYPE.WEP_OFF;`
	`sdccfg.bitRate = Member.BITRATE.BITRATE_AUTO;`
	`sdccfg.radioMode = Member.RADIOMODE.RADIOMODE_B_ONLY;`
	`sdccfg.userName.size = 0;`
	`sdccfg.userName.buffer = "";`

```
sdccfg.userName.offset = 0;
sdccfg.userPwd.size = 0;
sdccfg.userPwd.buffer = "";
sdccfg.userPwd.offset = 0;
sdccfg.PSK.size = 0;
sdccfg.PSK.buffer = "";
sdccfg.PSK.offset = 0;
sdccfg.WEPKeys.size = 0;
sdccfg.WEPKeys.buffer = "";
sdccfg.WEPKeys.offset = 0;
b1 = Cipherlab.SystemAPI.Member.AddConfig(sdccfg)
```

Return Value: If successful, it returns 1.
Otherwise, it returns 0.

DeleteConfig

Purpose: To delete a configuration profile.

Syntax: **int DeleteConfig (string** *name***);**

Parameters: *name*
[in] A string variable that stores the profile name.

Example for C#:

```
int b1 = 0;
string name = “AP”;
b1 = Cipherlab.SystemAPI.Member.DeleteConfig(name);
```

Example for VB:

```
Dim b1 As Integer
Dim name As String = “AP”
b1 = Cipherlab.SystemAPI.Member.DeleteConfig(name)
```

Return Value: If successful, it returns 1.
Otherwise, it returns 0.

Remarks:
- You are not allowed to delete the active configuration.
- NULL is not a valid name

SetDefaultConfigValues	
Purpose	To set default values for new configurations.
Syntax	**int SetDefaultConfigValues (SDCConfig** *config***);**
Parameters	*config* [in] SDCConfig structure that stores the information.

Example for C#	`int b1 = 0;`
	`Member.SDCConfig sdccfg = new Member.SDCConfig(1);`
	`sdccfg.configName = "AP";`
	`sdccfg.SSID = "AP";`
	`sdccfg.clientName = "";`
	`sdccfg.txPower = 0;`
	`sdccfg.authType = Member.AUTH.AUTH_OPEN;`
	`sdccfg.eapType = Member.EAPTYPE.EAP_NONE;`
	`sdccfg.powerSave = Member.POWERSAVE.POWERSAVE_OFF;`
	`sdccfg.wepType = Member.WEPTYPE.WEP_OFF;`
	`sdccfg.bitRate = Member.BITRATE.BITRATE_AUTO;`
	`sdccfg.radioMode = Member.RADIOMODE.RADIOMODE_B_ONLY;`
	`sdccfg.userName.size = 0;`
	`sdccfg.userName.buffer = "";`
	`sdccfg.userName.offset = 0;`
	`sdccfg.userPwd.size = 0;`
	`sdccfg.userPwd.buffer = "";`
	`sdccfg.userPwd.offset = 0;`
	`sdccfg.PSK.size = 0;`
	`sdccfg.PSK.buffer = "";`
	`sdccfg.PSK.offset = 0;`
	`sdccfg.WEPKeys.size = 0;`
	`sdccfg.WEPKeys.buffer = "";`
	`sdccfg.WEPKeys.offset = 0;`
	`b1 = Cipherlab.SystemAPI.Member.SetDefaultConfigValue(config);`

Example for VB	`Dim b1 As Integer`
	`Dim sdccfg As New Member.SDCConfig(1)`
	`sdccfg.configName = "AP";`
	`sdccfg.SSID = "AP";`
	`sdccfg.clientName = "";`
	`sdccfg.txPower = 0;`
	`sdccfg.authType = Member.AUTH.AUTH_OPEN;`
	`sdccfg.eapType = Member.EAPTYPE.EAP_NONE;`
	`sdccfg.powerSave = Member.POWERSAVE.POWERSAVE_OFF;`
	`sdccfg.wepType = Member.WEPTYPE.WEP_OFF;`
	`sdccfg.bitRate = Member.BITRATE.BITRATE_AUTO;`
	`sdccfg.wepType = Member.WEPTYPE.WEP_OFF;`
	`sdccfg.bitRate = Member.BITRATE.BITRATE_AUTO;`
	`sdccfg.radioMode = Member.RADIOMODE.RADIOMODE_B_ONLY;`
	`sdccfg.userName.size = 0;`
	`sdccfg.userName.buffer = "";`

```
sdccfg.userName.offset = 0;
sdccfg.userPwd.size = 0;
sdccfg.userPwd.buffer = "";
sdccfg.userPwd.offset = 0;
sdccfg.PSK.size = 0;
sdccfg.PSK.buffer = "";
sdccfg.PSK.offset = 0;
sdccfg.WEPKeys.size = 0;
sdccfg.WEPKeys.buffer = "";
sdccfg.WEPKeys.offset = 0;
b1 = Cipherlab.SystemAPI.Member.SetDefaultConfigValues (sdccfg)
```

Return Value: If successful, it returns 1.
Otherwise, it returns 0.

CreateConfig

Purpose: To create a configuration from the default values.

Syntax: **int CreateConfig (ref SDCConfig** *config***);**

Parameters:
config
[out] SDCConfig structure that stores the information.

Example for C#:

```
int b1 = 0;
Member.SDCConfig sdccfg = new Member.SDCConfig(1);
b1 = Cipherlab.SystemAPI.Member.CreateConfig(ref sdccfg);
```

Example for VB:

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig(1)
b1 = Cipherlab.SystemAPI.Member.CreateConfig(name, sdccfg)
```

Return Value: If successful, it returns 1.
Otherwise, it returns 0.

Remarks: You must add the config after it is created.

GetConfig

Purpose: To get a configuration profile.

Syntax: **int GetConfig (string** *name*,
ref SDCConfig *config*);

Parameters:
name
[in] A string variable that stores the profile name.
config
[out] SDCConfig structure that stores the information.
- Cannot be NULL.

Example for C#:

```
int b1 = 0;
string name = “AP”;
Member.SDCConfig sdccfg = new Member.SDCConfig(1);
b1 = Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
```

Example for VB

```
Dim b1 As Integer
Dim name As String = “AP”
Dim sdccfg As New Member.SDCConfig(1)
b1 = Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
```

Return Value

If successful, it returns 1.
Otherwise, it returns 0.

ModifyConfig

Purpose

To modify a configuration profile.

Syntax

int ModifyConfig (string *name*,
SDCConfig *config*);

Parameters

name
[in] A string variable that stores the profile name.
config
[in] SDCConfig structure that stores the information.

- Name of the configuration to update.

Example for C#

```
int b1 = 0;
string name = “AP”;
Member.SDCConfig sdccfg = new Member.SDCConfig(1);
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
sdccfg.txPower = 5;
b1 = Cipherlab.SystemAPI.Member.Modify(name, sdccfg);
```

Example for VB

```
Dim b1 As Integer
Dim name As String = “AP”
Dim sdccfg As New Member.SDCConfig(1)
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
sdccfg.txPower = 5
b1 = Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg)
```

Return Value

If successful, it returns 1.
Otherwise, it returns 0.

3.8.10. SUMMIT CONFIG SETTINGS

GetAllConfigs

Purpose: To retrieve all configuration profiles.

Syntax:
int GetAllConfigs (ref SDCConfig[] *config*,
ref int *num*);

Parameters:
config
[out] SDCConfig structure that stores the information. The total elements of this array must be greater than MAX_CFGS defined in Cipherlab.SystemAPI.Member.
num
[out] An integer variable that stores the number of elements.

Example for C#:
```
int b1 = 0, num = 0;
Member.SDCConfig[] sdccfg = new Member.SDCConfig[20];
b1 = Cipherlab.SystemAPI.Member.GetAllConfigs(ref sdccfg, ref num);
```

Example for VB:
```
Dim b1 As Integer
Dim num As Integer
Dim sdccfg(20) As New Member.SDCConfig
b1 = Cipherlab.SystemAPI.Member.GetAllConfigs(sdccfg, num)
```

Return Value:
If successful, it returns 1.
Otherwise, it returns 0.

SetAllConfigs

Purpose: To overwrite all configuration profiles.

Syntax:
int SetAllConfigs (int *num*,
SDCConfig[] *config*);

Parameters:
num
[in] An integer variable that specifies the number of elements in array.
config
[in] SDCConfig structure that stores the information.

Example for C#:
```
int b1 = 0, num = 0;
Member.SDCConfig[] sdccfg = new Member.SDCConfig[20];
Cipherlab.SystemAPI.Member.GetAllConfigs(ref sdccfg, ref num);
sdccfg[0].txPower = 5;
sdccfg[1].authType = Member.AUTH.AUTH_OPEN;
b1 = Cipherlab.SystemAPI.Member.SetAllConfigs(num, sdccfg);
```

Example for VB:
```
Dim b1 As Integer
Dim num As Integer
Dim sdccfg(20) As New Member.SDCConfig
Cipherlab.SystemAPI.Member.GetAllConfigs(sdccfg, num)
sdccfg(0).txPower = 5
sdccfg(1).authType = Member.AUTH.AUTH_OPEN
b1 = Cipherlab.SystemAPI.Member.SetAllConfigs(num, sdccfg)
```

Return Value:
If successful, it returns 1.
Otherwise, it returns 0.

3.8.11. GLOBAL CONFIG SETTINGS

GetGlobalSettings	
Purpose	To retrieve the global configuration settings.
Syntax	**int GetGlobalSettings (ref SDCGlobalConfig** *configGlobal***);**
Parameters	*configGlobal* [out] SDCGlobalConfig structure that stores the information.

Example for C#	`int b1 = 0;`
	`Member.SDCglobalConfig sdcgcfg = new Member.SDCglobalConfig(1);`
	`b1 = Cipherlab.SystemAPI.Member.GetGlobalSettings(ref sdcgcfg);`

Example for VB	`Dim b1 As Integer`
	`Dim sdcgcfg As New Member.SDCglobalConfig(1)`
	`b1 = Cipherlab.SystemAPI.Member.GetGlobalSettings(sdcgcfg)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

SetGlobalSettings	
Purpose	To overwrite the global configuration settings and restart the Wi-Fi module.
Syntax	**int SetGlobalSettings (SDCGlobalConfig** *configGlobal***);**
Parameters	*configGlobal* [in] SDCGlobalConfig structure that stores the information.

Example for C#	`int b1 = 0;`
	`Member.SDCglobalConfig sdcgcfg = new Member.SDCglobalConfig(1);`
	`Cipherlab.SystemAPI.Member.GetGlobalSettings(ref sdcgcfg);`
	`sdcgcfg.displayPasswords = 1;`
	`b1 = Cipherlab.SystemAPI.Member.SetGlobalSettings(sdcgcfg);`

Example for VB	`Dim b1 As Integer`
	`Dim sdcgcfg As New Member.SDCglobalConfig(1)`
	`Cipherlab.SystemAPI.Member.GetGlobalSettings(sdcgcfg)`
	`sdcgcfg.displayPasswords = 1`
	`b1 = Cipherlab.SystemAPI.Member.SetGlobalSettings(sdcgcfg)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

3.8.12. SUMMIT REGISTRY KEYS

FlushConfigKeys

Purpose: To flush the registry keys of a configuration into the storage.

Syntax: **int FlushConfigKeys (int** *configNumber***);**

Parameters: *configNumber*

[in] A value that specifies which registry keys are to be flushed.

-1	Flush global settings
0	Flush the third party configuration **(Reserved)**
1 ~ MAX_CFGS	Flush the configuration profile by identifier

Example for C#:

```
int b1 = 0, configNumber = 0;
b1 = Cipherlab.SystemAPI.Member.FlushConfigKeys(configNumber);
```

Example for VB:

```
Dim b1 As Integer
Dim configNumber AS Integer
b1 = Cipherlab.SystemAPI.Member.FlushConfigKeys(configNumber)
```

Return Value: If successful, it returns 1.
Otherwise, it returns 0.

Remarks: This function forces a flush so the Summit parameters are saved if a power cycle occurs before the system flushes the registry.

FlushAllConfigKeys

Purpose: To flush all registry keys related to Summit configuration.

Syntax: **int FlushAllConfigKeys ();**

Example for C#:

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.FlushAllConfigKeys();
```

Example for VB:

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.FlushAllConfigKeys()
```

Return Value: If successful, it returns 1.
Otherwise, it returns 0.

Remarks: This function forces a flush so the Summit parameters are saved if a power cycle occurs before the system flushes the registry.

3.8.13. CONFIGURATION FILE

GetConfigFileInfo

Purpose	To retrieve information of a Summit configuration file.
Syntax	**int GetConfigFileInfo (string** *fileName*, **ref CONFIG_FILE_INFO** *info*);
Parameters	*fileName* [in] A string variable that stores the filename. *info* [out] structure that stores the information.

Example for C#	`int b1 = 0;`
	`string name = "setting.sdc";`
	`Member.Config_File_Info info = new Member.Config_File_Info();`
	`b1 = Cipherlab.SystemAPI.Member.GetConfigFileInfo(name, ref info);`

Example for VB	`Dim b1 As Integer`
	`Dim name AS String = "setting.sdc"`
	`Dim info As new Member.Config_File_Info`
	`b1 = Cipherlab.SystemAPI.Member.GetConfigFileInfo(name, info)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

ExportSettings

Purpose	To export configurations and global settings to a specified file.
Syntax	**int ExportSettings (string** *fileName*);
Parameters	*fileName* [in] A string variable that stores the filename.

Example for C#	`int b1 = 0;`
	`string name = "setting.sdc";`
	`b1 = Cipherlab.SystemAPI.Member.ExportSettings(name);`

Example for VB	`Dim b1 As Integer`
	`Dim name AS String = "setting.sdc"`
	`b1 = Cipherlab.SystemAPI.Member.ExportSettings(name)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

ImportSettings	
Purpose	To import configurations and global settings from a specified file.
Syntax	**int ImportSettings (string** *fileName*, **ref SDC_ALL** *all*);
Parameters	*fileName* [in] A string variable that stores the filename. *all* [out] SDC_ALL structure that stores all the settings.

Example for C#	`int b1 = 0;`
	`uint Num = 0;`
	`string name = "setting.sdc";`
	`Member.SDC_All sdccfgall = new Member.SDC_All(1);`
	`b1 = Cipherlab.SystemAPI.Member.ImportSettings(name, ref sdccfgall);`
	`Num = sdccfgall.numConfigs;`
	`for (int i = 0; i <= Num - 1; i++)`
	`{`
	`Cipherlab.SystemAPI.Member.AddConfig(sdccfgall.configs[i]);`
	`}`

Example for VB	`Dim Num As Integer`
	`Dim i As Integer`
	`Dim name AS String = "setting.sdc"`
	`Dim sdccfgall As New Member.SDC_All(1)`
	`Cipherlab.SystemAPI.Member.ImportSettings(name, sdccfgall)`
	`Num = sdccfgall.numConfigs`
	`For i = 0 To Num - 1`
	`Cipherlab.SystemAPI.Member.AddConfig(sdccfgall.configs(i))`
	`Next`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

3.8.14. WEP KEY

GetWEPKey

Purpose

To retrieve a WEP key.

Syntax

int GetWEPKey (SDCConfig *config*,
int *keyIndex*,
ref WEPLEN *keyLength*,
ref string *wepKey*,
ref int *keyState*);

Parameters

config

[in] SDCConfig structure that stores the information.

keyIndex

[in] Integer variable

1 ~ 4	Index of the WEP key to be retrieved

keyLength

[out] A value that specifies the key length; if NULL, this parameter is ignored.

0	WEPLEN_NOT_SET	(Clear the key.)
1	WEPLEN_40BIT	(The key must be 10 hex characters.)
2	WEPLEN_128BIT	(The key must be 26 hex characters.)

wepKey

[out] Pointer to a buffer where the WEP key is stored. Buffer size must be greater than 26 bytes.

keyState

[out] An ingeger variable that stores the key state.

0	The key is not activated
1	The key is activated

Example for C#

```
int b1 = 0, WepKey = 1, txKey = 0;
string name = “AP”, key = “”;
Member.WEPLEN keyLength = 0;
Member.SDCConfig sdccfg = new Member.SDCConfig(1);
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.GetWEPKey(
sdccfg, WepKey, ref keyLength, ref key, ref txKey);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig(1)
Dim WepKey As Integer = 1
Dim txKey As Integer = 0
Dim name As String = “AP”
Dim key As String = “”
Dim keyLength As New Member.WEPLEN
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.GetWEPKey(
sdccfg, WepKey, keyLength, key, txKey)
```

Return Value	If successful, it returns 1. Otherwise, it returns 0.

SetWEPKey

Purpose	To set a WEP key.
Syntax	**int SetWEPKey (ref SDCConfig** *config*, **int** *keyIndex*, **WEPLEN** *keyLength*, **string** *wepKey*, **int** *keyState*);
Parameters	*config* [in] SDCConfig structure that stores the information. *keyIndex* [in] Integer variable

1~4	Index of the WEP key to be set

keyLength
[in] A value that specifies the key length.

0	WEPLEN_NOT_SET	(Clear the key.)
1	WEPLEN_40BIT	(The key must be 10 hex characters.)
2	WEPLEN_128BIT	(The key must be 26 hex characters.)

wepKey
[in] A string variable that stores the WEP key.
keyState
[in] An ingeger variable that stores the key state.

0	Do not activate the key
1	Activate the key

Example for C#

```
int b1 = 0, WepKey = 1, txKey = 1;
string name = “AP”, key = “abcde”;
Member.WEPLEN keyLength =  = Member.WEPLEN.WEPLEN_40BIT;
Member.SDCConfig sdccfg = new Member.SDCConfig();
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.SetWEPKey(
sdccfg, WepKey, keyLength, key, txKey);
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig
Dim WepKey As Integer = 1
Dim txKey As Integer = 1
Dim name As String = “AP”
Dim key As String = “abcde”
Dim keyLength As Member.WEPLEN = Member.WEPLEN.WEPLEN_40BIT
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.SetWEPKey(
```

Example for VB
```
sdccfg, WepKey, keyLength, key, txKey)
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg)
```

Return Value
If successful, it returns 1.
Otherwise, it returns 0.

GetMultipleWEPKeys

Purpose
To retrieve all four WEP keys.

Syntax
int GetMultipleWEPKey (SDCConfig *config*,
ref int *keyIndex*,
ref WEPLEN *keyLength1*,
ref string *wepKey1*,
ref WEPLEN *keyLength2*,
ref string *wepKey2*,
ref WEPLEN *keyLength3*,
ref string *wepKey3*,
ref WEPLEN *keyLength4*,
ref string *wepKey4*);

Parameters
config
[in] SDCConfig structure that stores the information.
keyIndex
[out] Integer variable

1~4	Index of the active WEP key

keyLength1(~4)
[out] A value that specifies the key length.

0	WEPLEN_NOT_SET	(Clear the key.)
1	WEPLEN_40BIT	(The key must be 10 hex characters.)
2	WEPLEN_128BIT	(The key must be 26 hex characters.)

wepKey1(~4)
[out] Pointer to a buffer where the WEP key 1(~4) is stored. Buffer size must be greater than 26 bytes.

Example for C#
```
int b1 = 0, txKey = 1;
string name = "AP", key1 = "", key2 = "", key3 = "", key4 = "";
Member.WEPLEN keyLength1 = 0, keyLength2 = 0, keyLength3 = 0, keyLength4
= 0;
Member.SDCConfig sdccfg = new Member.SDCConfig(1);
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.GetMultipleWEPKeys(
sdccfg, ref txKey, ref keyLength1, ref key1, ref keyLength2, ref key2,
ref keyLength3, ref key3, ref keyLength4, ref key4);
```

Example for VB
```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig(1)
Dim txKey As Integer = 1
Dim name As String = "AP"
Dim key1, key2, key3, key4 As String
```

Example for VB

```
Dim keyLength1, keyLength2, keyLength3, keyLength4 As New Member.WEPLEN
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.GetMultipleWEPKeys(
sdccfg, txKey, keyLength1, key1, keyLength2, key2, keyLength3, key3, keyLength4, key4)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

SetMultipleWEPKeys

Purpose

To set all four WEP keys.

Syntax

int SetMultipleWEPKey (ref SDCConfig *config*,
int *keyIndex*,
WEPLEN *keyLength1*,
string *wepKey1*,
WEPLEN *keyLength2*,
string *wepKey2*,
WEPLEN *keyLength3*,
string *wepKey3*,
WEPLEN *keyLength4*,
string *wepKey4*);

Parameters

config

[in] SDCConfig structure that stores the information.

keyIndex

[in] Integer variable

1~4	Index of the active WEP key

keyLength1(~4)

[in] A value that specifies the key length.

0	WEPLEN_NOT_SET	(Clear the key.)
1	WEPLEN_40BIT	(The key must be 10 hex characters.)
2	WEPLEN_128BIT	(The key must be 26 hex characters.)

wepKey1(~4)

[in] A string variable that stores the WEP key 1(~4).

Example for C#

```
int b1 = 0, txKey = 1;
string name = “AP”, key1 = “abcde”, key2 = “12345”, key3 = “thisisawepkey”, key4 = “xyz12”;
Member.WEPLEN keyLength1 = Member.WEPLEN.WEPLEN_40BIT, keyLength2 = Member.WEPLEN.WEPLEN_40BIT, keyLength3 =Member.WEPLEN.WEPLEN_128BIT, keyLength4 = Member.WEPLEN.WEPLEN_40BIT;
Member.SDCConfig sdccfg = new Member.SDCConfig();
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.SetMultipleWEPKeys(
sdccfg, txKey, keyLength1, key1, keyLength2, key2, keyLength3, key3, keyLength4, key4);
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig
Dim txKey As Integer = 1
Dim name As String = “AP”
Dim key1 As String = “abcde”
Dim key2 As String = “12345”
Dim key3 As String = “thisisawepkey”
Dim key4 As String = “xyz12”
Dim keyLength1 As Member.WEPLEN = Member.WEPLEN.WEPLEN_40BIT
Dim keyLength2 As Member.WEPLEN = Member.WEPLEN.WEPLEN_40BIT
Dim keyLength3 As Member.WEPLEN = Member.WEPLEN.WEPLEN_128BIT
Dim keyLength4 As Member.WEPLEN = Member.WEPLEN.WEPLEN_40BIT
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.SetMultipleWEPKeys(
sdccfg, txKey, keyLength1, key1, keyLength2, key2, keyLength3, key3,
keyLength4, key4)
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

3.8.15. PRE-SHARED KEY

GetPSK

Purpose: To retrieve the Pre-Shared Key.

Syntax:
int GetPSK (SDCConfig *config*,
ref string *psk*);

Parameters:
config
[in] SDCConfig structure that stores the information.
psk
[out] Pointer to a buffer where the Pre-Shared Key is stored. Buffer size must be greater than PSK_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

Example for C#:
```
int b1 = 0;
string name = "AP", psk = "";
Member.SDCConfig sdccfg = new Member.SDCConfig(1);
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.GetPSK(sdccfg, ref psk);
```

Example for VB:
```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig(1)
Dim name As String = "AP"
Dim psk As String = ""
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.GetPSK(sdccfg, psk)
```

Return Value:
If successful, it returns 1.
Otherwise, it returns 0.

SetPSK

Purpose: To set the Pre-Shared Key.

Syntax:
int SetPSK (ref SDCConfig *config*,
string *psk*);

Parameters:
pConfig
[in] SDCConfig structure that stores the information.
psk
[in] A string variable that stores the Pre-Shared Key. Its value is a null-terminated string, and will be cleared when "" is passed to the argument. The string length must be no greater than PSK_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

- For PSK, the string must be 64 hex characters.

For Passphrase, the string must be 8~63 printable ASCII characters.

Example for C#:
```
int b1 = 0;
string name = "AP", psk = "abc";
Member.SDCConfig sdccfg = new Member.SDCConfig();
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.SetPSK(sdccfg, psk);
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig
Dim name As String = “AP”
Dim psk As String = “abc”
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.SetPSK(sdccfg, psk)
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

3.8.16. LEAP CREDENTIALS

GetLEAPCred	
Purpose	To retrieve the LEAP credentials.
Syntax	**int GetLEAPCred (SDCConfig** *config*, **ref string** *username*, **ref string** *password*);
Parameters	*config* [in] SDCConfig structure that stores the information. *username* [out] Pointer to a buffer where the username is stored. Buffer size must be greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member. *password* [out] Pointer to a buffer where the password is stored. Buffer size must be greater than USER_PWD_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

Example for C#	`int b1 = 0;`
	`string name = "AP", username = "", password = "";`
	`Member.SDCConfig sdccfg = new Member.SDCConfig(1);`
	`Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);`
	`b1 = Cipherlab.SystemAPI.Member.GetLEAPCred(`
	`sdccfg, ref username, ref password);`

Example for VB	`Dim b1 As Integer`
	`Dim sdccfg As New Member.SDCConfig(1)`
	`Dim name As String = "AP"`
	`Dim username, password As String = ""`
	`Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)`
	`b1 = Cipherlab.SystemAPI.Member.GetLEAPCred(`
	`sdccfg, username, password)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

SetLEAPCred	
Purpose	To set the LEAP credentials.
Syntax	**int SetLEAPCred (ref SDCConfig** *config***,** **string** *username***,** **string** *password***);**
Parameters	*config* [in] SDCConfig structure that stores the information. *username* [in] A string variable that stores the username. Its value is a null-terminated string, and will be cleared when "" is passed to the argument. The string length must be no greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member. *password* [in] A string variable that stores the password. Its value is a null-terminated string, and will be cleared when "" is passed to the argument. The string length must be no greater than USER_PWD_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

Example for C#	`int b1 = 0;`
	`string name = "AP", username = "abc", password = "123";`
	`Member.SDCConfig sdccfg = new Member.SDCConfig();`
	`Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);`
	`b1 = Cipherlab.SystemAPI.Member.SetLEAPCred(`
	`sdccfg, username, password);`
	`Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg);`

Example for VB	`Dim b1 As Integer`
	`Dim sdccfg As New Member.SDCConfig`
	`Dim name As String = "AP"`
	`Dim username As String = "abc"`
	`Dim password As String = "123"`
	`Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)`
	`b1 = Cipherlab.SystemAPI.Member.SetLEAPCred(`
	`sdccfg, username, password)`
	`Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

3.8.17. EAP-FAST CREDENTIALS

GetEAPFASTCred

Purpose	To retrieve the EAP-FAST credentials.
Syntax	**int GetEAPFASTCred (SDCConfig** *config*, **ref string** *username*, **ref string** *password*, **ref string** *pacFileName*, **ref string** *pacPassword*);
Parameters	*config* [in] SDCConfig structure that stores the information. *username* [out] Pointer to a buffer where the username is stored. Buffer size must be greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member. *password* [out] Pointer to a buffer where the password is stored. Buffer size must be greater than USER_PWD_SZ (bytes) defined in Cipherlab.SystemAPI.Member. *pacFileName* [out] Pointer to a buffer where the PAC filename is stored. Buffer size must be greater than CRED_PFILE_SZ (bytes) defined in Cipherlab.SystemAPI.Member. *pacPassword* [out] Pointer to a buffer where the PAC password is stored. Buffer size must be greater than CRED_PFILE_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

Example for C#	`int b1 = 0;`
	`string name = "AP", username = "", password = "", pacfilename = "", pacpassword = "";`
	`Member.SDCConfig sdccfg = new Member.SDCConfig(1);`
	`Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);`
	`b1 = Cipherlab.SystemAPI.Member.GetEAPFASTCred(`
	`sdccfg, ref username, ref password, ref pacfilename, ref pacpassword);`

Example for VB	`Dim b1 As Integer`
	`Dim sdccfg As New Member.SDCConfig(1)`
	`Dim name As String = "AP"`
	`Dim username, password, pacfilename, pacpassword As String`
	`Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)`
	`b1 = Cipherlab.SystemAPI.Member.GetEAPFASTCred(`
	`sdccfg, username, password, pacfilename, pacpassword)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

SetEAPFASTCred

Purpose	To set the EAP-FAST credentials.
Syntax	**int SetEAPFASTCred (ref SDCConfig** *config*,

string *username*,
string *password*,
string *pacFileName*,
string *pacPassword*);

Parameters

config

[in] SDCConfig structure that stores the information.

username

[in] A string variable that stores the username. Its value is a null-terminated string, and will be cleared when "" is passed to the argument. The string length must be no greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

password

[in] A string variable that stores the password. Its value is a null-terminated string, and will be cleared when "" is passed to the argument. The string length must be no greater than USER_PWD_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

pacFileName

[in] A string variable that stores the PAC filename. Its value is a null-terminated string, and will be cleared when "" is passed to the argument. The string length must be no greater than CRED_PFILE_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

pacPassword

[in] A string variable that stores the PAC password. Its value is a null-terminated string, and will be cleared when "" is passed to the argument. The string length must be no greater than CRED_PFILE_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

Example for C#

```
int b1 = 0;
string name = "AP", username = "abc", password = "123", pacfilename = "xyz", pacpassword = "456";
Member.SDCConfig sdccfg = new Member.SDCConfig();
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.SetEAPFASTCred(
sdccfg, username, password, pacfilename, pacpassword);
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig
Dim name As String = "AP"
Dim username As String = "abc"
Dim password As String = "123"
Dim pacfilename As String = "xyz"
Dim pacpassword As String = "456"
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.SetEAPFASTCred(
sdccfg, username, password, pacfilename, pacpassword)
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

3.8.18. PEAP-GTC CREDENTIALS

GetPEAPGTCCred

Purpose: To retrieve the PEAP-GTC credentials.

Syntax:

```
int GetPEAPGTCCred (SDCConfig config,
                    ref string username,
                    ref string password,
                    ref CERTLOCATION certLocation,
                    ref string caCert);
```

Parameters:

config

[in] SDCConfig structure that stores the information.

username

[out] Pointer to a buffer where the username is stored. Buffer size must be greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member. Pass "" to ignore this parameter.

password

[out] Pointer to a buffer where the password is stored. Buffer size must be greater than USER_PWD_SZ (bytes) defined in Cipherlab.SystemAPI.Member. Pass "" to ignore this parameter.

certLocation

[out] A value that specifies whether to use a CA certificate to validate an authentication server. Pass 0 to ignore this parameter.

cerLocation		caCert
0	CERT_NONE	"caCert" is set to "". Don't validate the server.
1	CERT_FILE	"caCert" is the filename of root certificate authority (CA) digital certificate; string length must be no greater than CRED_CERT_SZ (bytes).
2	CERT_FULL_STORE	"caCert" is set to "". It will search for a valid certificate from the entire Microsoft certificate store.
3	CERT_IN_STORE	"caCert" is a 20-byte hash value representing one specific certificate from the Microsoft certificate store.

caCert

[out] Pointer to a buffer where CA certificate is stored. Buffer size must be greater than CRED_CERT_SZ (bytes) defined in Cipherlab.SystemAPI.Member. CA certificate depends on the value of "certLocation" above. Pass "" to ignore this parameter.

Example for C#

```
int b1 = 0;
string name = "AP", username = "", password = "", caCert = "";
Member.CERTLOCATION certLocation = new Member.CERTLOCATION();
Member.SDCConfig sdccfg = new Member.SDCConfig(1);
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.GetPEAPGTCCred(
sdccfg, ref username, ref password, ref certLocation, ref caCert);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig(1)
Dim name As String = “AP”
Dim username, password, caCert As String
Dim certLocation As New Member.CERTLOCATION
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.GetPEAPGTCCred(
sdccfg, username, password, certLocation, caCert)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

SetPEAPGTCCred

Purpose

To set the PEAP-GTC credentials.

Syntax

int SetPEAPGTCCred (ref SDCConfig *config*,
string *username*,
string *password*,
CERTLOCATION *certLocation*,
string *caCert*);

Parameters

config

[in] SDCConfig structure that stores the information.

username

[in] A string variable that stores the username. Its value is a null-terminated string, and will be cleared when “” is passed to the argument. The string length must be no greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

password

[in] A string variable that stores the password. Its value is a null-terminated string, and will be cleared when “” is passed to the argument. The string length must be no greater than USER_PWD_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

certLocation

[in] A value that specifies whether to use a CA certificate to validate an authentication server.

cerLocation		caCert
0	CERT_NONE	“caCert” is set to “”. Don’t validate the server.
1	CERT_FILE	“caCert” is the filename of root certificate authority (CA) digital certificate; string length must be no greater than CRED_CERT_SZ (bytes).
2	CERT_FULL_STORE	“caCert” is set to “”. It will search for a valid certificate from the entire Microsoft certificate store.
3	CERT_IN_STORE	“caCert” is a 20-byte hash value representing one specific certificate from the Microsoft certificate store.

	caCert [in] CA certificate depends on the value of “certLocation” above. Pass “” to ignore this parameter.

Example for C#	`int b1 = 0;`
	`string name = “AP”, username = “abc”, password = “123”, caCert = “xyz”;`
	`Member.CERTLOCATION certLocation = Member.CERTLOCATION.CERT_FILE;`
	`Member.SDCConfig sdccfg = new Member.SDCConfig();`
	`Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);`
	`b1 = Cipherlab.SystemAPI.Member.SetPEAPGTCCred(`
	`sdccfg, username, password, certLocation, caCert);`
	`Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg);`

Example for VB	`Dim b1 As Integer`
	`Dim sdccfg As New Member.SDCConfig`
	`Dim name As String = “AP”`
	`Dim username As String = “abc”`
	`Dim password As String = “123”`
	`Dim caCert As String = “xyz”`
	`Dim certLocation As Member.CERTLOCATION =`
	`Member.CERTLOCATION.CERT_FILE`
	`Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)`
	`b1 = Cipherlab.SystemAPI.Member.SetPEAPGTCCred(`
	`sdccfg, username, password, certLocation, caCert)`
	`Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.

3.8.19. PEAP-MSCHAP CREDENTIALS

GetPEAPMSCHAPCred

Purpose: To retrieve the PEAP-MSCHAP credentials.

Syntax:

```
int GetPEAPMSCHAPCred (SDCConfig config,
                       ref string username,
                       ref string password,
                       ref CERTLOCATION certLocation,
                       ref string caCert);
```

Parameters

config

[in] SDCConfig structure that stores the information.

username

[out] Pointer to a buffer where the username is stored. Buffer size must be greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member. Pass "" to ignore this parameter.

password

[out] Pointer to a buffer where the password is stored. Buffer size must be greater than USER_PWD_SZ (bytes) defined in Cipherlab.SystemAPI.Member. Pass "" to ignore this parameter.

certLocation

[out] A value that specifies whether to use a CA certificate to validate an authentication server. Pass 0 to ignore this parameter.

cerLocation		caCert
0	CERT_NONE	"caCert" is set to "". Don't validate the server.
1	CERT_FILE	"caCert" is the filename of root certificate authority (CA) digital certificate; string length must be no greater than CRED_CERT_SZ (bytes).
2	CERT_FULL_STORE	"caCert" is set to "". It will search for a valid certificate from the entire Microsoft certificate store.
3	CERT_IN_STORE	"caCert" is a 20-byte hash value representing one specific certificate from the Microsoft certificate store.

caCert

[out] Pointer to a buffer where CA certificate is stored. Buffer size must be greater than CRED_CERT_SZ (bytes) defined in Cipherlab.SystemAPI.Member. CA certificate depends on the value of "certLocation" above. Pass "" to ignore this parameter.

Example for C#

```
int b1 = 0;
string name = "AP", username = "", password = "", caCert = "";
Member.CERTLOCATION certLocation = new Member.CERTLOCATION();
Member.SDCConfig sdccfg = new Member.SDCConfig(1);
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.GetPEAPMSCHAPCred(
sdccfg, ref username, ref password, ref certLocation, ref caCert);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig(1)
Dim name As String = “AP”
Dim username, password, caCert As String
Dim certLocation As New Member.CERTLOCATION
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.GetPEAPMSCHAPCred(s
dccfg, username, password, certLocation, caCert)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

SetPEAPMSCHAPCred

Purpose

To set the PEAP-MSCHAP credentials.

Syntax

int SetPEAPMSCHAPCred (ref SDCConfig *config*,
string *username*,
string *password*,
CERTLOCATION *certLocation*,
string *caCert*);

Parameters

config

[in] SDCConfig structure that stores the information.

username

[in] A string variable that stores the username. Its value is a null-terminated string, and will be cleared when “” is passed to the argument. The string length must be no greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

password

[in] A string variable that stores the password. Its value is a null-terminated string, and will be cleared when “” is passed to the argument. The string length must be no greater than USER_PWD_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

certLocation

[in] A value that specifies whether to use a CA certificate to validate an authentication server.

cerLocation		caCert
0	CERT_NONE	“caCert” is set to “”. Don’t validate the server.
1	CERT_FILE	“caCert” is the filename of root certificate authority (CA) digital certificate; string length must be no greater than CRED_CERT_SZ (bytes).
2	CERT_FULL_STORE	“caCert” is set to “”. It will search for a valid certificate from the entire Microsoft certificate store.
3	CERT_IN_STORE	“caCert” is a 20-byte hash value representing one specific certificate from the Microsoft certificate store.

caCert

[in] CA certificate depends on the value of “certLocation” above. Pass “” to ignore this parameter.

Example for C#

```
int b1 = 0;
string name = “AP”, username = “abc”, password = “123”, caCert = “xyz”;
Member.CERTLOCATION certLocation = Member.CERTLOCATION.CERT_FILE;
Member.SDCConfig sdccfg = new Member.SDCConfig();
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.SetPEAPMSCHAPCred(
sdccfg, username, password, certLocation, caCert);
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig
Dim name As String = “AP”
Dim username As String = “abc”
Dim password As String = “123”
Dim caCert As String = “xyz”
Dim certLocation As Member.CERTLOCATION =
Member.CERTLOCATION.CERT_FILE
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.SetPEAPMSCHAPCred(
sdccfg, username, password, certLocation, caCert)
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

3.8.20. EAP-TLS CREDENTIALS

GetEAPTLSCred

Purpose: To retrieve the EAP-TLS credentials.

Syntax:

int GetEAPTLSCred (SDCConfig *config*,
ref string *username*,
ref string *userCert*,
ref CERTLOCATION *certLocation*,
ref string *caCert*);

Parameters:

pConfig

[in] SDCConfig structure that stores the information.

username

[out] Pointer to a buffer where the username is stored. Buffer size must be greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member. Pass "" to ignore this parameter.

userCert

[out] Pointer to a buffer where the user certificate is stored. Buffer size must be greater than 20 bytes. "userCert" is a 20-byte hash value representing one specific certificate from the Microsoft certificate store. Pass "" to ignore this parameter.

certLocation

[out] A value that specifies whether to use a CA certificate to validate an authentication server. Pass 0 to ignore this parameter.

cerLocation		caCert
0	CERT_NONE	"caCert" is set to "". Don't validate the server.
1	CERT_FILE	"caCert" is the filename of root certificate authority (CA) digital certificate; string length must be no greater than CRED_CERT_SZ (bytes).
2	CERT_FULL_STORE	"caCert" is set to "". It will search for a valid certificate from the entire Microsoft certificate store.
3	CERT_IN_STORE	"caCert" is a 20-byte hash value representing one specific certificate from the Microsoft certificate store.

caCert

[out] Pointer to a buffer where CA certificate is stored. Buffer size must be greater than CRED_CERT_SZ (bytes) defined in Cipherlab.SystemAPI.Member. CA certificate depends on the value of "certLocation" above. Pass "" to ignore this parameter.

Example for C#

```
int b1 = 0;
string name = "AP", username = "", password = "", caCert = "";
Member.CERTLOCATION certLocation = new Member.CERTLOCATION();
Member.SDCConfig sdccfg = new Member.SDCConfig(1);
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.GetEAPTLSCred(
sdccfg, ref username, ref password, ref certLocation, ref caCert);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig(1)
Dim name As String = "AP"
Dim username, password, caCert As String
Dim certLocation As New Member.CERTLOCATION
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.GetEAPTLSCred(
sdccfg, username, password, certLocation, caCert)
```

Return Value

If successful, it returns 1.
Otherwise, it returns 0.

SetEAPTLSCred

Purpose

To set the EAP-TLS credentials.

Syntax

int SetEAPTLSCred (ref SDCConfig *config*,
string *username*,
string *userCert*,
CERTLOCATION *certLocation*,
string *caCert*);

Parameters

config

[in] SDCConfig structure that stores the information.

username

[in] A string variable that stores the username. Its value is a null-terminated string, and will be cleared when "" is passed to the argument. The string length must be no greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

userCert

[in] A string variable that stores the password. Its value is a null-terminated string, and will be cleared when "" is passed to the argument. The string length must be no greater than USER_PWD_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

certLocation

[in] A value that specifies whether to use a CA certificate to validate an authentication server.

cerLocation		caCert
0	CERT_NONE	“caCert” is set to “”. Don’t validate the server.
1	CERT_FILE	“caCert” is the filename of root certificate authority (CA) digital certificate; string length must be no greater than CRED_CERT_SZ (bytes).
2	CERT_FULL_STORE	“caCert” is set to “”. It will search for a valid certificate from the entire Microsoft certificate store.
3	CERT_IN_STORE	“caCert” is a 20-byte hash value representing one specific certificate from the Microsoft certificate store.

caCert

[in] CA certificate depends on the value of “certLocation” above. Pass “” to ignore this parameter.

Example for C#

```
int b1 = 0;
string name = “AP”, username = “abc”, password = “123”, caCert = “xyz”;
Member.CERTLOCATION certLocation = Member.CERTLOCATION.CERT_FILE;
Member.SDCConfig sdccfg = new Member.SDCConfig();
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.SetEAPTLSCred(
sdccfg, username, password, certLocation, caCert);
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig
Dim name As String = “AP”
Dim username As String = “abc”
Dim password As String = “123”
Dim caCert As String = “xyz”
Dim certLocation As Member.CERTLOCATION =
Member.CERTLOCATION.CERT_FILE
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.SetEAPTLSCred(
sdccfg, username, password, certLocation, caCert)
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

3.8.21. EAP-TTLS CREDENTIALS

GetEAPTTLSCred

Purpose: To retrieve the EAP-TTLS credentials.

Syntax:

```
int GetEAPTTLSCred (SDCConfig config,
                    ref string username,
                    ref string password,
                    ref CERTLOCATION certLocation,
                    ref string caCert);
```

Parameters:

config

[in] SDCConfig structure that stores the information.

username

[out] Pointer to a buffer where the username is stored. Buffer size must be greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member. Pass "" to ignore this parameter.

password

[out] Pointer to a buffer where the password is stored. Buffer size must be greater than USER_PWD_SZ (bytes) defined in Cipherlab.SystemAPI.Member. Pass "" to ignore this parameter.

certLocation

[out] A value that specifies whether to use a CA certificate to validate an authentication server. Pass 0 to ignore this parameter.

cerLocation		caCert
0	CERT_NONE	"caCert" is set to "". Don't validate the server.
1	CERT_FILE	"caCert" is the filename of root certificate authority (CA) digital certificate; string length must be no greater than CRED_CERT_SZ (bytes).
2	CERT_FULL_STORE	"caCert" is set to "". It will search for a valid certificate from the entire Microsoft certificate store.
3	CERT_IN_STORE	"caCert" is a 20-byte hash value representing one specific certificate from the Microsoft certificate store.

caCert

[out] Pointer to a buffer where CA certificate is stored. Buffer size must be greater than CRED_CERT_SZ (bytes) defined in Cipherlab.SystemAPI.Member. CA certificate depends on the value of "certLocation" above. Pass "" to ignore this parameter.

Example for C#

```
int b1 = 0;
string name = "AP", username = "", password = "", caCert = "";
Member.CERTLOCATION certLocation = new Member.CERTLOCATION();
Member.SDCConfig sdccfg = new Member.SDCConfig(1);
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.GetEAPTTLSCred(
sdccfg, ref username, ref password, ref certLocation, ref caCert);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig(1)
Dim name As String = “AP”
Dim username, password, caCert As String
Dim certLocation As New Member.CERTLOCATION
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.GetEAPTTLSCred(
sdccfg, username, password, certLocation, caCert)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

SetEAPTTLSCred

Purpose

To set the EAP-TTLS credentials.

Syntax

int SetEAPTTLSCred (ref SDCConfig *config*,
string *username*,
string *password*,
CERTLOCATION *certLocation*,
string *caCert*);

Parameters

config

[in] SDCConfig structure that stores the information.

username

[in] A string variable that stores the username. Its value is a null-terminated string, and will be cleared when “” is passed to the argument. The string length must be no greater than USER_NAME_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

password

[in] A string variable that stores the password. Its value is a null-terminated string, and will be cleared when “” is passed to the argument. The string length must be no greater than USER_PWD_SZ (bytes) defined in Cipherlab.SystemAPI.Member.

certLocation

[in] A value that specifies whether to use a CA certificate to validate an authentication server.

cerLocation		caCert
0	CERT_NONE	"caCert" is set to "". Don't validate the server.
1	CERT_FILE	"caCert" is the filename of root certificate authority (CA) digital certificate; string length must be no greater than CRED_CERT_SZ (bytes).
2	CERT_FULL_STORE	"caCert" is set to "". It will search for a valid certificate from the entire Microsoft certificate store.
3	CERT_IN_STORE	"caCert" is a 20-byte hash value representing one specific certificate from the Microsoft certificate store.

caCert

[in] CA certificate depends on the value of "certLocation" above. Pass "" to ignore this parameter.

Example for C#

```
int b1 = 0;
string name = "AP", username = "abc", password = "123", caCert = "xyz";
Member.CERTLOCATION certLocation = Member.CERTLOCATION.CERT_FILE;
Member.SDCConfig sdccfg = new Member.SDCConfig();
Cipherlab.SystemAPI.Member.GetConfig(name, ref sdccfg);
b1 = Cipherlab.SystemAPI.Member.SetEAPTTLSCred(
sdccfg, username, password, certLocation, caCert);
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg);
```

Example for VB

```
Dim b1 As Integer
Dim sdccfg As New Member.SDCConfig
Dim name As String = "AP"
Dim username As String = "abc"
Dim password As String = "123"
Dim caCert As String = "xyz"
Dim certLocation As Member.CERTLOCATION =
Member.CERTLOCATION.CERT_FILE
Cipherlab.SystemAPI.Member.GetConfig(name, sdccfg)
b1 = Cipherlab.SystemAPI.Member.SetEAPTTLSCred(
sdccfg, username, password, certLocation, caCert)
Cipherlab.SystemAPI.Member.ModifyConfig(name, sdccfg)
```

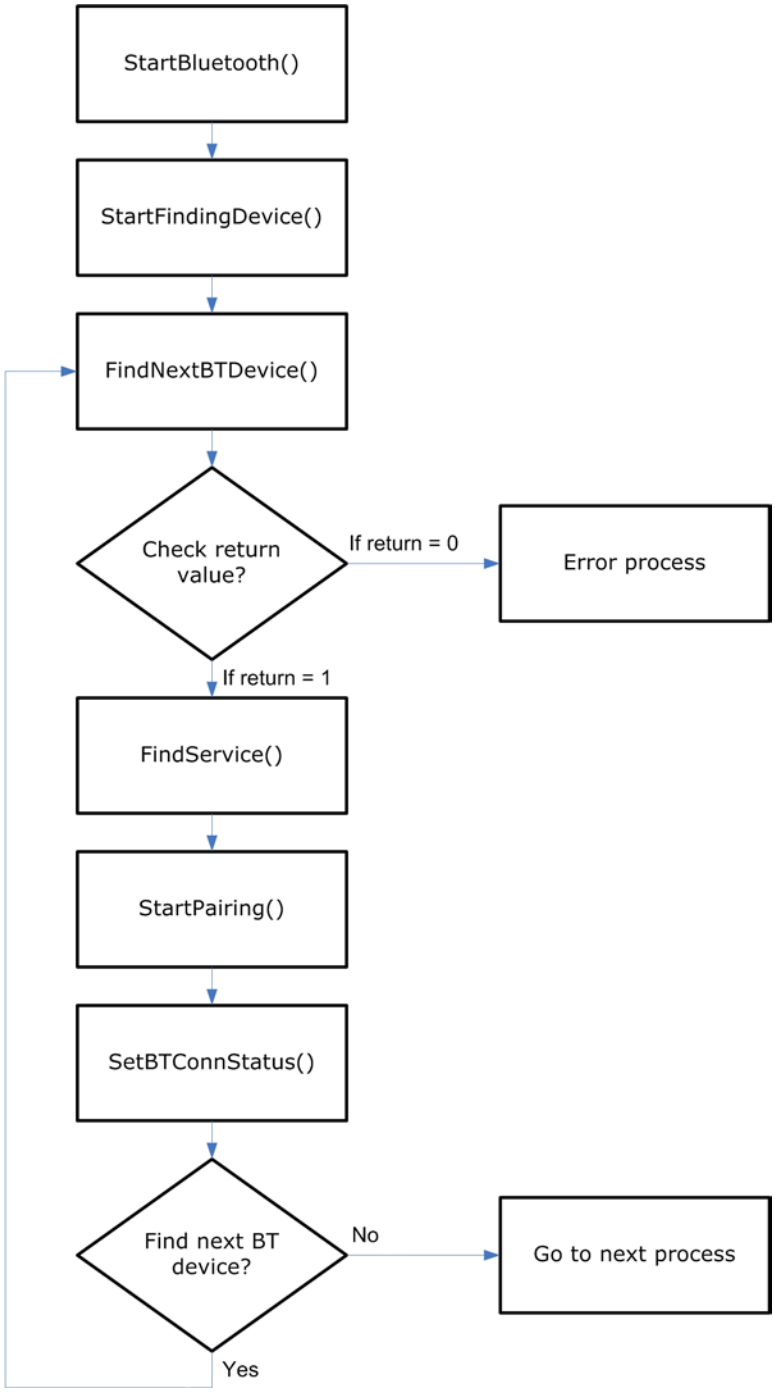
Return Value

If successful, it returns 1.

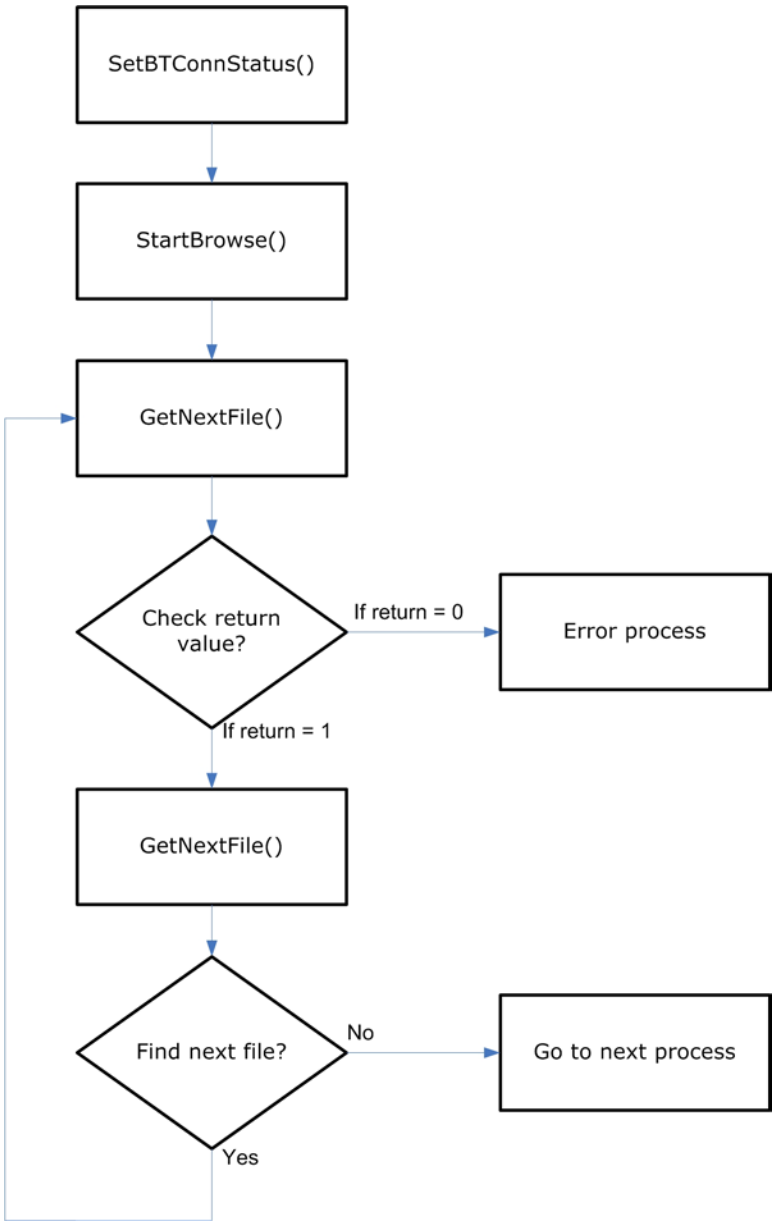
Otherwise, it returns 0.

3.9. BLUETOOTH

Here is the programming flow for Bluetooth initialization.



After initialization, you may start with a desired Bluetooth service by calling SetBTConnStatus(). Below is the programming flow for using the File Transfer service (FTP).



3.9.1. START/STOP BLUETOOTH

StartBluetooth	
Purpose	To initialize the Bluetooth module.
Syntax	**int StartBluetooth ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.StartBluetooth();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.StartBluetooth()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered: 32 ERROR_OPERATION_FAIL 33 ERROR_MACHINE_NOT_SUPPORTED
See Also	SetBTAudioState

StopBluetooth	
Purpose	To stop the Bluetooth module.
Syntax	**int StopBluetooth ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.StopBluetooth();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.StopBluetooth()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered: 32 ERROR_OPERATION_FAIL

3.9.2. FIND DEVICES

StartFindingDevice

Purpose	To initialize the search procedure.
Syntax	**int StartFindingDevice ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.StartFindingDevice();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.StartFindingDevice()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

32	ERROR_OPERATION_FAIL

FindNextBTDevice

Purpose	To search Bluetooth devices after StartFindingDevice() is called.
Syntax	**int FindNextBTDevice (ref int** *deviceInfo***);**
Parameters	*deviceInfo* [out] An integer variable that stores the device information.

Example for C#	`int b1 = 0;`
	`int deviceInfo = 0;`
	`b1 = Cipherlab.SystemAPI.Member.FindNextBTDevice(ref deviceInfo);`

Example for VB	`Dim b1 As Integer`
	`Dim deviceInfo As Integer`
	`b1 = Cipherlab.SystemAPI.Member.FindNextBTDevice(deviceInfo)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

30	ERROR_NO_STARTFINDING
31	ERROR_WRONG_ARG
32	ERROR_OPERATION_FAIL

3.9.3. PAIR DEVICES

StartPairing

Purpose	To pair with a □uttoned□Bluetooth device.
Syntax	**int StartPairing (int** *deviceInfo*, **string** *pinCode*);
Parameters	*deviceInfo* [in] A value returned by FindNextBTDevice(). *pinCode* [in] A string variable that stores the PIN code.

Example for C#	`int b1 = 0;`
	`string pinCode = "1234";`
	`b1 = Cipherlab.SystemAPI.Member.StartPairing(deviceInfo, pinCode);`

Example for VB	`Dim b1 As Integer`
	`Dim pinCode As String = "1234"`
	`b1 = Cipherlab.SystemAPI.Member. StartPairing(deviceInfo, pinCode)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

4	ERROR_PARAMETER
21	ERROR_NO_DEVICE_INFO
25	ERROR_NOT_DEVICEINFO_OBJ
32	ERROR_OPERATION_FAIL

See Also FindNextBTDevice

StartUnpairing

Purpose	To unpair with a □uttoned□Bluetooth device.
Syntax	**int StartPairing (int** *deviceInfo*);
Parameters	*deviceInfo* [in] A value returned by FindNextBTDevice().
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.StartUnpairing(deviceInfo);`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.StartUnpairing(deviceInfo)`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

21	ERROR_NO_DEVICE_INFO
25	ERROR_NOT_DEVICEINFO_OBJ
32	ERROR_OPERATION_FAIL

3.9.4. AVAILABLE SERVICES & DEVICE INFORMATION

FindService

Purpose: To find out Bluetooth services available on the connected device.

Syntax:

```
int FindService (int targetService,
                 int deviceInfo,
                 ref int serviceInfo);
```

Parameters:

targetService

[in] A value that specifies which Bluetooth service is inquired.

1	BT_HID_SERVICE	
2	BT_DUN_SERVICE	COM 0 (client)
3	BT_SPP_SERVICE	COM 0 (client)
4	BT_OPP_SERVICE	
5	(Reserved)	
6	BT_HEADSET_SERVICE	
7	BT_HANDSFREE_SERVICE	
8	BT_FTP_SERVICE	

deviceInfo

[in] A value returned by FindNextBTDevice().

serviceInfo

[out] An integer variable that stores the information.

Example for C#:

```
int b1 = 0;
int serviceInfo = 0;
b1 = Cipherlab.SystemAPI.Member.FindService(
3, deviceInfo, ref serviceInfo);
```

Example for VB:

```
Dim b1 As Integer
Dim serviceInfo As Integer
b1 = Cipherlab.SystemAPI.Member.FindService(
3, deviceInfo, serviceInfo)
```

Return Value: If successful, it returns 1.
Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

21	ERROR_NO_DEVICE_INFO
22	ERROR_NO_SERVICE_INFO
25	ERROR_NOT_DEVICEINFO_OBJ
32	ERROR_OPERATION_FAIL

See Also: GetServiceType

GetDeviceAddress

Purpose	To get the MAC address of the connected device.
Syntax	**int GetDeviceAddress (int** *deviceInfo***,** **ref ulong** *btAddr***);**
Parameters	*deviceInfo* [in] A value returned by FindNextBTDevice(). *btAddr* [out] An integer variable that stores the MAC address.

Example for C#	`int b1 = 0;`
	`ulong btAddr = 0;`
	`b1 = Cipherlab.SystemAPI.Member.GetDeviceAddress(`
	`deviceInfo, ref btAddr);`

Example for VB	`Dim b1 As Integer`
	`Dim btAddr As Ulong`
	`b1 = Cipherlab.SystemAPI.Member.GetDeviceAddress(deviceInfo, btAddr)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:
See Also	FindNextBTDevice

21	ERROR_NO_DEVICE_INFO
25	ERROR_NOT_DEVICEINFO_OBJ
31	ERROR_WRONG_ARG

GetDeviceName

Purpose	To get the name of the connected device.
Syntax	**int GetDeviceName (int** *deviceInfo***,** **ref string** *deviceName***);**
Parameters	*deviceInfo* [in] A value returned by FindNextBTDevice(). *deviceName* [out] Pointer to a buffer where the device name is stored.

Example for C#	`int b1 = 0;`
	`string deviceName = string.Empty;`
	`b1 = Cipherlab.SystemAPI.Member.GetDeviceName(`
	`deviceInfo, ref deviceName);`

Example for VB	`Dim b1 As Integer`
	`Dim deviceName As String`
	`b1 = Cipherlab.SystemAPI.Member.GetDeviceName(deviceInfo, deviceName)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

21	ERROR_NO_DEVICE_INFO
25	ERROR_NOT_DEVICEINFO_OBJ
31	ERROR_WRONG_ARG

See Also	FindNextBTDevice

GetServiceType	
Purpose	To get the Bluetooth service that is currently in use.
Syntax	**int GetServiceType (int** *serviceInfo*, **ref int** *serviceType*);
Parameters	*serviceInfo* [in] A value returned by FindService(). *serviceType* [out] An integer variable that stores the information.

Example for C#	`int b1 = 0;`
	`int serviceInfo = 0;`
	`b1 = Cipherlab.SystemAPI.Member.GetServiceType(`
	`serviceInfo, ref serviceType);`

Example for VB	`Dim b1 As Integer`
	`Dim serviceType As Integer`
	`b1 = Cipherlab.SystemAPI.Member.GetServiceType(`
	`serviceInfo, serviceType)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:
See Also	FindService

21	ERROR_NO_DEVICE_INFO
26	ERROR_NOT_SERVICEINFO_OBJ
31	ERROR_WRONG_ARG

3.9.5. BLUETOOTH CONNECTION & STATUS

ConnectByMacAddr

Purpose: To establish Bluetooth connection by MAC address and service type

Syntax:

int ConnectByMacAddr (ulong *deviceMacAddr*,
int *targetService*,
string *pinCode*,
int *sppComPort*,
ref int *deviceInfo*,
ref int *serviceInfo*);

Parameters:

deviceMacAddr

[in] A value that specifies the MAC address of the target device.

targetService

[in] A value that specifies which Bluetooth service to connect.

1	BT_HID_SERVICE	
2	BT_DUN_SERVICE	COM 0 (client)
3	BT_SPP_SERVICE	COM 0 (client)
4	(Reserved)	
5	(Reserved)	
6	BT_HEADSET_SERVICE	
7	BT_HANDSFREE_SERVICE	
8	BT_FTP_SERVICE	

pinCode

[in] A string variable that stores the PIN code.

sppComPort

[in] A value that specifies the serial port used for connecting SPP service.

deviceInfo

[out] An integer variable that stores the information.

serviceInfo

[out] An integer variable that stores the information.

Example for C#

```
int b1 = 0;
ulong deviceMacAddr = 12345678;
int targetService = Cipherlab.SystemAPI.Member.BT_SPP_SERVICE, sppComPort = 6;
int deviceInfo = 0, serviceInfo = 0;
string pinCode = “0000”;
b1 = Cipherlab.SystemAPI.Member.ConnectByMacAddr(
deviceMacAddr, targetService, pinCode, sppComPort, ref deviceInfo, ref serviceInfo);
```

Example for VB

```
Dim b1 As Integer
Dim deviceMacAddr As Ulong = 12345678
Dim targetService As Integer = Member.BT_SPP_SERVICE
Dim sppComPort As Integer = 6
Dim deviceInfo As Integer = 0
Dim serviceInfo As Integer = 0
Dim pinCode As String = “0000”
b1 = Cipherlab.SystemAPI.Member.ConnectByMacAddr(
deviceMacAddr, targetService, pinCode, sppComPort, deviceInfo,
serviceInfo)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

4	ERROR_PARAMETER
21	ERROR_NO_DEVICE_INFO
22	ERROR_NO_SERVICE_INFO
27	ERROR_WRONG_SERVICE_TYPE
32	ERROR_OPERATION_FAIL

GetBTConnStatus

Purpose

To find out the status of the current Bluetooth connection.

Syntax

int GetBTConnStatus (int *serviceInfo*,
ref int *connStatus*);

Parameters

serviceInfo
[in] A value returned by FindService().
connStatus
[out] An integer variable that stores the information.

0	Disconnect with the target Bluetooth device
1	Connect with the target Bluetooth device

Example for C#

```
int b1 = 0;
int status = 0;
b1 = Cipherlab.SystemAPI.Member.GetBTConnStatus(
serviceInfo, ref connStatus);
```

Example for VB

```
Dim b1 As Integer
Dim connStatus As Integer
b1 = Cipherlab.SystemAPI.Member.GetBTConnStatus(
serviceInfo, connStatus)
```

Return Value

If successful, it returns 1.
Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

4	ERROR_PARAMETER
22	ERROR_NO_SERVICE_INFO
26	ERROR_NOT_SERVICEINFO_OBJ

See Also

FindService

SetBTConnStatus

Purpose

To set Bluetooth connection.

Syntax

int SetBTConnStatus (int *serviceType*,
int *onOff*,
int *deviceInfo*,
int *serviceInfo*);

Parameters

serviceType

[in] A value that specifies which Bluetooth service is desired.

1	BT_HID_SERVICE	
2	BT_DUN_SERVICE	COM 0 (client)
3	BT_SPP_SERVICE	COM 0 (client)
4	(Reserved)	
5	(Reserved)	
6	BT_HEADSET_SERVICE	
7	BT_HANDSFREE_SERVICE	
8	BT_FTP_SERVICE	

connStatus

[in] A value that specifies whether to establish a connection using the specified Bluetooth service.

0	Disconnect with the target Bluetooth device
1	Connect with the target Bluetooth device

deviceInfo

[in] A value returned by FindNextBTDevice().

serviceInfo

[in] A value returned by FindService().

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetBTConnStatus(
1, 1, deviceInfo, serviceInfo);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetBTConnStatus(
1, 1, deviceInfo, serviceInfo)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

21	ERROR_NO_DEVICE_INFO
22	ERROR_NO_SERVICE_INFO
25	ERROR_NOT_DEVICEINFO_OBJ
26	ERROR_NOT_SERVICEINFO_OBJ

27	ERROR_WRONG_SERVICE_TYPE
28	ERROR_DEVICE_NOT_HAS_SERVICE
32	ERROR_OPERATION_FAIL

See Also FindNextBTDevice, FindService

3.9.6. FTP SERVICE – VIEW FILES

StartBrowse	
Purpose	To initialize the browse procedure.
Syntax	**int StartBrowse (int** *serviceInfo*, **string** *targetPath*);
Parameters	*serviceInfo* [in] A value returned by FindService(). *targetPath* [in] A string variable that stores the directory path information.
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.StartBrowse(serviceInfo, "\\");`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.StartBrowse(serviceInfo, "\")`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:
See Also	FindService

22	ERROR_NO_SERVICE_INFO
26	ERROR_NOT_SERVICEINFO_OBJ
27	ERROR_WRONG_SERVICE_TYPE
29	ERROR_SERVICE_NOT_CONNECTED
32	ERROR_OPERATION_FAIL

GetNextFile

Purpose	To view files after StartBrowse() is called.
Syntax	**int GetNextFile (int** *serviceInfo*, **ref Ftp_File_Info** *fileInfo*);
Parameters	*serviceInfo* [in] A value returned by FindService(). *fileInfo* [out] FTP_File_Info structure that stores the information.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.Ftp_File_Info fileInfo =`
	`new Cipherlab.SystemAPI.Member.Ftp_File_Info();`
	`b1 = Cipherlab.SystemAPI.Member.GetNextFile(serviceInfo, fileInfo);`

Example for VB	`Dim b1 As Integer`
	`Dim fileInfo As Member.Ftp_File_Info`
	`b1 = Cipherlab.SystemAPI.Member.GetNextFile(serviceInfo, fileInfo)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

22	ERROR_NO_SERVICE_INFO
26	ERROR_NOT_SERVICEINFO_OBJ
27	ERROR_WRONG_SERVICE_TYPE
29	ERROR_SERVICE_NOT_CONNECTED
31	ERROR_WRONG_ARG
32	ERROR_OPERATION_FAIL

See Also	FindService

3.9.7. FTP SERVICE – FILE TRANSFER

GetFile

Purpose	To download a file from the FTP share folder.
Syntax	**int GetFile (int** *serviceInfo*, **ref Ftp_File_Info** *fileInfo*);
Parameters	*serviceInfo* [in] A value returned by FindService(). *fileInfo* [out] FTP_File_Info structure that stores the information.
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.GetFile(serviceInfo, fileInfo);`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.GetFile(serviceInfo, fileInfo)`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:
See Also	FindService, GetNextFile, StartBrowse

22	ERROR_NO_SERVICE_INFO
26	ERROR_NOT_SERVICEINFO_OBJ
27	ERROR_WRONG_SERVICE_TYPE
29	ERROR_SERVICE_NOT_CONNECTED
31	ERROR_WRONG_ARG
32	ERROR_OPERATION_FAIL

PutFile	
Purpose	To upload a file to the FTP share folder.
Syntax	**int PutFile (int** *serviceInfo***,** **string** *filePath***);**
Parameters	*serviceInfo* [in] A value returned by FindService(). *filePath* [in] A string variable that stores the file path information.

Example for C#	`int b1 = 0;`
	`string filePath = "\\DiskOnChip\\Test.txt";`
	`b1 = Cipherlab.SystemAPI.Member.PutFile(serviceInfo, filePath);`

Example for VB	`Dim b1 As Integer`
	`Dim filePath As String = "\DiskOnChip\Test.txt"`
	`b1 = Cipherlab.SystemAPI.Member.PutFile(serviceInfo, filePath)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:
See Also	FindService, GetNextFile, StartBrowse

22	ERROR_NO_SERVICE_INFO
26	ERROR_NOT_SERVICEINFO_OBJ
27	ERROR_WRONG_SERVICE_TYPE
29	ERROR_SERVICE_NOT_CONNECTED
31	ERROR_WRONG_ARG
32	ERROR_OPERATION_FAIL

3.9.8. SPP COM PORT

GetSppComPort	
Purpose	To find out the serial port used for connecting SPP service.
Syntax	**int GetSppComPort (int** *serviceInfo*, **ref int** *comport*);
Parameters	*serviceInfo* [in] A value returned by FindService(). *Comport* [in] Pointer to a buffer where the information is stored.

Example for C#	`int b1 = 0, serviceInfo = 0, comport = 0;`
	`b1 = Cipherlab.SystemAPI.Member.FindService(`
	`Member.BT_SPP_SERVICE, deviceInfo, ref serviceInfo);`
	`b1 = Cipherlab.SystemAPI.Member.GetSppComPort(`
	`serviceInfo, ref □uttone);`

Example for VB	`Dim b1 As Integer`
	`Dim serviceInfo As Integer`
	`Dim □uttone As Integer`
	`b1 = Cipherlab.SystemAPI.Member.FindService(`
	`Member.BT_SPP_SERVICE, deviceInfo, serviceInfo)`
	`b1 = Cipherlab.SystemAPI.Member.GetSppComPort(`
	`serviceInfo, □uttone)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

4	ERROR_PARAMETER
22	ERROR_NO_SERVICE_INFO
26	ERROR_NOT_SERVICEINFO_OBJ
27	ERROR_WRONG_SERVICE_TYPE

SetSppComPort

Purpose	To set the serial port used for connecting SPP service.
Syntax	**int SetSppComPort (int** *serviceInfo*, **int** *comport*);
Parameters	*serviceInfo* [in] A value returned by FindService(). *Comport* [in] Always uses COM 0 (Client).

Example for C#	`int b1 = 0, serviceInfo = 0, uttone = 0;`
	`b1 = Cipherlab.SystemAPI.Member.FindService(`
	`Member.BT_SPP_SERVICE, deviceInfo, ref serviceInfo);`
	`b1 = Cipherlab.SystemAPI.Member.SetSppComPort(serviceInfo, uttone);`

Example for VB	`Dim b1 As Integer`
	`Dim serviceInfo As Integer`
	`Dim uttone As Integer`
	`b1 = Cipherlab.SystemAPI.Member.FindService(`
	`Member.BT_SPP_SERVICE, deviceInfo, serviceInfo)`
	`b1 = Cipherlab.SystemAPI.Member.SetSppComPort(`
	`serviceInfo, uttone)`

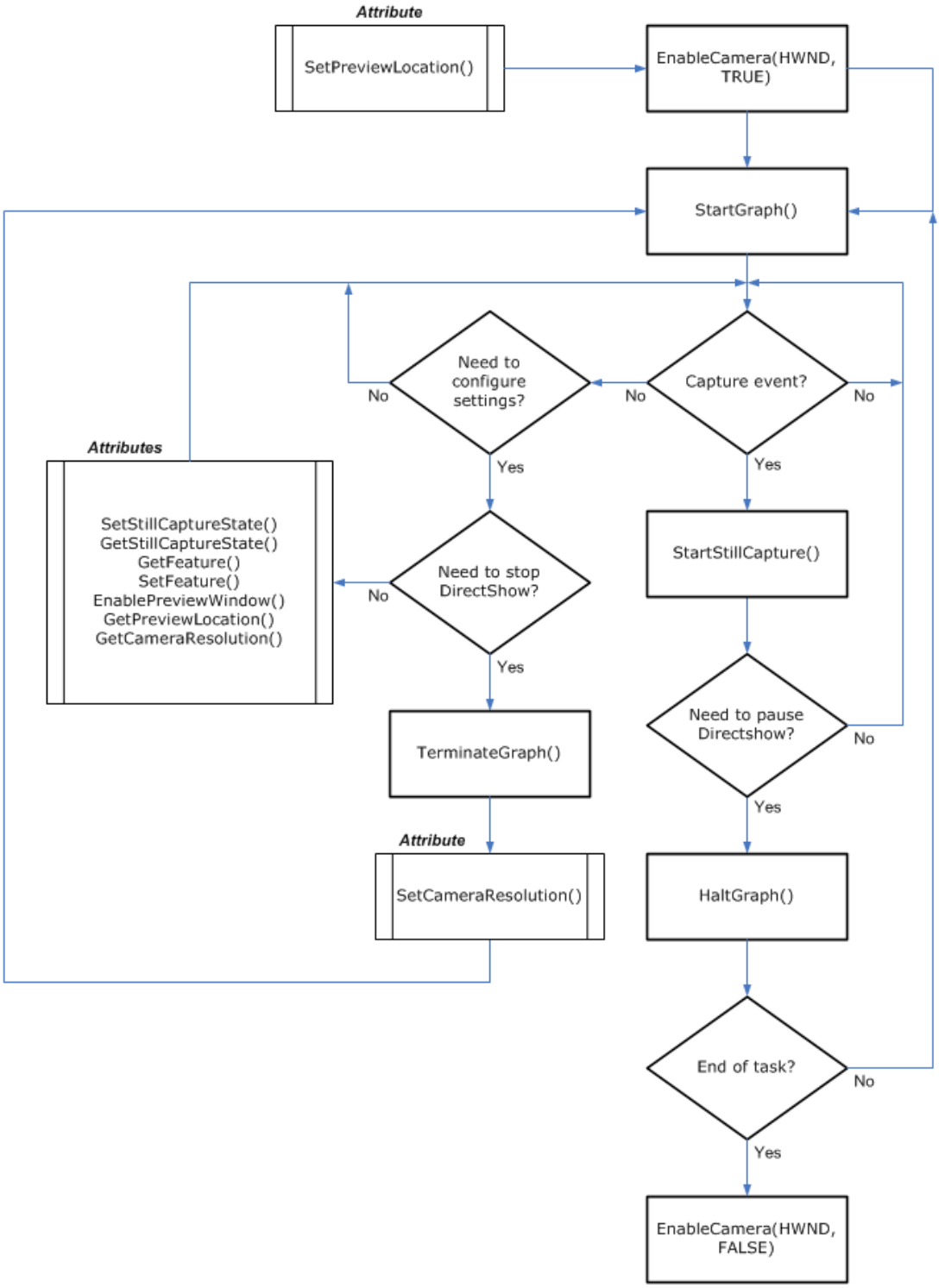
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

22	ERROR_NO_SERVICE_INFO
26	ERROR_NOT_SERVICEINFO_OBJ
27	ERROR_WRONG_SERVICE_TYPE

Remarks	By default, only COM 0 is the serial port used for connecting SPP service. This function must be called before SetBTConnStatus().

3.10. CAMERA

Here is the programming flow for camera operation.



3.10.1. CAMERA POWER

EnableCamera

Purpose	To turn on the camera or turn it off.
Syntax	**int EnableCamera (IntPtr** *wnd*, **int** *onOff*);
Parameters	*wnd* [in] IntPtr variable *onOff* [in] Integer variable

0	Disable (Turn off the camera.)
1	Enable (Turn on the camera.)

Example for C#

```
int b1 = 0;
int onOff = 1;
b1 = Cipherlab.SystemAPI.Member.EnableCamera(this.Handle, onOff);
```

Example for VB

```
Dim b1 As Integer
Dim onOff As Integer = 1
b1 = Cipherlab.SystemAPI.Member.EnableCamera(Me.Handle, onOff)
```

Return Value

If successful, it returns 1.
Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

See Also: EnablePreviewWindow, SetPreviewLocation

3.10.2. CAMERA SETTINGS

GetCameraResolution

Purpose	To get the camera resolution.
Syntax	**int GetCameraResolution (ref int** *resValue***);**
Parameters	*resValue* [out] An integer variable that stores the information.
Example for C#	`int b1 = 0, resValue = 0;` `b1 = Cipherlab.SystemAPI.Member.GetCameraResolution(ref resValue );`

Example for VB	`Dim b1 As Integer`
	`Dim resValue As Integer`
	`b1 = Cipherlab.SystemAPI.Member.GetCameraResolution(resValue)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

SetCameraResolution

Purpose	To set the camera resolution.
Syntax	**int SetCameraResolution (int** *resValue***);**
Parameters	*resValue* [in] Integer variable

0	640 x 480 pixels
1	1280 x 960 pixels
2	1600 x 1200 pixels
3	2048 x 1536 pixels

Example for C#	`int b1 = 0, resValue = 1;` `b1 = Cipherlab.SystemAPI.Member.SetCameraResolution(resValue);`

Example for VB	`Dim b1 As Integer`
	`Dim resValue As Integer = 1`
	`b1 = Cipherlab.SystemAPI.Member.SetCameraResolution(resValue)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

Remarks	TerminateGraph() must be called before this function. The new resolution will not take effect until StartGraph() is called again.
See Also	StartGraph, TerminateGraph

GetFlashLight

Purpose	Gets camera flashlight setting..
Syntax	**int GetFlashLight (ref FLASH** *flashLight***);**
Parameters	*flashLight* [out] Pointer to structure Flash where flashlight setting is stored.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.FLASH pFlash`
	`= new Cipherlab.SystemAPI.Member.FLASH();`
	`b1 = Cipherlab.SystemAPI.Member.GetFlashLight(ref pFlash );`

Example for VB	`Dim b1 As Integer`
	`Dim pFlash As New Cipherlab.SystemAPI.Member.FLASH`
	`b1 = Cipherlab.SystemAPI.Member.GetFlashLight(pFlash)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

SetFlashLight

Purpose	Sets if camera beams a light for the picture to take.
Syntax	**int SetFlashLight (FLASH** *flashLight***);**
Parameters	*flashLight* [in] Pointer to structure Flash where flashlight setting is stored.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.FLASH pFlash`
	`= new Cipherlab.SystemAPI.Member.FLASH();`
	`pFlash.CaptureFlash = 1;`
	`pFlash.PreviewFlash = 0;`
	`b1 = Cipherlab.SystemAPI.Member.StFlashLight(pFlash );`

Example for VB	`Dim b1 As Integer`
	`Dim pFlash As New Cipherlab.SystemAPI.Member.FLASH`
	`pFlash.CaptureFlash = 1;`
	`pFlash.PreviewFlash = 0;`
	`b1 = Cipherlab.SystemAPI.Member.StFlashLight(pFlash)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

3.10.3. PREVIEW

EnablePreviewWindow

Purpose	To show or hide the preview window.
Syntax	**int EnablePreviewWindow (int** *onOff***);**
Parameters	*onOff* [in] Integer variable

0	Disable (= No preview)
1	Enable (= Display a preview image)

Example for C#
```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.EnablePreviewWindow(1);
```

Example for VB
```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.EnablePreviewWindow(1)
```

Return Value
If successful, it returns 1.
Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)

See Also SetPreviewLocation

GetPreviewLocation

Purpose	To get the position of the preview window.
Syntax	**int GetPreviewLocation (ref int** *posX***,** **ref int** *□utt***);**
Parameters	*posX* Reserved [out] An integer variable that stores the information. *□utt* [out] An integer variable that stores the information.

Example for C#
```
int b1 = 0, posX = 0, posy = 0;
b1 = Cipherlab.SystemAPI.Member.GetPreviewLocation(
ref posX, ref □utt);
```

Example for VB
```
Dim b1 As Integer
Dim posX As Integer
Dim posy As Integer
b1 = Cipherlab.SystemAPI.Member.GetPreviewLocation(posX, posy)
```

Return Value
If successful, it returns 1.
Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

SetPreviewLocation	
Purpose	To set the position of the preview window.
Syntax	**void SetPreviewLocation (int** *posX*, **int** *utt*);
Parameters	*posX* Reserved [in] Integer variable *utt* [in] Integer variable
Example for C#	`int posX = 0, posy = 0;` `Cipherlab.SystemAPI.Member.SetPreviewLocation(posX, utt);`

Example for VB	`Dim posX As Integer`
	`Dim posy As Integer`
	`Cipherlab.SystemAPI.Member.SetPreviewLocation(posX, posy)`

Remarks	The size of the preview window is 240 x 180 pixels.
See Also	EnablePreviewWindow

GetFeature

Purpose: To get the special effect setting.

Syntax: **int GetFeature (ref int** *level***);**

Parameters: *level*

[out] An integer variable that stores the information.

Example for C#:

```
int b1 = 0, level = 0;
b1 = Cipherlab.SystemAPI.Member.GetFeature(ref level);
```

Example for VB:

```
Dim b1 As Integer
Dim level As Integer
b1 = Cipherlab.SystemAPI.Member.GetFeature(level)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

SetFeature

Purpose: To apply special effect to the preview image.

Syntax: **int SetFeature (int** *level***);**

Parameters: *level*

[in] Integer variable

0	Mono
1	Normal*
2	Negative
3	Sepia
4	Posterize
5	Whiteboard
6	Blackboard
7	Aqua

Example for C#:

```
int b1 = 0, level = 0;
b1 = Cipherlab.SystemAPI.Member.SetFeature(level);
```

Example for VB:

```
Dim b1 As Integer
Dim level As Integer
b1 = Cipherlab.SystemAPI.Member.SetFeature(level)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

StartGraph	
Purpose	To run DirectShow filters and start a graph in the preview window.
Syntax	**int StartGraph ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.StartGraph();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.StartGraph()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered: 1 — ERROR_NORESOURCE (Fail to get resource)
See Also	HaltGraph, TerminateGraph

TerminateGraph	
Purpose	To stop running DirectShow filters and flush the preview window.
Syntax	**int TerminateGraph ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.TerminateGraph();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.TerminateGraph()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered: 1 — ERROR_NORESOURCE (Fail to get resource)
See Also	HaltGraph, StartGraph

HaltGraph	
Purpose	To pause a graph in the preview window until StartGraph() is called again.
Syntax	**int HaltGraph ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.HaltGraph();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.HaltGraph()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered: 1 — ERROR_NORESOURCE (Fail to get resource)
See Also	StartGraph, TerminateGraph

3.10.4. STILL CAPTURE

GetStillCaptureState

Purpose	To get the capture settings.
Syntax	**int GetStillCaptureState (ref PicState** *picInfo***);**
Parameters	*picInfo* [out] PicState structure that stores the capture settings.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.PicState picInfo =`
	`new Cipherlab.SystemAPI.Member.PicState();`
	`b1 = Cipherlab.SystemAPI.Member.GetStillCaptureState(ref picInfo);`

Example for VB	`Dim b1 As Integer`
	`Dim picInfo As New Member.PicState`
	`b1 = Cipherlab.SystemAPI.Member.GetStillCaptureState(picInfo)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

SetStillCaptureState

Purpose	To set the capture settings.
Syntax	**int SetStillCaptureState (PicState** *picInfo***);**
Parameters	*picInfo* [in] PicState structure that stores the capture settings.

Example for C#	`int b1 = 0;`
	`Cipherlab.SystemAPI.Member.PicState picinfo =`
	`new Cipherlab.SystemAPI.Member.PicState();`
	`picinfo.ImageFormat = 0;`
	`picinfo.lpPathName = “\\DiskOnChip”;`
	`picinfo.PathSize = 260;`
	`picinfo.lpFileName = “Test001”;`
	`picinfo.FileSize = 260;`
	`b1 = Cipherlab.SystemAPI.Member.SetStillCaptureState(picinfo);`

Example for VB	`Dim b1 As Integer`
	`Dim picinfo As New Member.PicState`
	`picinfo.ImageFormat = 0`
	`picinfo.lpPathName = “\DiskOnChip”`
	`picinfo.PathSize = 260`
	`picinfo.lpFileName = “Test001”`
	`picinfo.FileSize = 260`
	`b1 = Cipherlab.SystemAPI.Member.SetStillCaptureState(picinfo)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:
Remarks	This function must be called before StartStillCapture().
See Also	StartStillCapture

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

StartStillCapture

Purpose	To start with still image capture.
Syntax	**int StartStillCapture (int** *playSound***);**
Parameters	*playSound* [in] Integer variable

0	No notification.
1	Play a sound to notify the image has been captured.

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.StartStillCapture(1, 1);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.StartStillCapture(1, 1)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

Remarks: Upon completion of the capture, it will send a Windows message WM_CAMERACAPTUREFINISH.

See Also: SetStillCaptureState

3.10.5. TORCH MODE

GetTorchMode

Purpose	Gets current torch mode.
Syntax	**int GetTorchMode (ref uint** *Mode***);**
Parameters	*Mode* [out] An unsigned integer variable that stores the information.

0	Disables
1	Enables

Example for C#

```
int b1 = 0;
uint Mode = 0;
b1 = Cipherlab.SystemAPI.Member.GetTorchMode(ref Mode);
```

Example for VB

```
Dim b1 As Integer
Dim Mode As Uinteger = 0
b1 = Cipherlab.SystemAPI.Member.GetTorchMode(Mode)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

SetTorchMode

Purpose	To enable or disable torch light.
Syntax	**int SetTorchMode (uint** *Mode***);**
Parameters	*Mode* [in] Unsigned integer variable.

0	Disables
1	Enables

Example for C#

```
int b1 = 0;
uint Mode = 1;
b1 = Cipherlab.SystemAPI.Member.SetTorchMode(Mode);
```

Example for VB

```
Dim b1 As Integer
Dim Mode As Uinteger = 1
b1 = Cipherlab.SystemAPI.Member.SetTorchMode(Mode)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Fail to get resource)
4	ERROR_PARAMETER (Wrong parameter)

3.11. GPS

The serial interface protocol is based on the national Marine Electronics Association's NMEA 0183 ASCII interface specification. For details, please refer to NMEA 0183 Version 3.01, which may be obtained from NMEA, www.nmea.org.

The table below lists the standard characteristics of the NMEA 0183 data transmissions:

Serial Configuration	NMEA Standard
Baud Rate	4800
Data Bits	8
Parity	None
Stop bits	1

The GPS receiver hosted on the mobile computer uses COM 7 to output NMEA messages, as long as the COM port is opened. You may use Microsoft's standard API to open the serial port and receive NMEA format data. The following URLs are documentations related to serial port programming techniques:

http://msdn.microsoft.com/en-us/library/bb202722.aspx
http://msdn.microsoft.com/en-us/library/ms810467.aspx
http://msdn.microsoft.com/en-us/library/bb201943.aspx

3.12. BUTTON ASSIGNMENT

GetButtonAssignment

Purpose: To find out the assignment of a user-definable key.

Syntax:
int GetButtonAssignment (int *buttonID*,
ref int *keyCode*,
ref string *buffer*);

Parameters: *buttonID*
[in] Integer variable

1	Side-Trigger-Left	(default: Scan)
2	Side-Trigger-Right	(default: Scan)
3	(Reserved)	(Reserved)
4	(Reserved)	(Reserved)
5	(Reserved)	(Reserved)
6	(Reserved)	(Reserved)
7	Send	(default: Send)
8	End	(default: End)
9	(Reserved)	(Reserved)
10	(Reserved)	(Reserved)
11	Up	(default: Up)
12	Down	(default: Down)
13	Left	(default: Left)
14	Right	(default: Right)
15	ESC	(default: Esc)
16	TAB	(default: Tab)
17	Backspace	(default: Backspace)
18	Volume Up	(default: Volume Up)
19	Volume Down	(default: Volume Down)
20	Application (programming key)	(default: VK_OEM_KEY_1)
21	*Shift	(default: Shift)
	▸ This is a special key code which is not assignable; lock/unlock available only. Unlocked after system reset.	
22	(Reserved)	(Reserved)
23	Enter	(default: Enter)
24	*Alpha	(default: Alpha)
	▸ This is a special key code which is not assignable; lock/unlock available only. Unlocked after system reset.	

25	*Fn	(Default: Fn)
	▸ This is a special key code which is not assignable; lock/unlock available only. Unlocked after system reset.	
26	(Reserved)	(Reserved)
27	(Reserved)	(Reserved)
28	-	(default: -)
29	.	(default: .)
107	(Reserved)	(Reserved)
108	(Reserved)	(Reserved)
109	(Reserved)	(Reserved)
110	(Reserved)	(Reserved)
111	(Reserved)	(Reserved)
112	(Reserved)	(Reserved)
113	(Reserved)	(Reserved)
114	(Reserved)	(Reserved)
115	(Reserved)	(Reserved)
116	(Reserved)	(Reserved)
117	(Reserved)	(Reserved)

keyCode

[out] An integer variable that stores the key assignment information.

Buffer

[out] Pointer to a buffer where user-defined key code or the full file path of the program is stored.

Example for C#

```
int b1 = 0;
int ikeyCode = 0;
string sprogram = string.Empty;
b1 = Cipherlab.SystemAPI.Member.GetButtonAssignment(
1, ref ikeyCode, ref sprogram);
```

Example for VB

```
Dim b1 As Integer
Dim ikeyCode As Integer
Dim sprogram As String = “”
b1 = Cipherlab.SystemAPI.Member.GetButtonAssignment(
1, ikeyCode, sprogram)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

▸ Call GetErrorCode() to find the error condition encountered:

1	Fail to open registry
2	Fail to set registry
4	ERROR_PARAMETER (Wrong parameter)

SetButtonAssignment

Purpose: Assigns a user-definable key to act as another key or quick-launch a program.

Syntax:

```
int GetButtonAssignment (int buttonID,
                         int keyCode,
                         string buffer);
```

Parameters

buttonID

[in] Integer variable

1	Side-Trigger-Left	(default: Scan)
2	Side-Trigger-Right	(default: Scan)
3	(Reserved)	(Reserved)
4	(Reserved)	(Reserved)
5	(Reserved)	(Reserved)
6	(Reserved)	(Reserved)
7	Send	(default: Send)
8	End	(default: End)
9	(Reserved)	(Reserved)
10	(Reserved)	(Reserved)
11	Up	(default: Up)
12	Down	(default: Down)
13	Left	(default: Left)
14	Right	(default: Right)
15	ESC	(default: Esc)
16	TAB	(default: Tab)
17	Backspace	(default: Backspace)
18	Volume Up	(default: Volume Up)
19	Volume Down	(default: Volume Down)
20	Application (programming key)	(default:VK_OEM_KEY_1)
21	*Shift	(default: Shift)
	▸ This is a special key code which is not assignable; lock/unlock available only. Unlocked after system reset.	
22	(Reserved)	(Reserved)
23	Enter	(default: Enter)
24	*Alpha	(default: Alpha)
	▸ This is a special key code which is not assignable; lock/unlock available only. Unlocked after system reset.	
25	*Fn	(Default: Fn)
	▸ This is a special key code which is not assignable; lock/unlock available only. Unlocked after system reset.	

26	(Reserved)	(Reserved)
27	(Reserved)	(Reserved)
28	-	(default: -)
29	.	(default: .)
107	(Reserved)	(Reserved)
108	(Reserved)	(Reserved)
109	(Reserved)	(Reserved)
110	(Reserved)	(Reserved)
111	(Reserved)	(Reserved)
112	(Reserved)	(Reserved)
113	(Reserved)	(Reserved)
114	(Reserved)	(Reserved)
115	(Reserved)	(Reserved)
116	(Reserved)	(Reserved)
117	(Reserved)	(Reserved)

keyCode

[in] Integer variable

0	User-defined key code (0x01~0xFF)
1	Launch the specified program
2	Enter
3	Scan
4	Esc
5	Delete
6	Backspace
7	Space
8	Tab
9 **10** **...** **20**	F1 F2 ... F12
21	Start Menu
22	Alt
23	OEM_Key1 (0xE9)
24	OEM_Key2 (0xEA)
25	OEM_Key3 (0xEB)
26	OEM_Key4 (0xEC)
27	OEM_Key5 (0xED)

28	OEM_Key6 (0xEE)
29	OEM_Key7 (0xEF)
30	OEM_Key8 (0xF0)
31	OEM_Key9 (0xF1)
32	OEM_Key10 (0x2A)
33	*
34	#
35	Send(VK_TTALK)
36	End(VK_END)
37	(Reserved)
38	Up
39	Down
40	Left
41	Right
42	TAB
43	Volume Up
44	Volume Down
45	(Reserved)
46	(Reserved)
47	OK
48	(Reserved)
49	(Reserved)
50	Home(VK_HOME)
51	End(VK_END)
52	Fn+ESC(0xF5)
53 **54** **...** **64**	F13 F14 ... F24
65	Home(VK_LWIN+ VK_APP2)
66	(Reserved)
67	-
68	.
997	Lock key ▸ Locks keys; the locked key will have no function under any mode (Shift+key, Blue+key, Orange+key)
998	Unlock key

buffer

[in] A string variable that stores user-defined key code or the full path to the application to launch.

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetButtonAssignment(
1, 1, “\\Windows\\Buttons.exe”);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetButtonAssignment(
1, 1, “\Windows\Buttons.exe”)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	Fail to open registry
2	Fail to set registry
4	ERROR_PARAMETER (Wrong parameter)

3.13. SIGNATURE CAPTURE

3.13.1. INITIALIZATION/EXIT

InitSigScreen

Purpose

To initialize and show the signature area.

Syntax

```
int InitSigScreen (int x,
                   int y,
                   int width,
                   int height,
                   IntPtr parentWnd);
```

Parameters

x

[in] A value that specifies the X-coordinate of the start position for the area that allows signature capture.

Y

[in] A value that specifies the Y-coordinate of the start position for the area that allows signature capture.

Width

[in] A value that specifies the width of the area.

Height

[in] A value that specifies the height of the area.

parentWnd

[in] A handle to the parent or owner window where the signature capture functionality is implemented.

Example for C#

```
int b1 = 0;
int x = 0;
int y = 0;
int width = 150;
int height = 80;
b1 = Cipherlab.DrawPanelAPI.Member.InitSigScreen(
x, y, width, height, this.Handle);
```

Example for VB

```
Dim b1 As Integer
Dim x As Integer = 0
Dim y As Integer = 0
Dim width As Integer = 150
Dim height As Integer = 80
b1 = Cipherlab.DrawPanelAPI.Member.InitSigScreen(
x, y, width, height, this.Handle)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	SIGERROR_UNKNOWN
2	SIGERROR_MACHINE_NOT_SUPPORTED

Remarks	This function must be called before any other signature-related functions. The signature area is enabled upon completion of initialization.

CloseSigScreen	
Purpose	To exit the signature area.
Syntax	**void CloseSigScreen ();**
Example for C#	`Cipherlab.DrawPanelAPI.Member.CloseSigScreen();`
Example for VB	`Cipherlab.DrawPanelAPI.Member.CloseSigScreen()`
Remarks	The signature-related functions will not take effect until InitSigScreen() is called again.

3.13.2. CONTROL OF SIGNATURE AREA

ClearSigArea

Purpose	To clear the entire signature.
Syntax	**void ClearSigArea ();**
Example for C#	`Cipherlab.DrawPanelAPI.Member.ClearSigArea();`
Example for VB	`Cipherlab.DrawPanelAPI.Member.ClearSigArea()`

EnableSigScreen

Purpose	To enable or disable the signature area.
Syntax	**void EnableSigScreen (int** *enabled***);**
Parameters	*enabled* [in] Integer variable

0	Disable
1	Enable

Example for C#	`int enabled = 1;` `Cipherlab.DrawPanelAPI.Member.EnableSigScreen(enabled);`
Example for VB	`Dim enabled As Integer = 1` `Cipherlab.DrawPanelAPI.Member.EnableSigScreen(enabled)`

ShowSigScreen

Purpose	To show or hide the signature area.
Syntax	**void ShowSigScreen (int** *showArea***);**
Parameters	*showArea* [in] Integer variable

0	Hide
1	Show

Example for C#	`int showArea = 1;` `Cipherlab.DrawPanelAPI.Member.ShowSigScreen(showArea);`
Example for VB	`Dim showArea As Integer = 1` `Cipherlab.DrawPanelAPI.Member.ShowSigScreen(showArea)`

3.13.3. BACKGROUND COLOR

SetSigBgColor	
Purpose	To set the background color of the signature area.
Syntax	**void SetSigBgColor (int** *redOfRGB*, **int** *greenOfRGB*, **int** *blueOfRGB*);
Parameters	*redOfRGB* [in] Integer variable **0~255** 255* *greenOfRGB* [in] Integer variable **0~255** 255* *blueOfRGB* [in] Integer variable **0~255** 255*
Example for C#	`Cipherlab.DrawPanelAPI.Member.SetSigBgColor(255, 255, 255);`
Example for VB	`Cipherlab.DrawPanelAPI.Member.SetSigBgColor(255, 255, 255)`

3.13.4. PEN COLOR AND WIDTH

SetSigLineColor

Purpose	To set the pen color.
Syntax	**void SetSigLineColor (int** *redOfRGB*, **int** *greenOfRGB*, **int** *blueOfRGB*);
Parameters	*redOfRGB* [in] Integer variable **0~255** | 0* *greenOfRGB* [in] Integer variable **0~255** | 0* *blueOfRGB* [in] Integer variable **0~255** | 0*
Example for C#	`Cipherlab.DrawPanelAPI.Member.SetSigLineColor(0, 0, 0);`
Example for VB	`Cipherlab.DrawPanelAPI.Member.SetSigLineColor(0, 0, 0)`

SetSigLineWidth

Purpose	To set the pen width, in pixels.
Syntax	**void SetSigLineWidth (int** *width*);
Parameters	*width* [in] Integer variable **1~5** | 1*
Example for C#	`int width = 1;` `Cipherlab.DrawPanelAPI.Member.SetSigLineWidth(width);`
Example for VB	`Dim width As Integer = 1` `Cipherlab.DrawPanelAPI.Member.SetSigLineWidth(width)`

3.13.5. IMAGE SAVING & LOADING

LoadSigImage	
Purpose	To load the signature image.
Syntax	**int LoadSigImage (string** *filePath***);**
Parameters	*filePath* [in] A string variable that stores the full file path of the image.

Example for C#	`int b1 = 0;`
	`string filePath = "\\DiskOnChip\\sig.jpg";`
	`b1 = Cipherlab.DrawPanelAPI.Member.LoadSigImage(filePath);`

Example for VB	`Dim b1 As Integer`
	`Dim filePath As String = "\DiskOnChip\sig.jpg"`
	`b1 = Cipherlab.DrawPanelAPI.Member.LoadSigImage(filePath)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:
Remarks	We suggest not calling this function while initializing the signature area. Also, it cannot load an image with a resolution greater than 640 x 480 pixels.

1	SIGERROR_UNKNOWN
3	SIGERROR_SIG_WINDOW_NOT_CREATED
4	SIGERROR_FILE_NOT_FOUND
6	SIGERROR_HIDDEN_WINDOW

SaveSigImage

Purpose

To save the signature image at a resolution no greater than 640 x 480 pixels.

Syntax

int SaveSigImage (int *imageFormat*,
string *filePath*);

Parameters

imageFormat

[in] Integer variable

0	BMP_FORMAT	To save in BMP file format.
1	JPG_FORMAT	To save in JPEG file format.
2	LOCUS_FORMAT	To save in LOCUS file format.

filePath

[in] A string variable that stores the full file path of the image.

Example for C#

```
int b1 = 0;
string filePath = "\\DiskOnChip\\sig.jpg";
b1 = Cipherlab.DrawPanelAPI.Member.SaveSigImage(1, filePath);
```

Example for VB

```
Dim b1 As Integer
Dim filePath As String = "\DiskOnChip\sig.jpg"
b1 = Cipherlab.DrawPanelAPI.Member.SaveSigImage(1, filePath)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

- Call GetErrorCode() to find the error condition encountered:

1	SIGERROR_UNKNOWN
3	SIGERROR_SIG_WINDOW_NOT_CREATED
6	SIGERROR_HIDDEN_WINDOW

3.13.6. SIGNATURE DLL VERSION

GetSigNetDllVer	
Purpose	To obtain the current .NET library version for Signature DLL.
Syntax	**int GetSigNetDllVer (ref string** *buffer***);**
Parameters	*buffer* [out] Pointer to a buffer where the version information is stored.

Example for C#	`int b1 = 0;`
	`string buf = string.Empty;`
	`b1 = Cipherlab.SystemAPI.Member.GetSigNetDllVer(ref buf);`

Example for VB	`Dim b1 As Integer`
	`Dim buf As String = ""`
	`b1 = Cipherlab.SystemAPI.Member.GetSigNetDllVer(buf)`

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

5	SIGERROR_ILLEGAL_PARAMETER

3.14. ERROR INFORMATION

GetErrorCode	
Purpose	To find the current error condition encountered.
Syntax	**int GetErrorCode ();**

Example for C#	`int b1 = 0;`
	`int errCode = 0;`
	`b1 = Cipherlab.SystemAPI.Member.SystemSoftReset();`
	`if(b1 == 0)`
	`{`
	`errCode = Member.GetErrorCode();`
	`}`

Example for VB	`Dim errCode As Integer`
	`If Cipherlab.SystemAPI.Member.SystemSoftReset() = 0 Then`
	`errCode = Member.GetErrorCode()`
	`End If`

Return Value	If successful, it returns the error code.

Chapter 4

DATA STRUCTURE

4. IN THIS CHAPTER

4.1 System Information Structure 258
4.2 Backlight Structure 260
4.3 Keypad Structure 262
4.4 Wi-Fi Structures 263
4.5 Bluetooth Structure 263
4.6 Camera Structures 285

4.1. SYSTEM INFORMATION STRUCTURE

4.1.1. SYSINFO

This structure stores the system information, which is identical to the settings displayed on the mobile computer under **Start Screen** | **Settings** | **System** | **Device Information**.

```
public struct SysInfo
{
    public string        DevDesc;
    public string        DevCFG;
    public string        SerialNum;
    public string        SerialNumReadOnly;
    public string        SerialNumPCBA;
    public string        Manufactory;
    public string        ManuDate;
    public string        OsVer;
    public string        BootloaderVer;
    public string        MicroPVer;
    public string        BTAddress;
    public string        WiFiAddress;
    public string        IMEI;
} SYSINFO, *PSYSINFO;
```

Data Type	Member Name	Description
string	DevDesc	Device Model
string	DevCFG	(Reserved)
string	SerialNum	Serial Number
string	SerialNumReadOnly	Serial number for factory use[Note]
string	SerialNumPCBA	Serial number of PCBA[Note] — always 16 characters "XXXXXXXXXXXXXXXX"
string	Manufactory	Manufacturer
string	ManuDate	Manufacturer Date
string	OsVer	OS Version
string	BootloaderVer	Bootloader version
string	MicroPVer	MicroP firmware version
string	BTAddress	Bluetooth MAC address
string	WiFiAddress	Wi-Fi MAC address
string	IMEI	GSM's unique number

Note: (1) By default, SerialNum and SerialNumReadOnly are the same.
(2) Information on SerialNumReadOnly and SerialNumPCBA can be accessed by calling GetSysInfo() only.

4.2. BACKLIGHT STRUCTURE

4.2.1. BKLCTL

This structure stores information about the backlight control, which is identical to the settings displayed on the mobile computer under | **Settings** | **System** | **Backlight.**

- Battery mode - Battery power in use
- AC mode - The mobile computer is connected to an AC power source via the cradle.

```
public struct BklCtl
{
    public uint          batttimeout;
    public uint          lightlevel;
    public uint          actimeout;
    public uint          aclightlevel;
    public uint          backlightontap;
    public uint          acbacklightontap;
    public uint          usebattery;
    public uint          useext;
}
```

Data Type	Member Name	Description
uint	batttimeout	In battery mode, turn off backlight after a specified period of idle time – 15, 30, 60, 120, 300 (in units of second). ▸ "usebattery" must be enabled!
uint	lightlevel	Backlight level in battery mode – 0~11 from dark to bright
uint	actimeout	In AC mode, turn off backlight after a specified period of idle time – 15, 30, 60, 120, 300 (in units of second). ▸ "useext" must be enabled!
uint	aclightlevel	Backlight level in AC mode – 0~11 from dark to bright
uint	backlightontap	When backlight is off in battery mode, it will be automatically turned on when you tap the touch screen or press a key – 0 Disable backlight on tap in battery mode 1 Enable backlight on tap in battery mode
uint	acbacklightontap	When backlight is off in AC mode, it will be automatically turned on when you tap the touch screen or press a key – 0 Disable backlight on tap in AC mode 1 Enable backlight on tap in AC mode

<table>
<tr><td>uint</td><td>usebattery</td><td>Power-saving mode for backlight when using battery power –<table><tr><td>0</td><td>Disable power-saving in battery mode</td></tr><tr><td>1</td><td>Enable power-saving in battery mode</td></tr></table>▸ When enabled, proceed to define "batttimeout" and "backlightontap".</td></tr>
<tr><td>uint</td><td>useext</td><td>Power-saving mode for backlight when using external power –<table><tr><td>0</td><td>Disable power-saving in AC mode</td></tr><tr><td>1</td><td>Enable power-saving in AC mode</td></tr></table>▸ When enabled, proceed to define "actimeout" and "acbacklightontap".</td></tr>
</table>

4.3. KEYPAD STRUCTURE

4.3.1. KEYPADBIND

This structure stores the information about the program to launch upon system initialization.

```
public struct KeyPadBind
{
   public string        szClass;
}
```

Data Type	Member Name	Description
string	szClass	Name of the program to be executed.

4.4. WI-FI STRUCTURES

4.4.1. BSSIDINFO

This structure stores information about the preferred wireless zero configuration.

```
public struct BSSIDINFO
{
    public string macAddress;
    public string ssid;
    public int privacy;
    public int rssi;
    public int networkTypeInUse;
    public int configFreq;
    public int infrastructureMode;
    public int authentication;
    public byte[] supportedRates;
}
```

Data Type	Member Name	Description
string	macAddress	MAC address of the associated access point.
string	ssid	NIC service set identifier (SSID) value.
int	privacy	Wireless Equivalent Privacy (WEP) implementation status for a particular 802.11 NIC.
int	rssi	Received signal strength indication (RSSI) in dB-milliwatts (dBm).
int	networkTypeInUse	Network subtype values for the physical layer of used with a specific driver.
int	configFreq	Specifies the frequency of the selected channel in kHz.
int	infrastructureMode	This enumerated type holds specific infrastructure values for the miniport driver to set in an 802.11 NIC.
int	authentication	802.11 authentication mode value
byte[]	supportedRates	This data type provides a way to handle the data rates associated with 802.11 delivery.

4.4.2. WLANADPTINFO

Structure WLANADPTINFO stores the information about the mobile computer's Wi-Fi networking which is viewable at **Start screen** | **Settings** | **Connections** | **Wi-Fi** | **Network Adapters**.

```
public struct WlanAdptInfo
{
    public uint          fUseDHCP;
    public string        IPAddr;
    public string        SubnetMask;
    public string        Gateway;
    public string        DNSAddr;
    public string        DNSAltAddr;
    public string        WINSAddr;
    public string        WINSAltAddr;
}
```

Data Type	Member Name	Description
uint	fUseDHCP	Whether DHCP server is in use – 0 Disable DHCP 1 Enable DHCP
string	IPAddr	IP address of the mobile computer
string	SubnetMask	IP address of Subnet Mask
string	Gateway	IP address of Default Gateway
string	DNSAddr	Address of DNS Server – Primary DNS
string	DNSAltAddr	Address of DNS Server – Secondary DNS
string	WINSAddr	Address of WINS Server – Primary WINS
string	WINSAltAddr	Address of WINS Server – Secondary WINS

Note: When DHCP is enabled, the other IP-related information will be set to 0.

4.4.3. WLANCONNECTEDST

This structure stores information about wireless connection status, which is identical to the information displayed on the Wireless Manager settings page.

```
public struct WlanConnectedSt
{
    public string        SSID;
    public uint          SSIDLength;
    public byte[]        BSSID;

}
```

Data Type	Member Name	Description
string	SSID	Service Set ID of the access point, maximum 32 characters
uint	SSIDLength	Length of SSID, including spaces

4.4.4. WLANCTL

This structure stores information about new wireless networks, which is identical to the information displayed by tapping **Start Screen** | **Settings** | **Connections** | **Wi-Fi** | **Wireless** tab | **Add New…** button.

```
public struct WlanCtl
{
    public string       SSID;
    public uint         authentication;
    public uint         encryption;
    public uint         adhoc;
    public uint         eap;
    public byte[]       key;
}
```

Data Type	Member Name	Description
string	SSID	Maximum 32 characters
uint	authentication	Authentication in use – 0 – Open (Ndis802_11AuthModeOptn) 1 – Share key (Ndis802_11AuthModeShared) 3 – WPA (Ndis802_11AuthModeWPA) 4 – WPA-PSK (Ndis802_11AuthModeWPAPSK) 5 – WPA-NONE (Ndis802_11AuthModeWPANone) 6 – WPA2 (Ndis802_11AuthModeWPA2) 7 – WPA2-PSK (Ndis802_11AuthModeWPA2PSK)
uint	encryption	Encryption in use – 0 – WEP (Ndis802_11WEPEnabled) 1 – None (Ndis802_11WEPDisabled) 4 – TKIP (Ndis802_11Encryption2Enabled) 6 – AES (Ndis802_11Encryption3Enabled)
uint	adhoc	Network type – 0 – Infrastructure 1 – Ad hoc network
uint	eap	EAP type – 25 – PEAP

<table>
<tr><td>uint</td><td>key</td><td>WEP key in any of the following formats –<table>
<tr><td>1/0x1234567890</td><td>[index=1, 40-bit (10-digit hexa)]</td></tr>
<tr><td>4/zxcvb</td><td>[index=4, 40-bit (5-char)]</td></tr>
<tr><td>3/0x12345678901
234567890123456</td><td>[index=3, 104-bit (26-digit hexa)]</td></tr>
<tr><td>2/abcdefghij123</td><td>[index=2, 104-bit (13-char)]</td></tr>
<tr><td>auto</td><td>[key comes from successful EAP]</td></tr>
</table></td></tr>
</table>

4.4.5. CF10G_STATUS

This structure stores information about the connection status.

```
public struct CF10G_STATUS
{
    public CARDSTATE     cardState;
   public string         configName;
   public string         client_MAC;
   public string         client_IP;
   public string         clientName;
   public string         AP_MAC;
   public string         AP_IP;
   public string         APName;
   public EAPTYPE        eapType;
   public uint           channel;
   public int            rssi;
   public BITRATE        bitRate;
   public int            txPower;
   public uint           driverVersion;
   public RADIOTYPE            radioType;
   public uint           DTIM;
   public uint           beaconPeriod;
   public uint           beaconsReceived;
}
```

<table>
<tr><th>Data Type</th><th>Member Name</th><th>Description</th></tr>
<tr><td>CARDSTATE</td><td>cardState</td><td>The association status —<table><tr><td>0</td><td>CARDSTATE_NOT_INSERTED</td></tr><tr><td>1</td><td>CARDSTATE_NOT_ASSOCIATED</td></tr><tr><td>2</td><td>CARDSTATE_ASSOCIATED</td></tr><tr><td>3</td><td>CARDSTATE_AUTHENTICATED</td></tr><tr><td>4</td><td>CARDSTATE_FCCTEST</td></tr><tr><td>5</td><td>CARDSTATE_NOT_SDC</td></tr></table></td></tr>
<tr><td>string</td><td>configName</td><td>Name of active profile.
No more than 33 characters. See CONFIG_NAME_SZ defined in Cipherlab.SystemAPI.Member.</td></tr>
<tr><td>string</td><td>client_MAC</td><td>MAC address of Summit radio.</td></tr>
<tr><td>string</td><td>client_IP</td><td>IP address of Summit radio.</td></tr>
<tr><td>string</td><td>clientName</td><td>Name of Summit radio
No more than 17 characters. See CLIENT_NAME_SZ defined in Cipherlab.SystemAPI.Member.</td></tr>
<tr><td>UCHAR</td><td>AP_MAC</td><td>MAC address of the access point</td></tr>
<tr><td>UCHAR</td><td>AP_IP</td><td>IP address of the access point (= Null, if not supported by AP).</td></tr>
<tr><td>string</td><td>APName</td><td>Name of the access point (= Null, if not supported by AP).
No more than 17 characters. See CLIENT_NAME_SZ defined in Cipherlab.SystemAPI.Member.</td></tr>
<tr><td>EAPTYPE</td><td>eapType</td><td>The EAP type in use —<table><tr><td>0</td><td>EAP_NONE</td></tr><tr><td>1</td><td>EAP_LEAP</td></tr><tr><td>2</td><td>EAP_EAPFAST</td></tr><tr><td>3</td><td>PEAPMSCHAP</td></tr><tr><td>4</td><td>PEAPGTC</td></tr><tr><td>5</td><td>EAPTLS</td></tr></table></td></tr>
<tr><td>uint</td><td>channel</td><td>Channel in use — information on WLAN connection between Summit radio and AP.</td></tr>
<tr><td>int</td><td>rssi</td><td>Signal strength — information on WLAN connection between Summit radio and AP.</td></tr>
<tr><td>BITRATE</td><td>bitRate</td><td>Data rate — information on WLAN connection between Summit radio and AP.<table><tr><td>0</td><td>BITRATE_AUTO</td></tr><tr><td>2</td><td>BITRATE_1</td></tr><tr><td>4</td><td>BITRATE_2</td></tr><tr><td>11</td><td>BITRATE_5_5</td></tr></table></td></tr>
</table>

		12 BITRATE_6 18 BITRATE_9 22 BITRATE_11 24 BITRATE_12 36 BITRATE_18 48 BITRATE_24 72 BITRATE_36 96 BITRATE_48 108 BITRATE_54
int	txPower	Transmit power — information on WLAN connection between Summit radio and AP. 0 TXPOWER_MAX 1 TXPOWER_1 5 TXPOWER_5 10 TXPOWER_10 20 TXPOWER_20 30 TXPOWER_30 50 TXPOWER_50
uint	driverVersion	Driver version
RADIOTYPE	radioType	The radio type in use — 0 RADIOTYPE_BG 1 RADIOTYPE_ABG 100 RADIOTYPE_NOT_SDC 101 RADIOTYPE_NOT_SDC_1
DWORD	DTIM	A multiple of the beacon period that specifies how often the beacon contains a delivery traffic indication message (DTIM), which tells power-save client devices that a packet is waiting for them (e.g. a DTIM interval of 3 means that every third beacon contains a DTIM).
DWORD	beaconPeriod	Amount of time between access point beacons in kilomicroseconds, where one Kµsec equals 1024 microseconds.
DWORD	beaconsReceived	Beacons received.

4.4.6. CONFIG_FILE_INFO

This structure stores information about the configuration files and SDK version.

```
public struct CONFIG_FILE_INFO
{
    public uint          numConfigs;
   public byte           globalConfigPresent;
   public byte           thirdPartyConfigPresent;
   public uint           sdkVersion;
}
```

Data Type	Member Name	Description
uint	numConfigs	Number of profiles. No more than 20. See MAX_CFGS defined in Cipherlab.SystemAPI.Member.
byte	globalConfigPresent	Whether global configuration is present — 0 No 1 Yes
byte	thirdPartyConfigPresent	**(Function reserved)** Whether 3rd party configuration is present — 0 No 1 Yes
uint	sdkVersion	SDK version for Summit radio

4.4.7. SDC_ALL

This structure stores information about all the configurations.

```
public struct SDC_ALL
{
    public uint                 numConfigs;
    public SDCConfig            *configs;
    public SDC3rdPartyConfig    *configThirdParty;
    public SDCGlobalConfig      *configGlobal;
}
```

Data Type	Member Name	Description
uint	numConfigs	Number of profiles No more than 20. See MAX_CFGS defined in Cipherlab.SystemAPI.Member.
SDCConfig	*configs	Refer to SDCConfig structure.
SDC3rdPartyConfig	*configThirdParty	**(Function reserved)** Refer to SDC3rdPartyConfig structure.
SDCGlobalConfig	*configGlobal	Refer to SDCGlobalConfig structure.

4.4.8. SDCCONFIG

```
public struct SDCConfig
{
    public string        configName;
    public string        SSID;
    public string        clientName;
    public uint          txPower;
    public AUTH          authType;
    public EAPTYPE       eapType;
    public POWERSAVE     powerSave;
    public WEPTYPE       wepType;
    public BITRATE       bitRate;
    public RADIOMODE     radioMode;
    public CRYPT         userName;
    public CRYPT         userPwd;
    public CRYPT         PSK;
    public CRYPT         WEPKeys;
}
```

<table>
<tr><th>Data Type</th><th>Member Name</th><th>Description</th></tr>
<tr><td>string</td><td>configName</td><td>Name of profile.
No more than 33 characters. See CONFIG_NAME_SZ defined in Cipherlab.SystemAPI.Member.</td></tr>
<tr><td>string</td><td>SSID</td><td>Service Set Identifier (SSID) which Summit radio will connect to.
No more than 33 characters. See SSID_SZ defined in Cipherlab.SystemAPI.Member.</td></tr>
<tr><td>string</td><td>clientName</td><td>Name of Summit radio.
No more than 33 characters. See CLIENT_NAME_SZ defined in Cipherlab.SystemAPI.Member.</td></tr>
<tr><td>int</td><td>txPower</td><td>The TX power in use —<table><tr><td>0</td><td>Maximum defined for the current regulatory domain</td></tr><tr><td>1</td><td>1 mW</td></tr><tr><td>5</td><td>5 mW</td></tr><tr><td>10</td><td>10 mW</td></tr><tr><td>20</td><td>20 mW</td></tr><tr><td>30</td><td>30 mW</td></tr><tr><td>50</td><td>50 mW</td></tr></table></td></tr>
</table>

<table>
<tr><td>AUTH</td><td>authType</td><td>The authentication type in use —<table><tr><td>0</td><td>Open</td></tr><tr><td>1</td><td>Shared-key</td></tr><tr><td>2</td><td>LEAP (Network-EAP)</td></tr></table></td></tr>
<tr><td>EAPTYPE</td><td>eapType</td><td>The authentication type in use —<table><tr><td>0</td><td>EAP_NONE</td></tr><tr><td>1</td><td>EAP_LEAP</td></tr><tr><td>2</td><td>EAP_EAPFAST</td></tr><tr><td>3</td><td>PEAPMSCHAP</td></tr><tr><td>4</td><td>PEAPGTC</td></tr><tr><td>5</td><td>EAPTLS</td></tr></table></td></tr>
<tr><td>POWERSAVE</td><td>powerSave</td><td>The power saving setting —
Constantly Awake Mode (CAM) keeps the client adapter powered up continuously so there is little lag in message response time. It consumes the most power but offers the highest throughput. It is recommended when AC power is in use.
Max Power Savings (Max PSP) causes the access point to buffer incoming messages for the client adapter, which wakes up periodically and polls the access point to see if any buffered messages are waiting for it. The client adapter can request each message and then go back to sleep. It conserves the most power but offers the lowest throughput. It is recommended when battery power is in use.
Power Save Mode (Fast PSP) switches between the two modes described above, depending on network traffic. This mode switches to CAM when retrieving a large number of packets and switches back to PSP (= PS-Poll Procedure) after the packets have been retrieved. It is recommended when power consumption is a concern but you need greater throughput than that allowed by Max PSP.
<table><tr><td>0</td><td>Constantly awake mode (CAM)</td></tr><tr><td>1</td><td>Maximum power saving</td></tr><tr><td>2</td><td>Fast power save mode</td></tr></table></td></tr>
<tr><td>WEPTYPE</td><td>wepType</td><td>The WEP type in use —<table><tr><td>0</td><td>WEP_OFF</td></tr><tr><td>1</td><td>WEP_ON</td></tr><tr><td>2</td><td>WEP_AUTO</td></tr><tr><td>3</td><td>WEP_PSK</td></tr><tr><td>4</td><td>WEP_TKIP</td></tr><tr><td>5</td><td>WEP2_PSK</td></tr></table></td></tr>
</table>

		6 WEP2_AES 7 CCKM_TKIP 8 WEP_CKIP 9 WEP_AUTO_CKIP 10 CCKM_AES
BITRATE	bitRate	Data rate used by Summit radio when interacting with AP. 0 BITRATE_AUTO 2 BITRATE_1 4 BITRATE_2 11 BITRATE_5_5 12 BITRATE_6 18 BITRATE_9 22 BITRATE_11 24 BITRATE_12 36 BITRATE_18 48 BITRATE_24 72 BITRATE_36
RADIOMODE	radioMode	The radio mode in use — 0 RADIOMODE_B_ONLY 1 RADIOMODE_BG 2 RADIOMODE_G_ONLY 3 RADIOMODE_BG_LRS 4 RADIOMODE_A_ONLY 5 RADIOMODE_ABG 6 RADIOMODE_BGA
CRYPT	userName	Username for authentication (EAP). No more than 72 characters. See CRED_CA_POS defined in Cipherlab.SystemAPI.Member.
CRYPT	userPwd	Password for authentication (EAP) No more than 72 characters. See CRED_UCA_POS defined in Cipherlab.SystemAPI.Member.
CRYPT	PSK	Pre-Shared Key for encryption.
CRYPT	WEPKeys	WEP Keys for encryption.

4.4.9. SDC3RDPARTYCONFIG

This structure stores detailed information of the 3rd party configuration.

```
public struct SDC3rdPartyConfig
{
      public string     clientName;
      POWERSAVE         powerSave;
      public int        txPower;
      BITRATE           bitRate;
      RADIOMODE         radioMode;
}
```

<table>
<tr><th>Data Type</th><th>Member Name</th><th>Description</th></tr>
<tr><td>string</td><td>clientName</td><td>Name of Summit radio
No more than 33 characters. See CLIENT_NAME_SZ defined in Cipherlab.SystemAPI.Member.</td></tr>
<tr><td>POWERSAVE</td><td>powerSave</td><td>The power saving setting —
▸ Constantly Awake Mode (CAM) keeps the client adapter powered up continuously so there is little lag in message response time. It consumes the most power but offers the highest throughput. It is recommended when AC power is in use.
▸ Max Power Savings (Max PSP) causes the access point to buffer incoming messages for the client adapter, which wakes up periodically and polls the access point to see if any buffered messages are waiting for it. The client adapter can request each message and then go back to sleep. It conserves the most power but offers the lowest throughput. It is recommended when battery power is in use.
▸ Power Save Mode (Fast PSP) switches between the two modes described above, depending on network traffic. This mode switches to CAM when retrieving a large number of packets and switches back to PSP (= PS-Poll Procedure) after the packets have been retrieved. It is recommended when power consumption is a concern but you need greater throughput than that allowed by Max PSP.
<table>
<tr><td>0</td><td>Constantly awake mode (CAM)</td></tr>
<tr><td>1</td><td>Max Power Savings (Max PSP)</td></tr>
<tr><td>2</td><td>Power Save Mode (Fast PSP)</td></tr>
</table>
</td></tr>
<tr><td>int</td><td>txPower</td><td>The TX power in use —
<table>
<tr><td>0</td><td>Maximum defined for the current regulatory domain</td></tr>
<tr><td>1</td><td>1 mW</td></tr>
</table>
</td></tr>
</table>

		5 — 5 mW 10 — 10 mW 20 — 20 mW 30 — 30 mW 50 — 50 mW
BITRATE	bitRate	Data rate used by Summit radio when interacting with AP. 0 — BITRATE_AUTO 2 — BITRATE_1 4 — BITRATE_2 11 — BITRATE_5_5 12 — BITRATE_6 18 — BITRATE_9 22 — BITRATE_11 24 — BITRATE_12 36 — BITRATE_18 48 — BITRATE_24 72 — BITRATE_36
RADIOMODE	radioMode	The radio mode in use — 0 — RADIOMODE_B_ONLY 1 — RADIOMODE_BG 2 — RADIOMODE_G_ONLY 3 — RADIOMODE_BG_LRS 4 — RADIOMODE_A_ONLY 5 — RADIOMODE_ABG 6 — RADIOMODE_BGA

4.4.10. SDCGLOBALCONFIG

This structure stores detailed information of global configuration.

```
public struct _SDCGlobalConfig
{
      public uint       fragThreshold;
      public uint       RTSThreshold;
      RX_DIV            RxDiversity;
      TX_DIV            TxDiversity;
      ROAM_TRIG         roamTrigger;
      ROAM_DELTA        roamDelta;
      ROAM_PERIOD       roamPeriod;
      PREAMBLE          preamble;
      GSHORTSLOT        g_shortslot;
      BT_COEXIST        BTcoexist;
      PING_PAYLOAD      pingPayload;
      public uint       pingTimeout;
      public uint       pingDelay;
      public uint       radioState;
      public uint       displayPasswords;
      public uint       adminOverride;
      public uint       txMax;
      FCC_TEST          FCCtest;
      public uint       testChannel;
      BITRATE           testRate;
      public int        testPower;
      public uint       regDomain;
      public uint       ledUsed;
      public uint       txTestTimeout;
      public uint       WMEenabled;
      public uint       CCXfeatures;
      public string     certPath;
      CRYPT             adminPassword;
      public uint       bLRS;
      public uint       avgWindow;
      public uint       probeDelay;
      public uint       polledIRQ;
```

```
        public uint       keepAlive;
        public uint       trayIcon;
        public uint       aggScanTimer;
        public uint       authTimeout;
        public uint       autoProfile;
        public uint[]     Reserved0;
        public uint       txMaxA;
        public uint       adminFiles;
        public uint       DFSchannels;
        public uint       interferenceMode;
        public uint       authServerType;
        public uint[]     Reserved1;
}
```

Data Type	Member Name	Description
uint	fragThreshold	If packet size exceeds threshold, then it is fragmented — ▸ 256 ~ 2346 (bytes) See FRAG_LOW and FRAG_HIGH defined in Cipherlab.SystemAPI.Member.
uint	RTSThreshold	Packet size above which RTS/CTS is required on link — ▸ 0 ~ 2347 (bytes) See RTS_LOW and RTS_HIGH defined in Cipherlab.SystemAPI.Member.
RX_DIV	RxDiversity	How to handle antenna diversity when receiving data from AP — 0 — Use main antenna only 1 — Use auxiliary antenna only 2 — On startup, use auxiliary antenna 3 — On startup, use main antenna
TX_DIV	TxDiversity	How to handle antenna diversity when transmitting data to AP — 0 — Use main antenna only 1 — Use auxiliary antenna only 2 — --- 3 — Use diversity
ROAM_TRIG	roamTrigger	When the moving average RSSI from the current AP is weaker than Roam Trigger, radio does a roam scan where it probes for an AP with a signal that is at least Roam Delta dBm stronger — 50 — -50 dBm 55 — -55 dBm

		60 -60 dBm 65 -65 dBm 70 -70 dBm 75 -75 dBm 80 -80 dBm 85 -85 dBm 90 -90 dBm
ROAM_DELTA	roamDelta	When Roam Trigger is met, a second AP's signal strength (RSSI) must be Roam Delta dBm stronger than the moving average RSSI for the current AP before radio will attempt to roam to the second AP — 5 5 dBm 10 10 dBm 15 15 dBm 20 20 dBm 25 25 dBm 30 30 dBm 35 35 dBm 40 40 dBm 45 45 dBm 50 50 dBm 55 55 dBm 60 60 dBm
ROAM_PERIOD	roamPeriod	After association or roam scan (with no roam), radio will collect RSSI scan data from Roam Period seconds before considering roaming — 5 5 sec 10 10 sec 15 15 sec 20 20 sec 25 25 sec 30 30 sec 35 35 sec 40 40 sec 45 45 sec 50 50 sec 55 55 sec 60 60 sec

<table>
<tr><td>PREAMBLE</td><td>preamble</td><td>(Not implemented)</td></tr>
<tr><td>GSHORTSLOT</td><td>g_shortslot</td><td>(Not implemented)</td></tr>
<tr><td>BT_COEXIST</td><td>BTcoexist</td><td>(Not implemented)</td></tr>
<tr><td>PING_PAYLOAD</td><td>pingPayload</td><td>The amount of data to be transmitted on a pin —<table><tr><td>32</td><td>32 bytes</td></tr><tr><td>64</td><td>64 bytes</td></tr><tr><td>128</td><td>128 bytes</td></tr><tr><td>256</td><td>256 bytes</td></tr><tr><td>512</td><td>512 bytes</td></tr><tr><td>1024</td><td>1024 bytes</td></tr></table></td></tr>
<tr><td>uint</td><td>pingTimeout</td><td>The amount of time that elapses without a response before ping request is considered a failure —
0 ~ 30000 (milliseconds)
See PTIME_LOW and PTIME_HIGH defined in the header file.</td></tr>
<tr><td>uint</td><td>pingDelay</td><td>The amount of time that elapses between successive ping requests —
0 ~ 7200000 (milliseconds)
See PDELAY_LOW and PDELAY_HIGH defined in Cipherlab.SystemAPI.Member.</td></tr>
<tr><td>uint</td><td>radioState</td><td>Whether to enable radio or not —<table><tr><td>0</td><td>Disable</td></tr><tr><td>1</td><td>Enable</td></tr></table></td></tr>
<tr><td>uint</td><td>displayPasswords</td><td>(Not implemented)</td></tr>
<tr><td>uint</td><td>adminOverride</td><td>(Not implemented)</td></tr>
<tr><td>uint</td><td>txMax</td><td>(Not implemented)</td></tr>
<tr><td>FCC_TEST</td><td>FCCtest</td><td>(Not implemented)</td></tr>
<tr><td>uint</td><td>testChannel</td><td>(Not implemented)</td></tr>
<tr><td>BITRATE</td><td>testRate</td><td>(Not implemented)</td></tr>
<tr><td>int</td><td>testPower</td><td>(Not implemented)</td></tr>
<tr><td>uint</td><td>regDomain</td><td>Check the regulatory domain in use —<table><tr><td>0</td><td>FCC (Americas)</td></tr><tr><td>1</td><td>ETSI (Europe)</td></tr><tr><td>2</td><td>TELEC (Japan)</td></tr><tr><td>3</td><td>Worldwide</td></tr></table></td></tr>
<tr><td>uint</td><td>ledUsed</td><td>(Not implemented)</td></tr>
<tr><td>uint</td><td>txTestTimeout</td><td>(Not implemented)</td></tr>
<tr><td>uint</td><td>WMEenabled</td><td>Whether to allow the use of Wi-Fi Multimedia Extensions (WME) or not —</td></tr>
</table>

		0 Disable 1 Enable
uint	CCXfeatures	Whether to allow the use of three CCX features (AP-assisted roaming, AP-specified maximum transmit power, and radio management) — 0 Disable 1 Enable (for Cisco APs only)
string	certPath	File path where the certificate is stored. No more than 65 characters. See MAX_CERT_PATH defined in Cipherlab.SystemAPI.Member.
CRYPT	adminPassword	(Not implemented)
uint	bLRS	(Not implemented)
uint	avgWindow	(Not implemented)
uint	probeDelay	(Not implemented)
uint	polledIRQ	(Not implemented)
uint	keepAlive	Specify how often a null packet gets sent in seconds in CAM mode (= no power saving) — 0 Never 9 Default
uint	trayIcon	Whether to enable the system tray icon or not — 0 Disable 1 Enable
uint	aggScanTimer	Aggressive scanning complements and works in conjunction with the standard scanning that is configured through the Roam Trigger, Roam Delta, and Roam Period settings. It is recommended that aggressive scanning is enabled unless there is significant co-channel interference because of overlapping coverage from APs that are on the same channel. 0 Disable 1 Enable
uint	authTimeout	For EAP credentials: 8 seconds by default
uint	autoProfile	(Not implemented)
uint[]	Reserved0	(Reserved)
uint	txMaxA	(Not implemented)
uint	adminFiles	Whether to allow import/export settings to file — 0 No 1 Yes
uint	DFSchannels	(Not implemented)
uint	interferenceMode	Specify the interference mode —

		0 Off 1 Non-WLAN 2 WLAN 3 Auto
uint	authServerType	Specify the authentication server type — 0 ACS (type1) 1 SBP (type 2)
uint[]	Reserved1	(Reserved)

4.5. BLUETOOTH STRUCTURE

4.5.1. FTP_FILE_INFO

```
public struct Ftp_File_Info
{
   public int           fileType;
   public string        fileName;
   public int           fileSize;
}
```

Data Type	Member Name	Description
int	fileType	Specify file type of the saved file — 0 File 1 Folder
string	fileName	File name of the saved file.
int	fileSize	Size of the file name.

4.6. CAMERA STRUCTURES

4.6.1. PICSTATE

```
public struct PicState
{
   public int           AutoExposure;
   public int           AutoWhiteBalance;
   public int           ImageFormat;
   public string        lpPathName;
   public int           PathSize;
   public string        lpFileName;
   public int           FileSize;
}
```

Data Type	Member Name	Description
int	AutoExposure	Reserved
int	AutoWhiteBalance	Reserved
int	ImageFormat	The format for the image to capture– 0 JPEG 1 BMP
string	lpPathName	The directory where to save the image to capture.
int	PathSize	Size of the path name.
string	lpFileName	The name for the image to save.
int	FileSize	Size of the file name.

4.6.2. FLASH

```
public struct Flash
{
    public int          PreviewFlash;
    public int          CaptureFlash;
}
```

<table>
<tr><th>Data Type</th><th>Member Name</th><th>Description</th></tr>
<tr><td>int</td><td>PreviewFlash</td><td>Whether to turn on/off flashlight (for example, when previewing an image) –<table><tr><td>0</td><td>Turns off flashlight</td></tr><tr><td>1</td><td>Turns on flashlight</td></tr></table></td></tr>
<tr><td>int</td><td>CaptureFlash</td><td>Whether to flash a light for the image to capture –<table><tr><td>0</td><td>Turns off flashlight</td></tr><tr><td>1</td><td>Turns on flashlight</td></tr><tr><td>2</td><td>Leaves flashlight on automatic</td></tr></table></td></tr>
</table>

4.7. CAMERA STRUCTURES

4.7.1. PICSTATE

```
public struct PicState
{
   public int           AutoExposure;
   public int           AutoWhiteBalance;
   public int           ImageFormat;
   public string        lpPathName;
   public int           PathSize;
```

4.8. USER PREFERRED SETTINGS

4.8.1. USERPREFERENCESPARAMETERS

This structure stores the information about user preferred settings, such as LED illumination power level.

```
public struct UserPreferencesParameters
{
    public int          iVersion
    public int          iSize
    public int          illuminationPowerLevel
}
```

Data Type	Member Name	Description
int	iVersion	Current version number: 1
int	iSize	Size of the structure (`UserPreferencesParameters`).
int	illuminationPowerLevel	Level of LED illumination, in the range of 0 to 10. A larger value indicates higher illumination level. Default value is 10.

Appendix I

SCAN ENGINE SETTINGS

9200 Series Mobile Computer supports the following reader types. Reader availability depends on the hardware integration of the mobile computer.

Scan Engine		ID
1D	*CCD*	*SM1*
1D	*Laser*	*SE955*
2D	*2D Imager (Software decode)*	*SE4500*
RFID	*ID_MOD_MP_RFID*	

Reader combination allowed is either 1D+RFID or 2D+RFID. Each combination is able to initialize both readers at the same time (dual mode operation). For example, when you hit the physical scan key, the mobile computer reads a printed barcode in position or an RFID tag in proximity depending on which one comes first.

Note: (1) 1D and 2D scan engines don't coexist on the mobile computer because they are both barcode readers and the mobile computer allows only one barcode reader on board.
(2) Run only one reader-controlling application at a time. For example, while running MIRROR Browser, avoid running Reader Configure, Terminal Emulation, or any other application that uses ReaderDLL.

5. IN THIS CHAPTER

Symbologies Supported..... 290
RFID Tags Supported 292

SYMBOLOGIES SUPPORTED

According to the scan engine installed, the supported symbologies are listed below.

		CCD	Laser	2D
Codabar		✓	✓	✓
Code 11		✘	✓	✓
Code 39	Code 39	✓	✓	✓
	Trioptic Code 39	✘	✓	✓
	Italian Pharmacode (Code 32)	✓	✓	✓
Code 93		✓	✓	✓
Code 128	Code 128	✓	✓	✓
	GS1-128 (EAN-128)	✓	✓	✓
	ISBT 128	✓	✓	✓
Code 2 of 5	Chinese 25	✘	✓	✓
	Industrial 25 (Discrete 25)	✓	✓	✓
	Interleaved 25	✓	✓	✓
	Convert Interleaved 25 to EAN-13	✘	✓	✓
	Matrix 25	✘	✘	✓
Composite Code	Composite CC-A/B	✘	✘	✓
	Composite CC-C	✘	✘	✓
	Compostie TLC 39	✘	✘	✓
GS1 DataBar (RSS)	GS1 DataBar-14 (RSS-14)	✓	✓	✓
	GS1 DataBar Limited (RSS Limited)	✓	✓	✓
	GS1 DataBar Expanded (RSS Expanded)	✓	✓	✓
	Convert to UPC/EAN	✘	✓	✓
Inverse	Inverse 1D barcodes	✘	✘	✓
Korean 3 of 5		✘	✘	✓
MSI		✓	✓	✓
Postal Codes	Australian Postal	✘	✘	✓
	Japan Postal	✘	✘	✓
	Netherlands KIX Code	✘	✘	✓
	US Postnet	✘	✘	✓
	US Planet	✘	✘	✓
	UK Postal	✘	✘	✓
EAN/UPC	EAN-8	✓	✓	✓
	EAN-8 Extend	✓	✓	✓

	EAN-13	✓	✓	✓
	Bookland EAN (ISBN)	✓	✓	✓
	UCC Coupon Extended Code	✘	✓	✓
	ISSN EAN	✘	✘	✓
	UPC-A	✓	✓	✓
	UPC-E	✓	✓	✓
	Convert UPC-E to UPC-A	✓	✓	✓
	UPC-E1	✓	✓	✓
	Convert UPC-E1 to UPC-A	✓	✓	✓
2D Symbologies	Aztec	✘	✘	✓
	Data Matrix	✘	✘	✓
	Maxicode	✘	✘	✓
	MicroPDF417	✘	✘	✓
	MicroQR	✘	✘	✓
	PDF417	✘	✘	✓
	QR Code	✘	✘	✓

RFID TAGS SUPPORTED

RFID reader supports both reading and writing RFID data. Supported labels include ISO 14443A and ISO 15693.

The supported RFID tags are:

ID_MOD_MP_RFID		UID Only	Read Page	Write Page
ISO 14443A	Mifare Standard S50 1K	✓	✓	✓
	Mifare Standard S70 4K	✓	✓	✓
	Mifare Ultralight UL/ULC	✓	✓	✓
	Mifare DESFire	✓	---	---
	SLE66R35	✓	✓	✓
ISO 15693	ICODE SLI	✓	✓	✓
	SRF55V02P	✓	✓	✓
	SRF55V02S	✓	---	---
	SRF55V10P	✓	✓	✓
	TI Tag-it HF-I Pro/Plus	✓	✓	✓
	LRI512	✓	✓	✓
ISO 18092	Felica	✓	---	---

Note: Study the specifications of the RFID tags to read before reading.

Appendix II

SAMPLE CODE

VISUAL C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using Microsoft.WindowsCE.Forms;
using System.Runtime.InteropServices;


namespace ReaderBarcodeTest
{

    public partial class Form1 : Form
    {
        MsgWindow MsgWin;

        public Form1()
        {
            InitializeComponent();
            MsgWin = new MsgWindow(this);
            int b1 = 0;
            string ver = string.Empty;
            b1 = Reader.ReaderEngineAPI.GetDllVer(ref ver);
            label1.Text = ver;
            b1 = Reader.ReaderEngineAPI.InitReader();


        }

        public class MsgWindow : MessageWindow
        {
            public UInt32 decodeMsg = 0;
            int b1 = 0;
            byte[] lpbuf = new byte[25];
            string tmp;


            public Form1 msgform;
```

```
            public MsgWindow(Form1 msgform)
          {
              this.msgform = msgform;
              decodeMsg = Win32API.RegisterWindowMessage("WM_DECODEDATA");
          }
        protected override void WndProc(ref Message msg)
          {
              if (msg.Msg == decodeMsg)
              {
                  switch (msg.WParam.ToInt32())
                  {
                      case 7:
                          b1 = Reader.ReaderEngineAPI.GetDecodeData(ref tmp);
                          this.msgform.label4.Text = tmp;
                          break;
                      default:
                          break;
                  }

              }
              base.WndProc(ref msg);
          }
      }
  }

  public class Win32API
  {
      [DllImport("coredll.dll", SetLastError = true)]
      public static extern uint RegisterWindowMessage(string lpString);
  }
}
```

VISUAL BASIC

```
Imports System
Imports System.Collections.Generic
Imports System.ComponentModel
Imports System.Data
Imports System.Drawing
Imports System.Text
Imports System.Windows.Forms
Imports Microsoft.WindowsCE.Forms
Imports System.Runtime.InteropServices


Public Class Form1

    Private MsgWin As MsgWindow

    Public Sub New()
        InitializeComponent()
        Me.MsgWin = New MsgWindow(Me)

        Reader.ReaderEngineAPI.InitReader()
    End Sub

    Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles
MyBase.Load

        Dim Ver As String = ""
        Reader.ReaderEngineAPI.GetDllVer(Ver)
        Label1.Text = Ver
    End Sub
End Class

Public Class MsgWindow

    Inherits MessageWindow
    Private MsgForm As Form1
    Dim DecodeMsg As UInteger
    Dim Tmp As String = ""

    <DllImport("coredll.dll", CharSet:=CharSet.Auto)> _
    Private Shared Function RegisterWindowMessage(ByVal lpString As String) As UInteger
    End Function

    Public Sub New(ByVal MsgForm As Form1)
        Me.MsgForm = MsgForm
        DecodeMsg = RegisterWindowMessage("WM_DECODEDATA")
    End Sub

    Protected Overrides Sub WndProc(ByRef msg As Microsoft.WindowsCE.Forms.Message)

        'Get 1D decode data
        If msg.Msg = DecodeMsg Then
            Select Case msg.WParam.ToInt32
                Case 7
                    Reader.ReaderEngineAPI.GetDecodeData(Tmp)
                    Me.MsgForm.Label4.Text = Tmp
                Case Else
                    Exit Select
            End Select
        End If
```

```
        MyBase.WndProc(msg)
    End Sub
End Class
```

INDEX

ActivateConfig, 171
AddConfig, 173
AddWlanSsidToPreferredList, 166
Backlight Structures, 258
Beeper, 51
BklCtl, 258
BssidInfo, 261
Camera Structures, 283
CF10G_STATUS, 266
ChangeMifareKey, 36
Chinese2Of5_1D_SE955, 87
Chinese2Of5_2D_SE4500, 113
ClearSigArea, 247
CloseSigScreen, 246
Codabar_1D_SE955, 88
Codabar_1D_SM1, 67
Codabar_2D_SE4500, 114
Code11_1D_SE955, 94
Code11_2D_SE4500, 120
Code128_1D_SE955, 93
Code128_1D_SM1, 72
Code128_2D_SE4500, 118
Code39_1D_SE955, 82
Code39_1D_SM1, 62
Code39_2D_SE4500, 107
Code93_1D_SE955, 84
Code93_1D_SM1, 64
Code93_2D_SE4500, 109
Composite_2D_SE4500, 123
CONFIG_FILE_INFO, 269
ConnectByMacAdd, 215
CreateConfig, 176
DataOutputSettings, 28
DeleteConfig, 174
Development Tool, 2
EanJan_1D_SE955, 80
EanJan_1D_SM1, 60
EanJan_2D_SE4500, 105
EnableCamera, 227
EnablePreviewWindow, 230
EnableSigScreen, 247
ExportSettings, 181
FindNextBTDevice, 209
FindService, 211
Flash, 284
FlushAllConfigKeys, 180
FlushConfigKeys, 180
GetActiveDevice, 23
GetAllConfigs, 178
GetAMicGain, 157
GetBacklightCTL, 139
GetBacklightLV, 141
GetBTConnStatus, 217
GetButtonAssignment, 239
GetCameraResolution, 228
GetConfig, 176
GetConfigFileInfo, 181
GetCurrentConfig, 171
GetCurrentDomain, 164
GetCurrentStatus, 169
GetDecodeData, 38
GetDecodeRfidData, 44
GetDecodeRfidUID, 44
GetDecoderVersion, 53
GetDecodeTagType, 45
GetDecodeType, 39
GetDeviceAddress, 212
GetDeviceName, 213
GetDllVer, 52
GetEAPFASTCred, 192
GetEAPTLSCred, 200
GetEAPTTLSCred, 203
GetErrorCode, 54, 253
GetFeature, 232
GetFile, 222
GetFlashLight, 229
GetGlobalSettings, 179
GetHALUUID, 133
GetHSMicGain, 159
GetKeylightOnOff, 146
GetKeypadLock, 150
GetKeypadMode, 152
GetKeypadSensitivity, 149
GetKeypadState, 154
GetLEAPCred, 190
GetMultipleWEPKeys, 185
GetNETAPIVersion, 132
GetNETDLLVersion, 52
GetNextFile, 221
GetNumConfigs, 172
GetOutputRecord, 31
GetPEAPGTCCred, 194

GetPEAPMSCHAPCred, 197
GetPreviewLocation, 230
GetPSK, 188
GetReaderType, 25
GetSDKVersion, 164
GetServiceType, 214
GetSigNetDllVer, 252
GetSppComPort, 224
GetStillCaptureState, 234
GetSysDevName, 134
GetSysInfo, 135
GetTchLock, 147
GetTorchMode, 237
GetVibratorPower, 138
GetWEPKey, 183
GetWiFiBSSIDList, 170
GetWiFiMac, 165
GetWiFiPower, 160
GetWlanAvailableList, 167
GetWlanConnectedStatus, 169
GetWlanIpInfo, 162
GetWlanPreferredList, 168
GS1_DataBar_1D_SE955, 92
GS1_DataBar_1D_SM1, 71
GS1_DataBar_2D_SE4500, 117
HaltGraph, 233
ImportSettings, 182
Industrial2Of5_1D_SE955, 86
Industrial2Of5_1D_SM1, 66
Industrial2Of5_2D_SE4500, 111
InitReader, 22
InitSigScreen, 245
Interleaved2Of5_1D_SE955, 85
Interleaved2Of5_1D_SM1, 65
Interleaved2Of5_2D_SE4500, 110
Inverse1D_2D_SE4500, 124
Keypad Structures, 260
KeyPadBind, 260
Korean3Of5_2D_SE4500, 125
LoadSigImage, 250
Matrix2Of5_2D_SE4500, 112
MiscellaneousOption_1D_SE955, 75
MiscellaneousOption_2D_SE4500, 98
ModifyConfig, 177
Msi_1D_SE955, 90
Msi_1D_SM1, 69
Msi_2D_SE4500, 115
NotificationSettings, 50
PicState, 283
PostalCode_2D_SE4500, 121
PutFile, 223
RadioDisable, 161
RadioEnable, 161
ReadBarcodeData, 42
ReadRfidData, 46
ReconnectWlanPreferredList, 170
ResetReaderToDefault, 129
ResetWlanPreferredList, 168
RfidSettings, 33
SaveSigImage, 251
ScanWiFiBSSID, 170
SDC_ALL, 270
SDC3rdPartyConfig, 274
SDCConfig, 271
SDCGlobalConfig, 276
SeKeypadLock, 151
SetActiveDevice, 24
SetAllConfigs, 178
SetAMicGain, 156
SetBacklightCTL, 140
SetBacklightDefault, 143
SetBacklightLV, 142
SetBacklightMax, 144
SetBacklightMin, 145
SetBTConnStatus, 218
SetButtonAssignment, 241
SetCameraResolution, 228
SetDefaultConfigValues, 175
SetEAPFASTCred, 192
SetEAPTLSCred, 201
SetEAPTTLSCred, 204
SetFeature, 232
SetFlashLight, 229
SetGlobalSettings, 179
SetHSMicGain, 158
SetInitLoaderBind, 136
SetKeylightOnOff, 146
SetKeypadMode, 153
SetKeypadSensitivity, 149
SetKeypadState, 155
SetLEAPCred, 191
SetLEDLight, 137
SetMultipleWEPKeys, 186
SetPEAPGTCCred, 195
SetPEAPMSCHAPCred, 198
SetPreviewLocation, 231
SetPSK, 188
SetSigBgColor, 248
SetSigLineColor, 249
SetSigLineWidth, 249
SetSppComPort, 225
SetStillCaptureState, 235
SetSysDevName, 134
SetTchLock, 148
SetTorchMode, 237
SetVibratorPower, 138

SetWEPKey, 184
SetWiFiPower, 160
SetWlanIpInfo, 163
ShowSigScreen, 247
StartBluetooth, 208
StartBrowse, 220
StartFindingDevice, 209
StartGraph, 233
StartPairing, 210
StartStillCapture, 236
StartUnpairing, 210
StopBluetooth, 208
Symbologies_2D_SE4500, 126
SymbologySecurityLevel_2D_SE4500, 96
SysInfo, 256
System Information Structures, 256
SystemSoftReset, 136
TerminateGraph, 233
Upc_1D_SE955, 77
Upc_1D_SM1, 57
Upc_2D_SE4500, 102
UserPreferences_1D_SE955, 73
UserPreferences_1D_SM1, 55
UserPreferences_2D_SE4500, 95
Wi-Fi Structures, 261
WlanAdptInfo, 262
WlanConnectedSt, 263
WlanCtl, 264
WriteRfidData, 48