CipherLab
User Guide

.NET Programming

For 9600 Series Mobile Computers

Version 1.08

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

RELEASE NOTES

Version	Date	Notes
1.08	Jun. 14, 2011	▸ New: 3.6.3 Screen Lock ▸ Modified: 3.11 GSM/GPRS — update flowchart ▸ New: 3.11.3 GSM Signal Strength ▸ Modified: 3.11.5 CipherLab GSM-Connect — add SetGPRSConfig()
1.07	May 10, 2011	▸ Modified: 3 System API — add Signature DLL file information ▸ Modified: 3.15.6 Signature DLL Version — remove GetSigDllVer()
1.06	Mar. 30, 2011	▸ New: 3.15 Signature Capture
1.05	Jan. 28, 2011	▸ Modified: 2.6.1 Preferences — UserPreferences_1D() timeout applied to Laser and Auto Off modes only ▸ New: 3.1.7 CipherLab Original Device — GetCipherlabDeviceID(), GetCipherlabPlatformName() ▸ Modified: 3.10.5 Bluetooth Connection & Status — add ConnectByMacAddr() ▸ New: 3.10.9 CipherLab BT-Connect — GetBTService(), SetBTService() ▸ New: 3.11.5 CipherLab GSM-Connect — GetGSMService(), SetGSMService() ▸ Modified: 3.13 Button Assignment — add support for user-defined key code
1.04	Oct. 06, 2010	▸ Modified: Correct typos
1.03	Jul. 09, 2010	▸ Modified: 2.7.13 Code128_2D_4507 — enableIsbtConcatenation, isbtConcatenationRedundancy parameters appended for ISBT 128 ▸ Modified: 2.7.18 MacroPDF_2D_4507 — modeTransmitDecode parameter updated (2* → 4*) ▸ Modified: 3 System API — COM 7 for serial communications ▸ Modified: 3.1.3 Device Name — list invalid device name ▸ Modified: 3.2.4 Reset Backlight to Default — update defaults ▸ Modified: 3.4.2 Display Type — add "For the change to take effect, it will automatically warm boot the system." for SetDisplayType() ▸ Modified: 3.9.3 Domain, SDK & MAC Information — add error code for GetWiFiMac ▸ Modified: 3.10.5 Bluetooth Status — Parameter for BT_OPP_SERVICE is reserved ▸ New: 3.10.8 SPP COM Port — add GetSppComPort, SetSppComPort ▸ Modified: 3.11.3 Information — GetGPRSManufacturer ▸ Modified: 3.14 Camera — update APIs ▸ Modified: 3.14.2 Camera Settings — remove GetCameraExposureValue, SetCameraExposureValue ▸ Modified: 3.14.3 Preview — SetPreviewLocation (size of the preview window)

		▸ Modified: 4.1.1 SysInfo — SerialNumPCBA ▸ Modified: 4.1.1 SysInfo — remove Bootloader, MicroP firmware from note ▸ Modified: 4.4.5 SDCConfig — add description for power-saving modes ▸ Modified: 4.4.6 SDC3rdPartyConfig — add description for power-saving modes ▸ Modified: 4.6.2 PICSTATE — remove ResX,Y (240, 180)
1.02	Jan. 11, 2010	▸ Modified: 2.2.3 Barcode Data — ReadBarcodeData() ignore parameter “timeout” ▸ Modified: 2.4.1 Notification Settings — Range for Good Read LED changed ▸ Modified: 2.4.3 Vibrator — removed (SetVibrator) ▸ Modified: 2.6.10 ISBT128_1D — changes to Isbt128_1D
1.01	Jan. 04, 2010	▸ Modified: 2.1.3 Reader Type — Two error conditions ▸ Modified: 2.6.20 UpcE_1D — supports UPC-E1
1.00	Dec. 21, 2009	Initial release

CONTENTS

INTRODUCTION

CipherLab 9600 .NET Programming Guide provides necessary information on dealing with the reader configuration, the retrieval of decoded data, as well as the system configuration of 9600 Series Mobile Computers, in the application development process.

Using the .NET Framework, it is easy to write applications to control the reader and retrieve the data decoded, as well as hardware resources. All you need to do is add the desired .NET component (.NET DLL Class Library) to your .NET Smart Device Applications. The two DLL files provided on the CD-ROM are **Reader_Ce_Net.dll** and **SystemApi_Ce_Net.dll**.

This programming guide is meant for users who write Windows CE application programs for CipherLab 9600 Mobile Computers. 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

It is assumed that the programmer has prior knowledge of Windows programming. For information on Windows application programming interface (API), essential online resource is available on Microsoft website –

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

For the 9600 mobile computer is running on Windows CE 6.0, the development environment and tools can be one of the following choices:

- Visual Studio 2008
- Visual Studio 2005

Chapter 1

CREATING APPLICATIONS

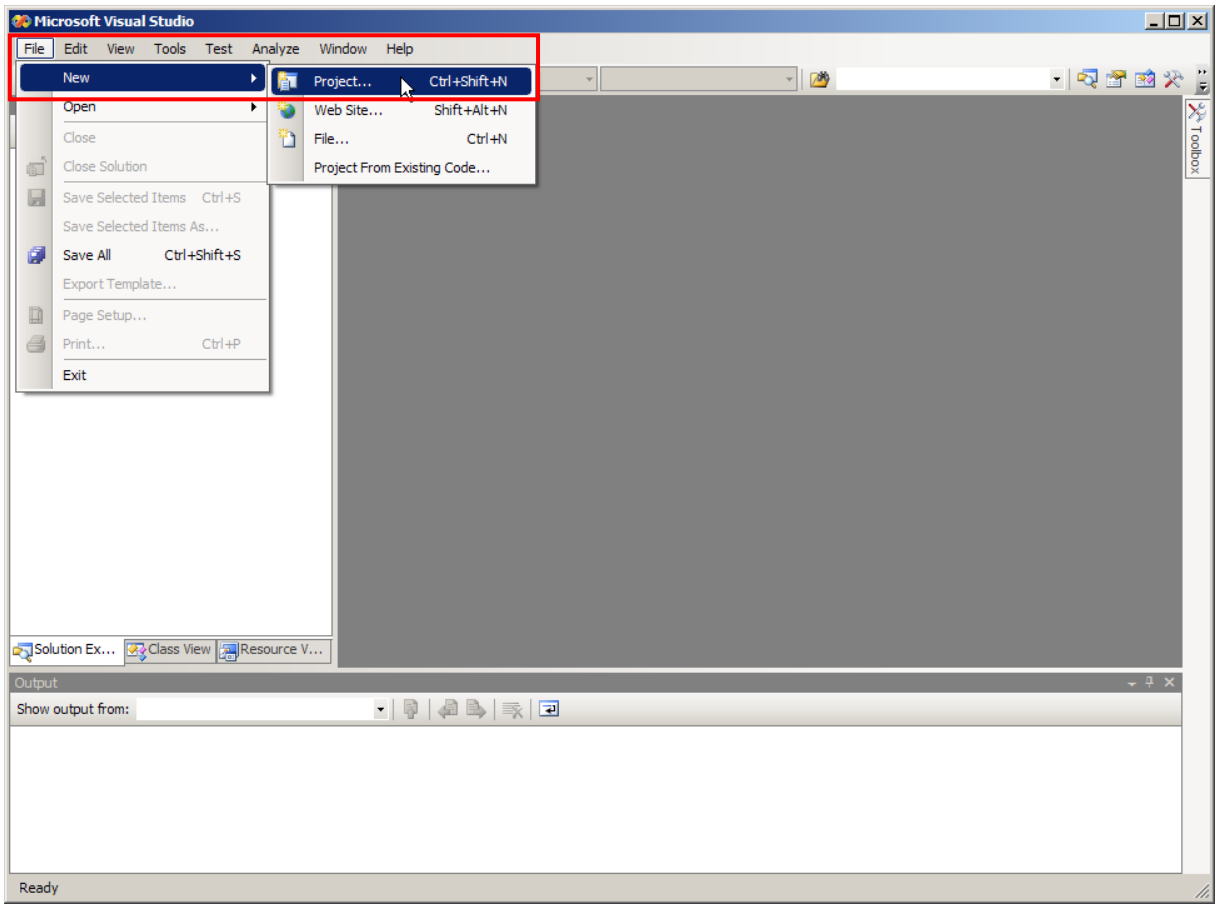
We provide the two DLL files **Reader_Ce_Net.dll** and **SystemApi_Ce_Net.dll**. Find the necessary file on the CD-ROM before you start to write your application.

IN THIS CHAPTER

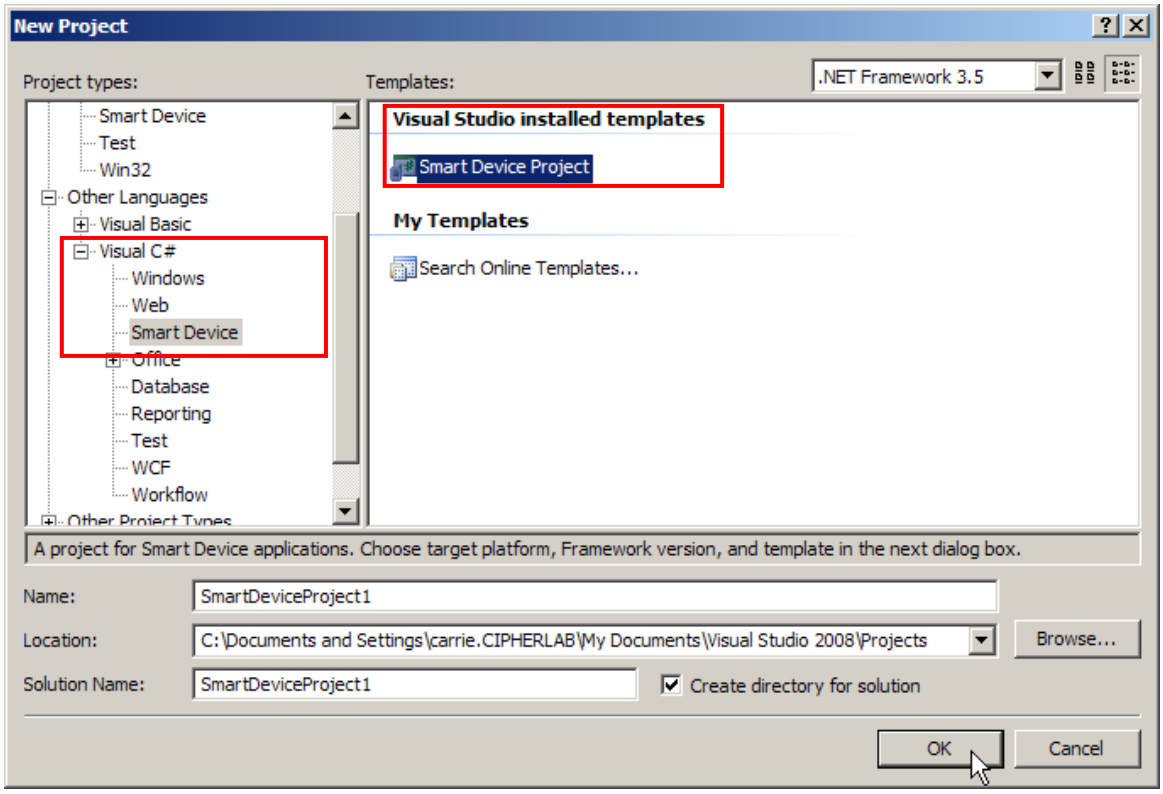
1.1 CREATE A VISUAL C# APPLICATION

Follow these steps to create a Visual C# application in Visual Studio 2008. Refer to Appendix II Sample Code.

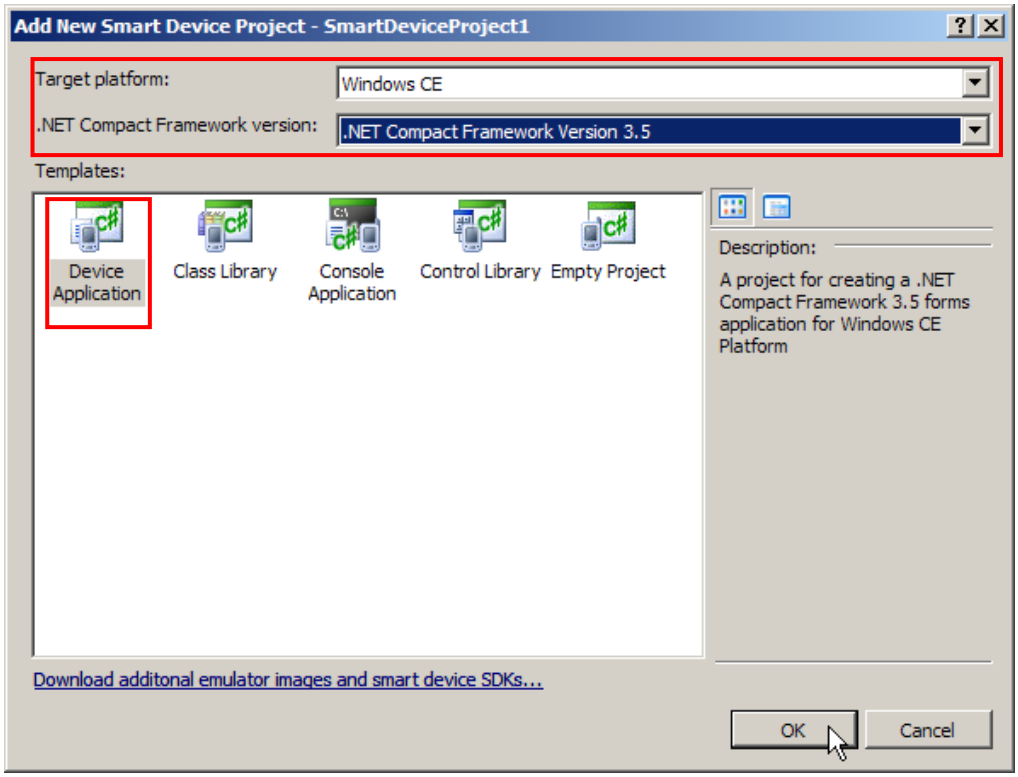
I) On the **File** menu, point to **New**, and then click **Project**.



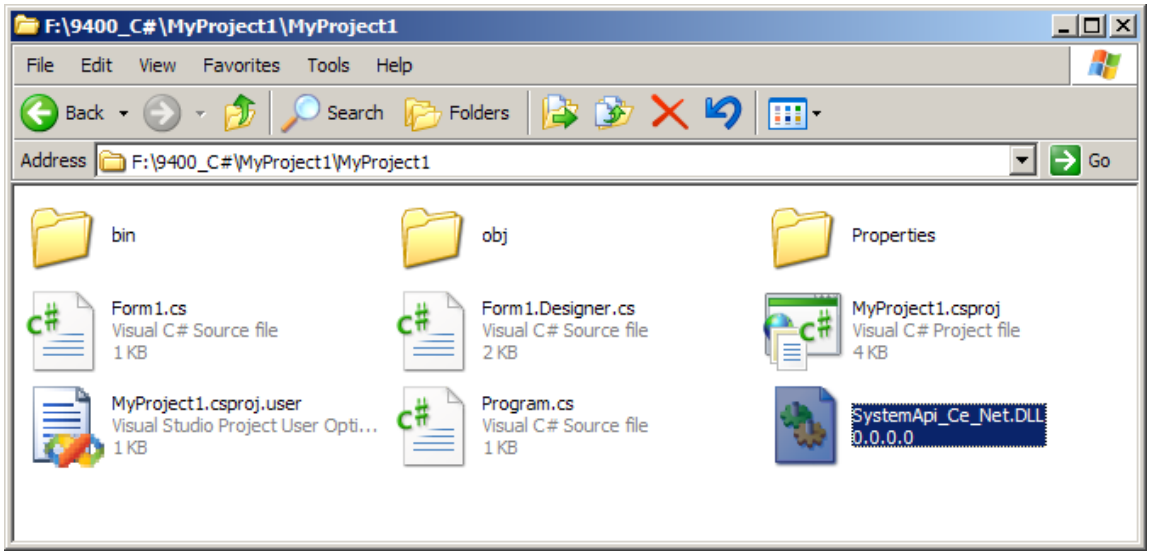
2) Select the **Smart Device** project type and the **Smart Device Project** template.



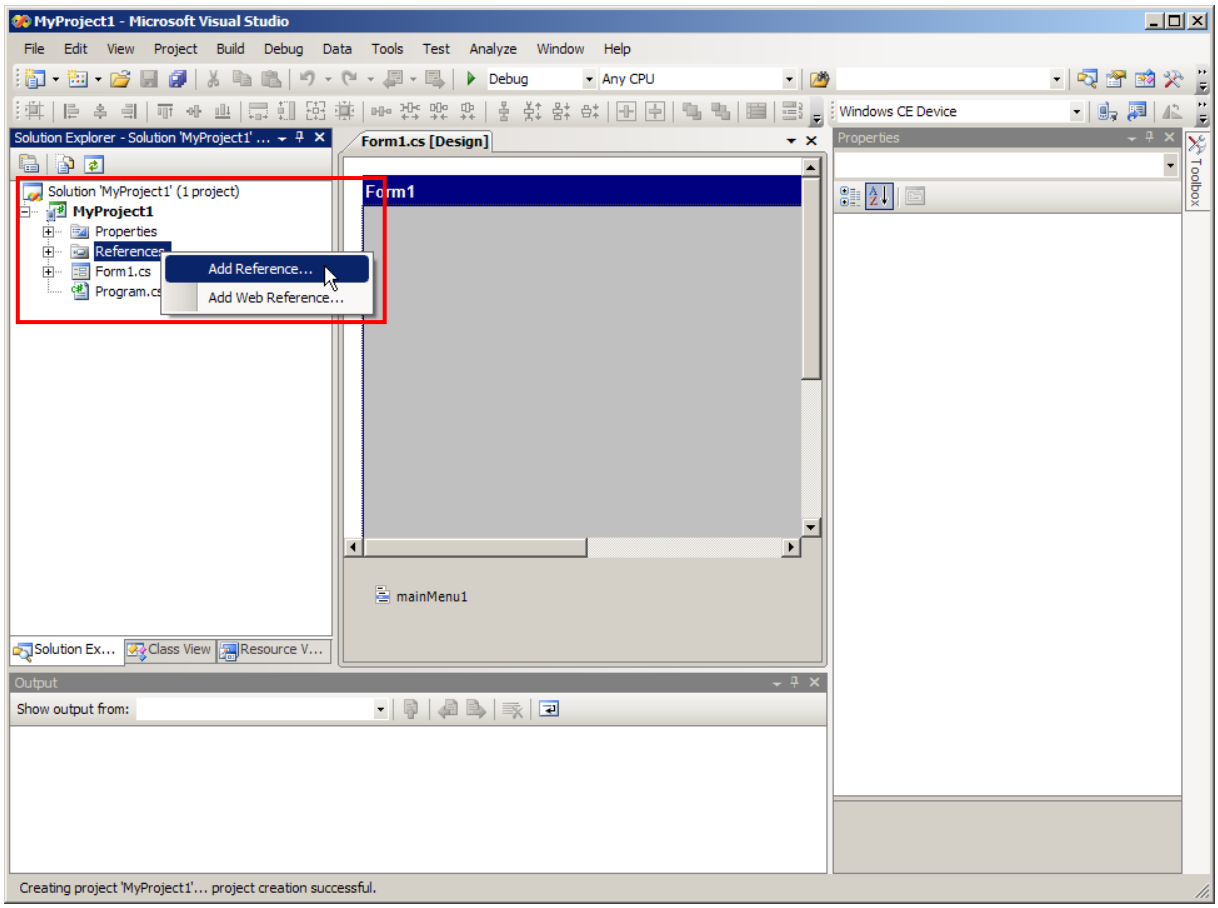
3) Select **Windows CE** for the target platform and **Device Application** for the template to use.



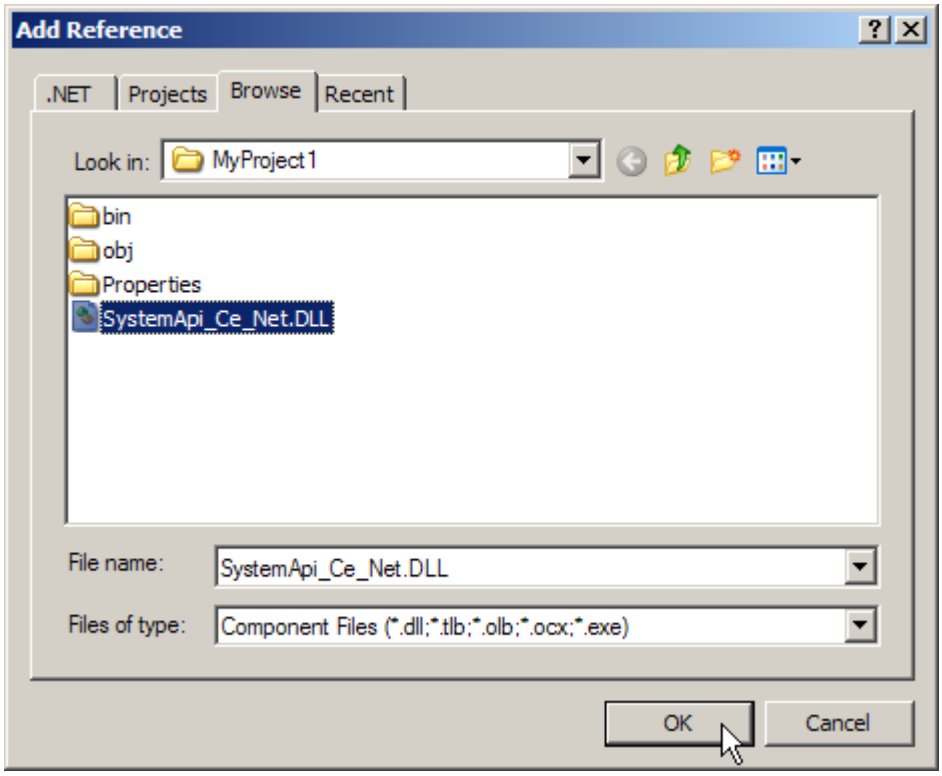
4) Copy the DLL file to the project folder.
   - SystemApi_Ce_Net.dll
   - Reader_Ce_Net.dll



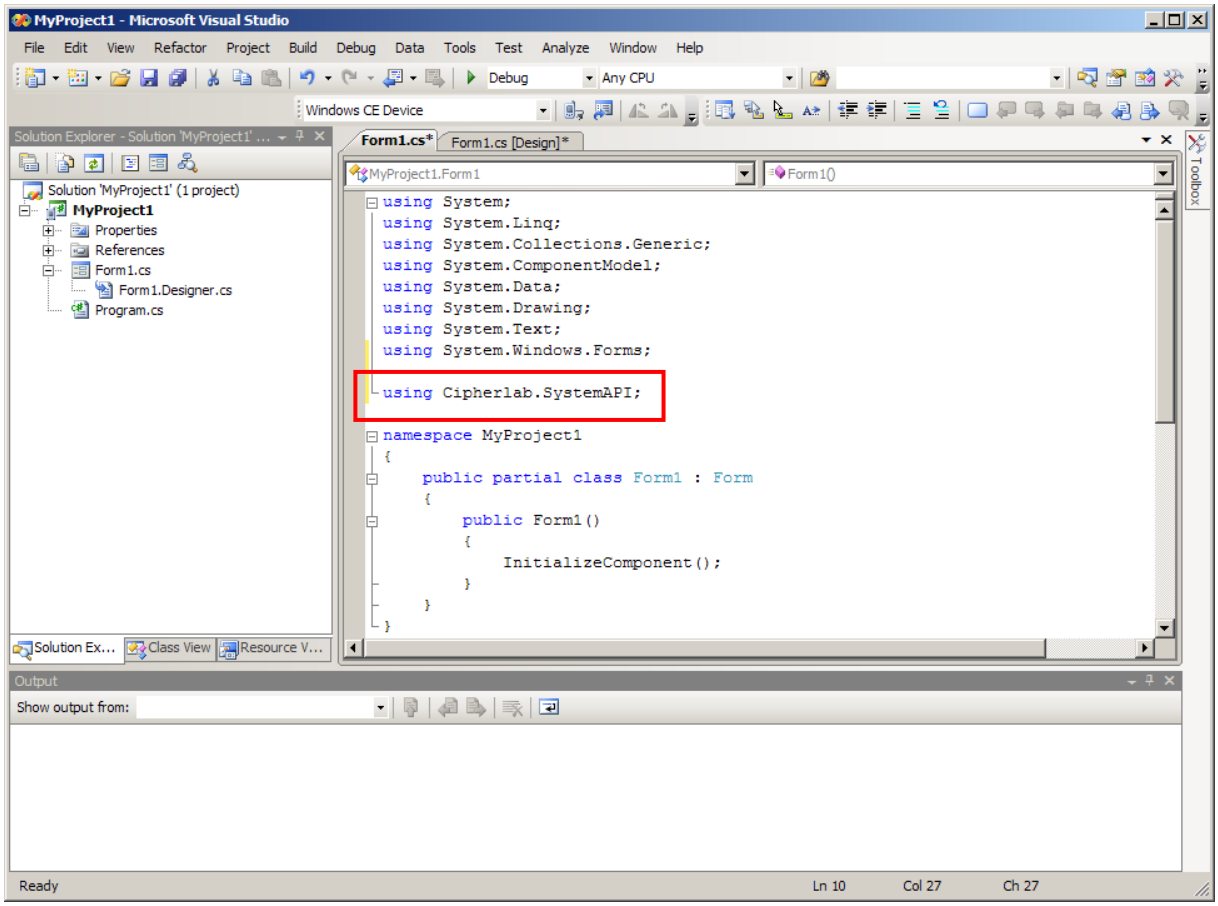
5) Right-click **References** and select **Add Reference**.



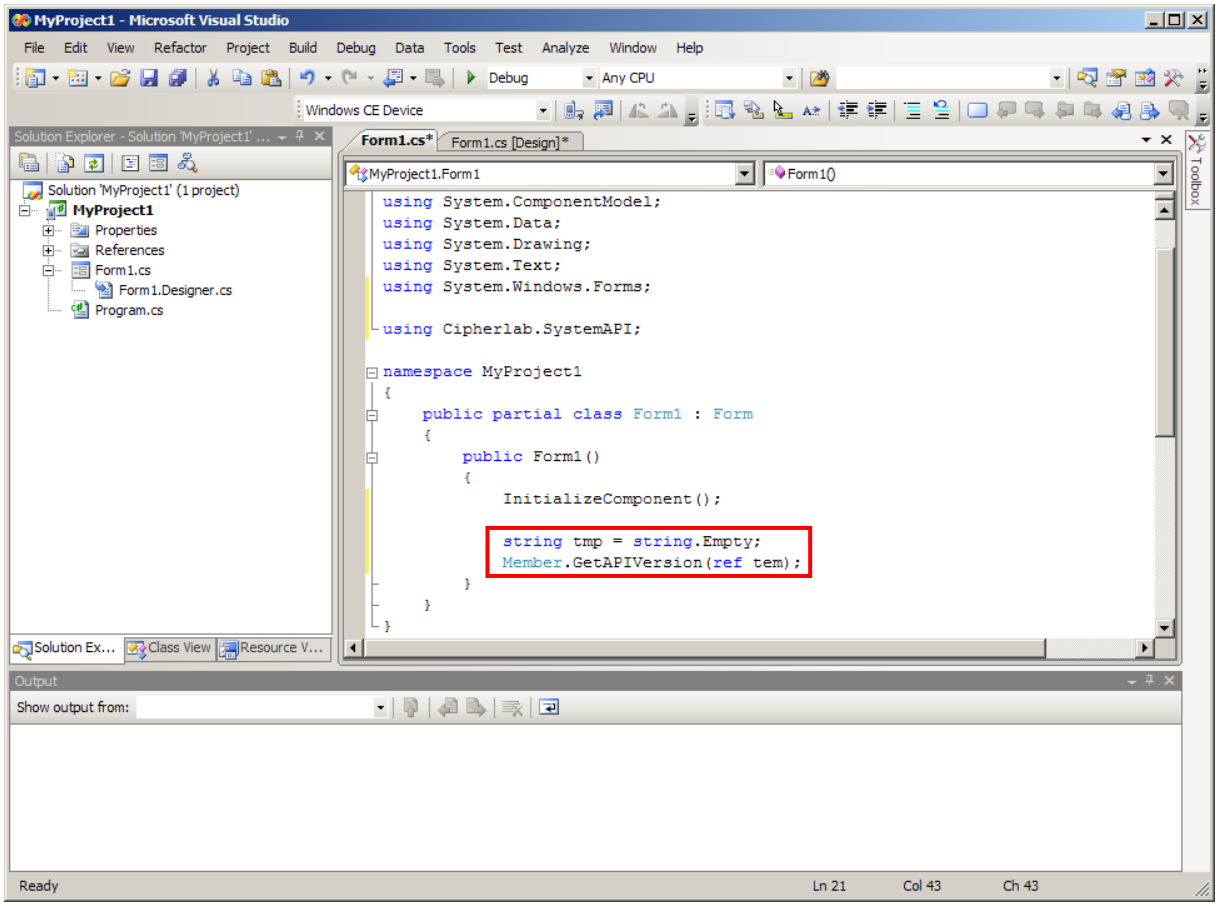
6) Click on the **Browse** tab and locate the DLL file. Select the file and click **OK**.



7) You should include "using Cipherlab.SystemAPI" to use the System DLL file.



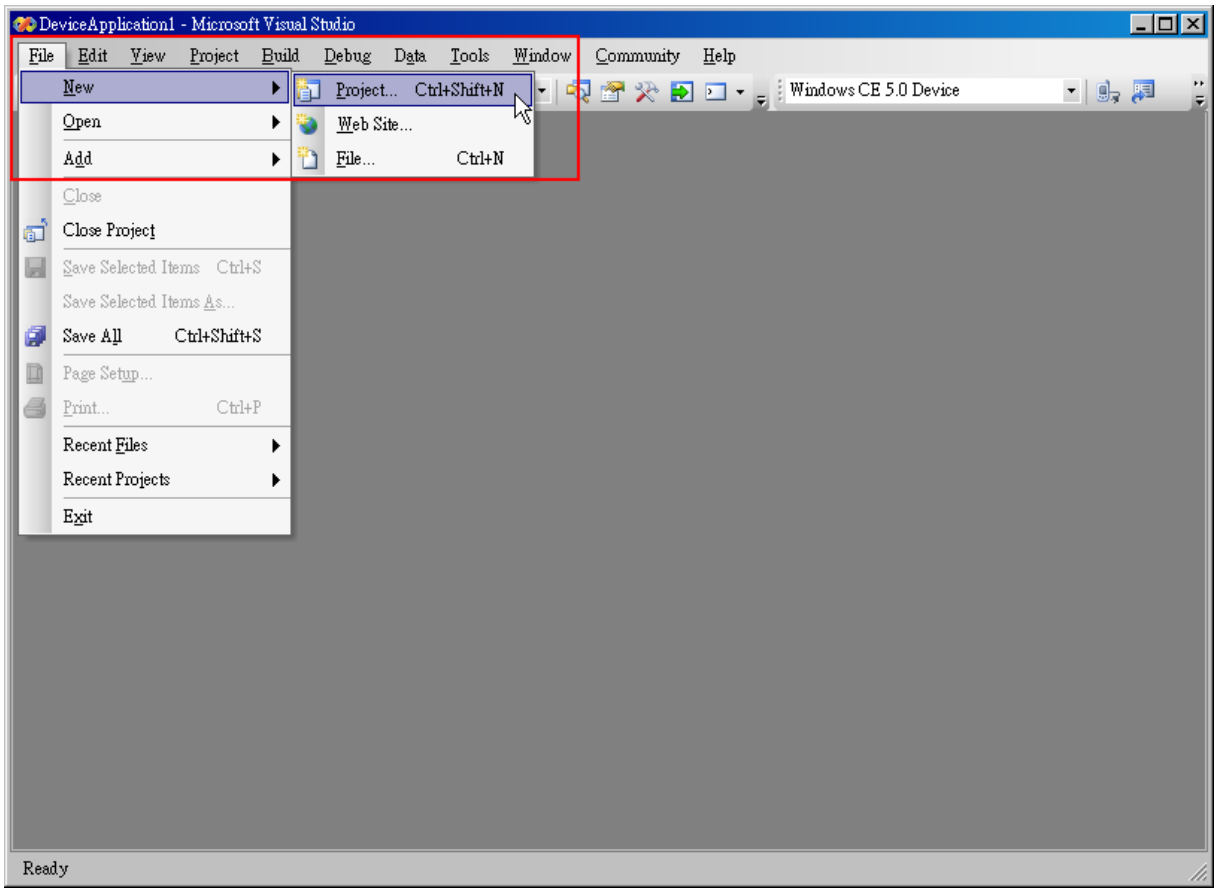
8) Start to write your code…



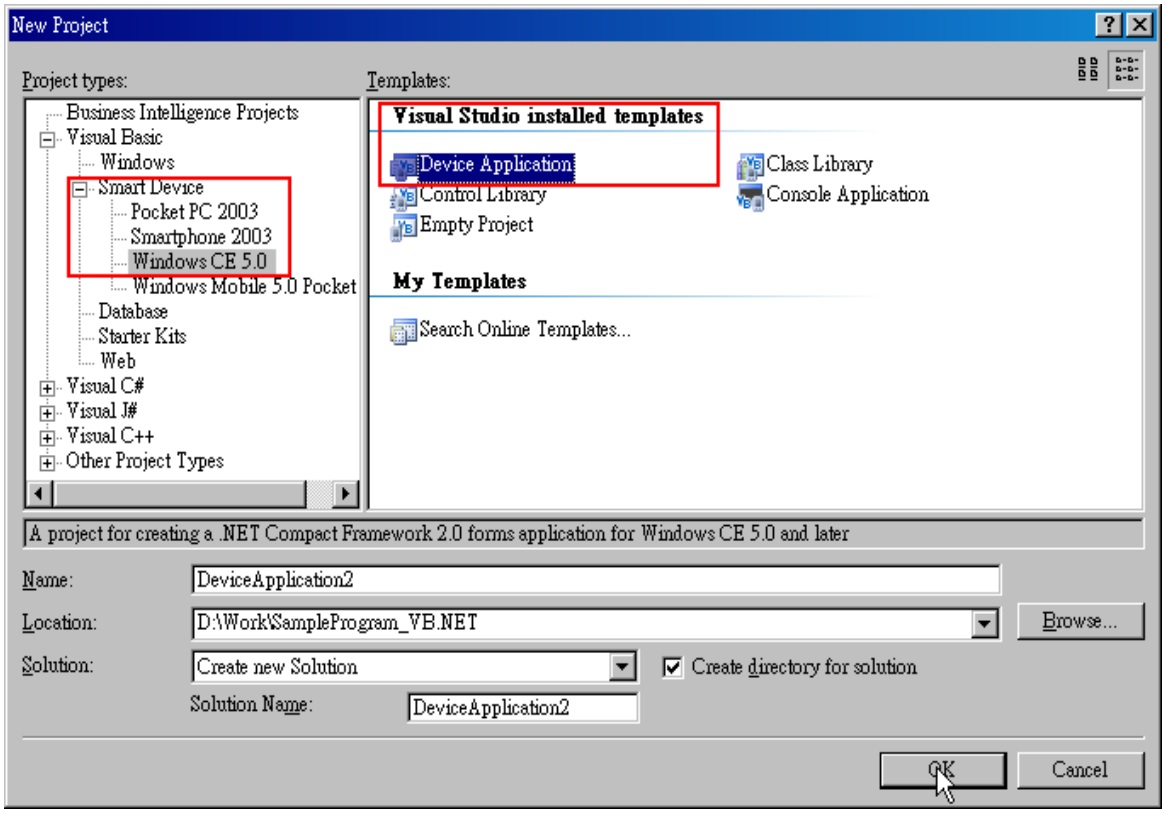
1.2 CREATE A VISUAL BASIC APPLICATION

Follow these steps to create a Visual Basic application in Visual Studio 2005. Refer to Appendix II Sample Code.

I) On the **File** menu, point to **New**, and then click **Project**.

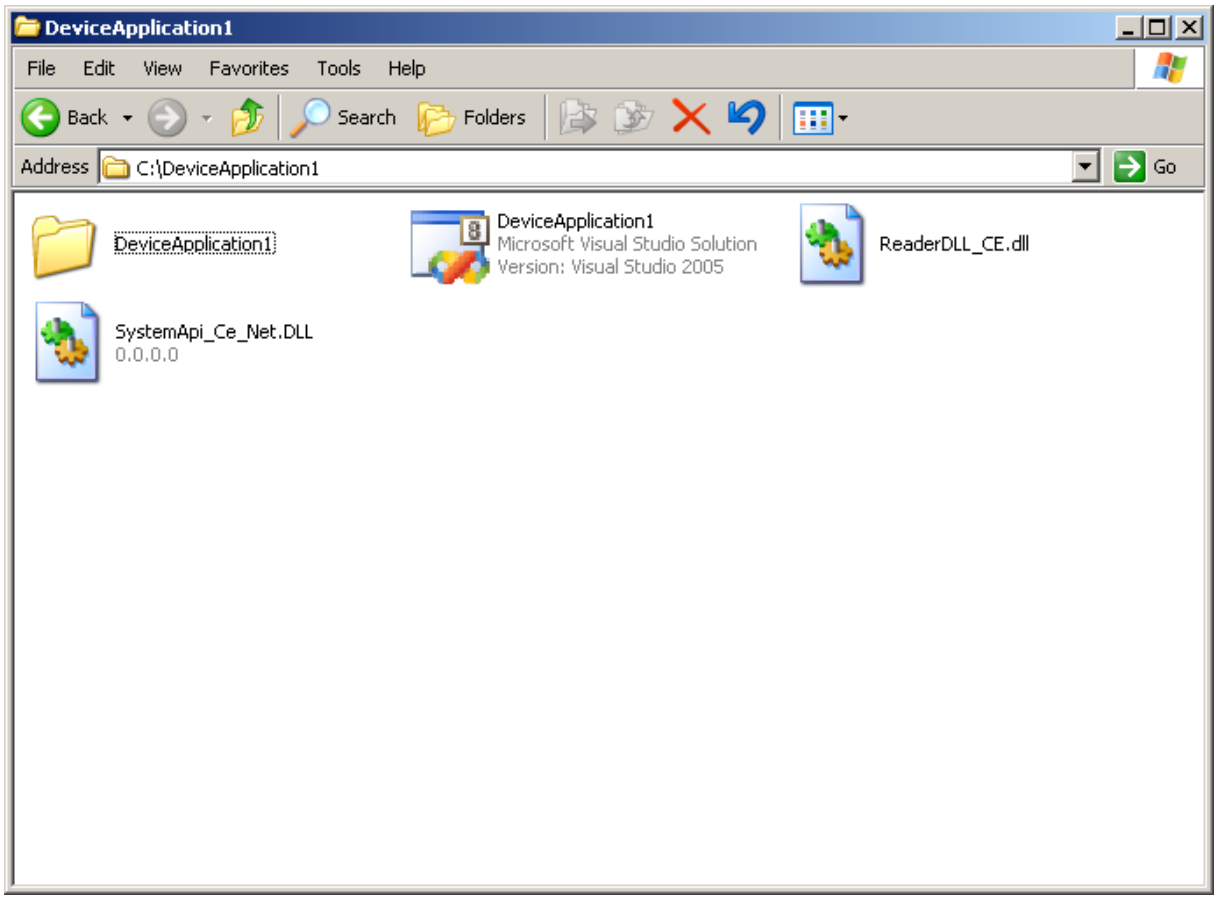


2) Select the **Smart Device** project type and the **Device Application** template.

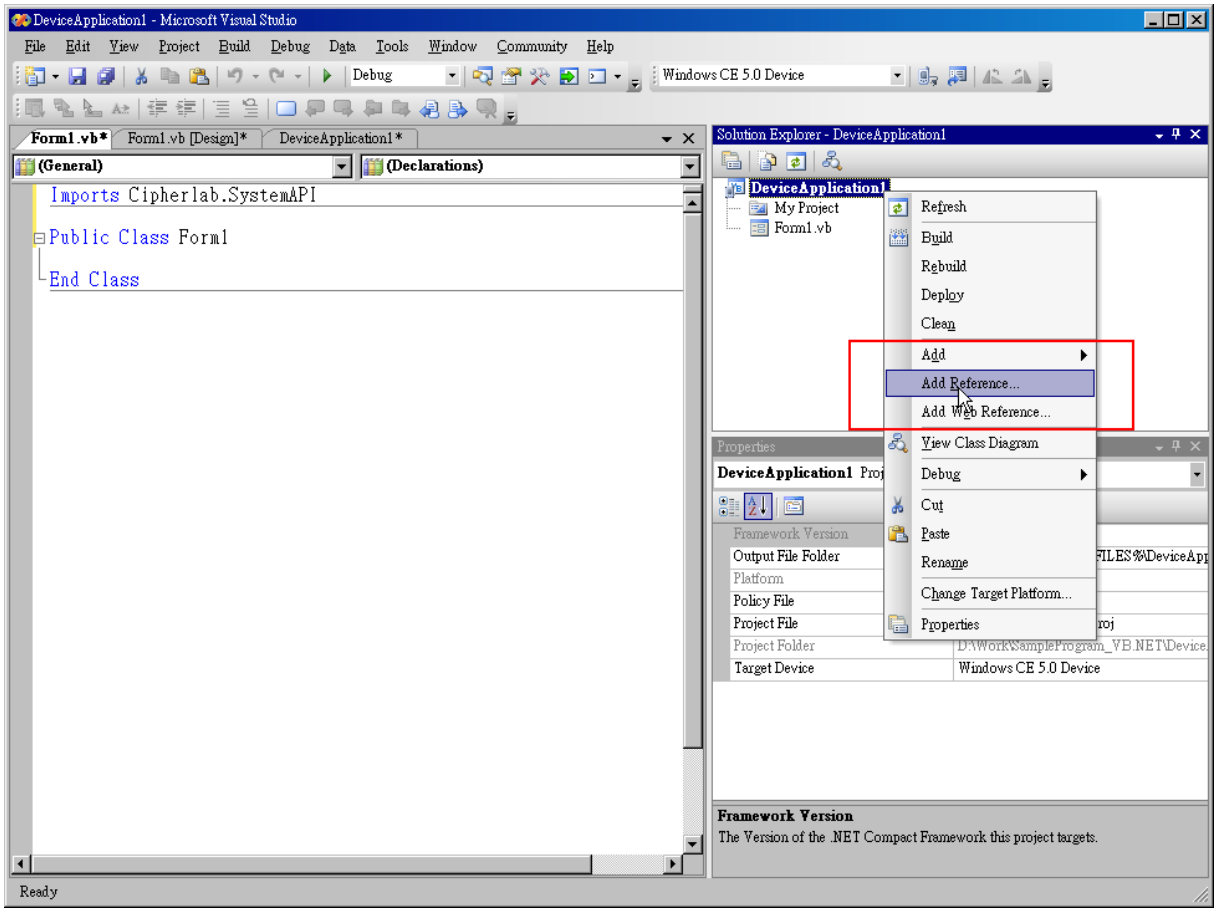


3) Copy the DLL file to the project folder.

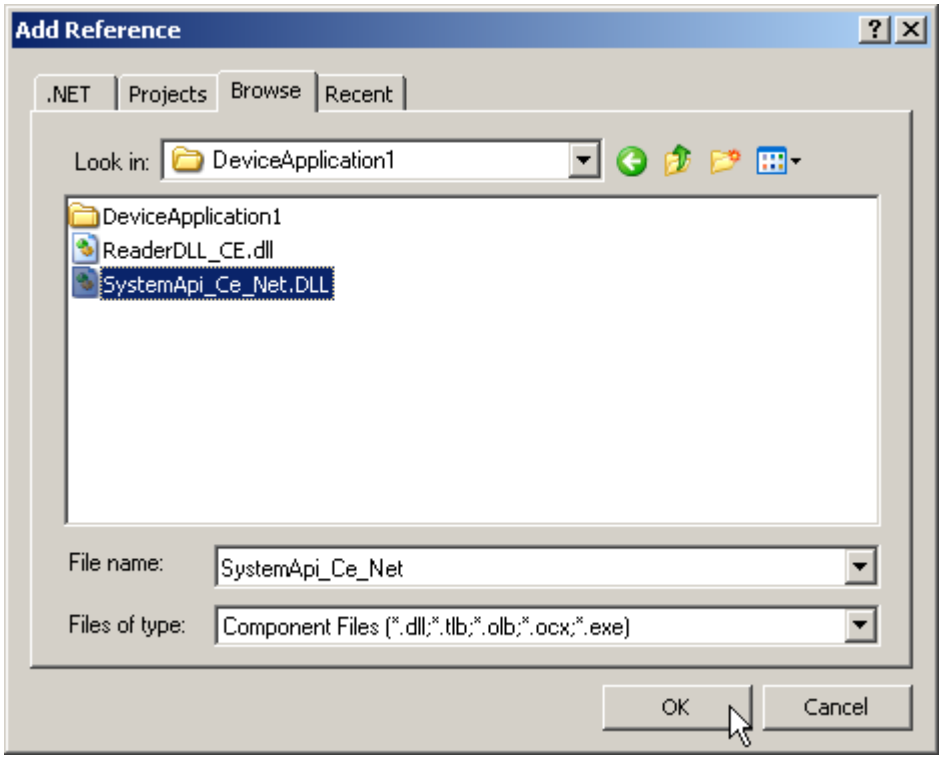
- SystemApi_Ce_Net.dll
- Reader_Ce_Net.dll



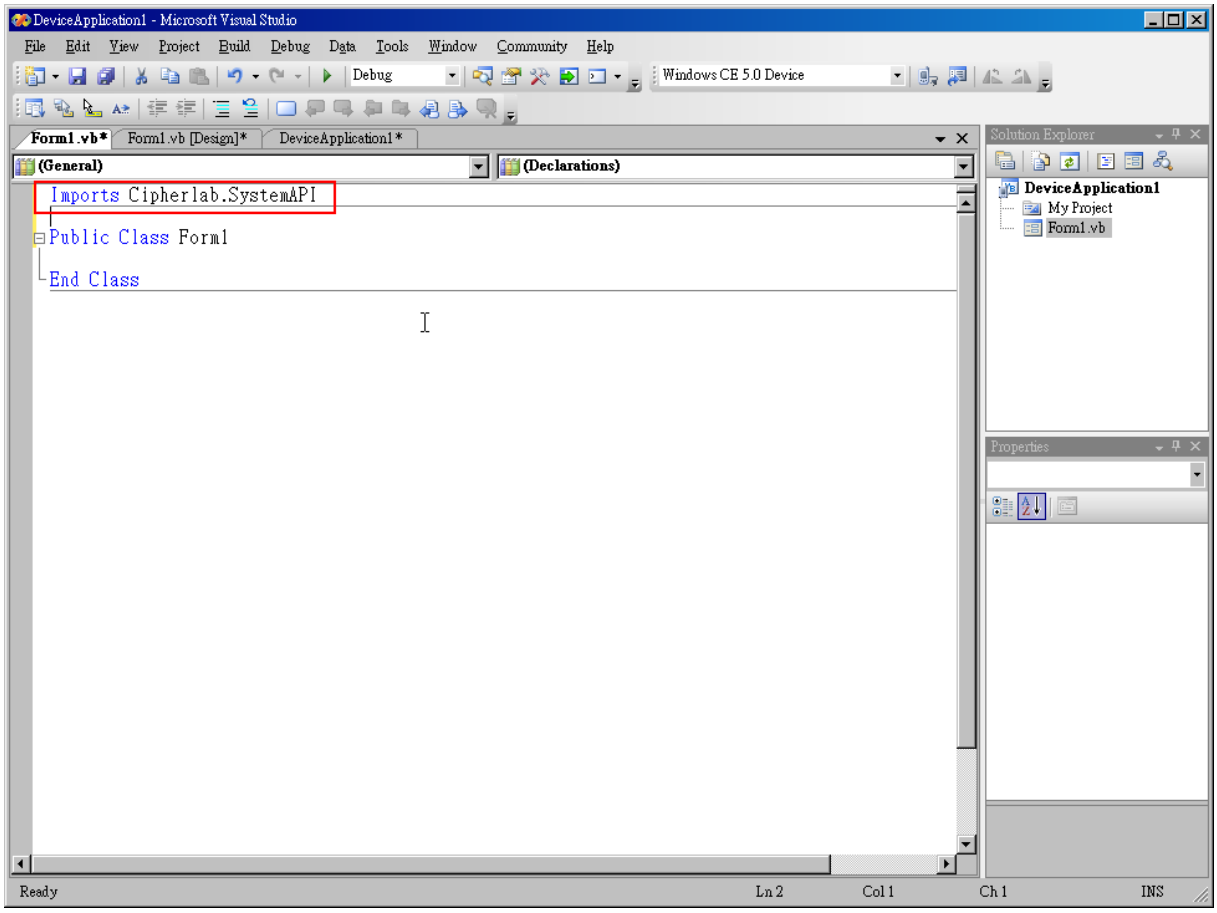
4) Right-click **Device Application1** and select **Add Reference**.



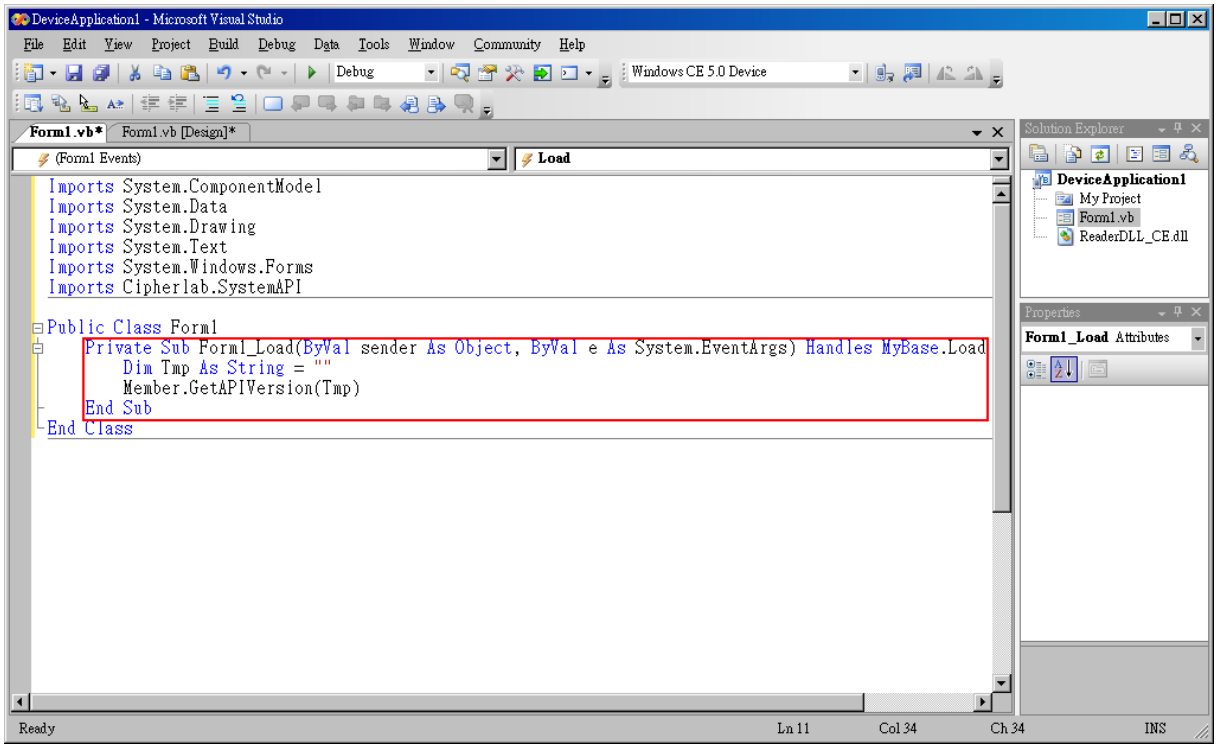
5) Click on the **Browse** tab and locate the DLL file. Select the file and click **OK**.



6) You should include "using Cipherlab.SystemAPI" to use the System DLL file.



7) Start to write your code…



1.3 REGISTER DECODE MESSAGE

1.3.1 VISUAL C#

First, register the *WM_DECODEDATA* or *WM_DECODEIMAGE* message.

For example,

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

where *decodeMsg* is the message identifier to be broadcasted when the scan engine or RFID reader decodes data.

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

where *decodeMsg* is the message identifier to be broadcasted when the 2D scan engine captures an image.

Then, when receiving the identifier of the *WM_DECODEDATA* message or the *WM_DECODEIMAGE* message in your application,

- a *WM_DECODEDATA* message is broadcasted when the scan engine or RFID reader decodes data.

  a *WM_DECODEIMAGE* message is broadcasted when the 2D scan engine captures an image.
- retrieve the *WM_DECODEDATA* or *WM_DECODEIMAGE* message in your application and use *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);
}
```

1.3.2 VISUAL BASIC

First, register the *WM_DECODEDATA* or *WM_DECODEIMAGE* message.

For example,

```
decodeMsg = RegisterWindowMessage("WM_DECODEDATA")
```

where *decodeMsg* is the message identifier to be broadcasted when the scan engine or RFID reader decodes data.

```
decodeMsg = RegisterWindowMessage("WM_DECODEIMAGE")
```

where *decodeMsg* is the message identifier to be broadcasted when the 2D scan engine captures an image.

Then, when receiving the identifier of the *WM_DECODEDATA* message or the *WM_DECODEIMAGE* message in your application,

- a *WM_DECODEDATA* message is broadcasted when the scan engine or RFID reader decodes data.

  a *WM_DECODEIMAGE* message is broadcasted when the 2D scan engine captures an image.
- retrieve the *WM_DECODEDATA* or *WM_DECODEIMAGE* message in your application and use *wParam* to identify the decoded data.

For example,

```
Protected Overrides Sub WndProc(ByRef msg As Microsoft.WindowsCE.Forms.Message)

       If msg.Msg = DecodeMsg1D Then

              Select Case msg.WParam.ToInt32

                 Case 7:
                    Reader.ReaderEngineAPI.GetDecodeData(tmp)
                    Reader.ReaderEngineAPI.GetOutputRecord(DC_READER_BC, tmp)


                 Case 32:
                    Reader.ReaderEngineAPI.GetDecodeRfidUID(uuidTmp)
                    Reader.ReaderEngineAPI.GetDecodeRfidData(databuf)
                    Reader.ReaderEngineAPI.GetOutputRecord(DC_READER_RFID, tmp)


               Case Else

                  Exit Select

           End Select

       End If
             MyBase.WndProc(msg)
End Sub
```

Note: ReadBarcodeData(), ReadRfidData() and ReadImage() are provided for users who want to scan and retrieve data without intercepting WM_DECODEDATA and WM_DECODEIMAGE. For example, if users want to scan data by clicking a GUI button, all they need to do is to invoke one of the three functions in the clicking event handler function.

Chapter 2

READER API

Find the necessary files on the CD-ROM before you start to write your application.

DLL required:
Reader_Ce_Net.dll and ReaderDll_CE.dll

IN THIS CHAPTER

2.1 INITIALIZE/IDENTIFY READER

2.1.1 INITIALIZATION

InitReader

Purpose	To initialize the reader (or readers, depending on hardware configuration).
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 1. Otherwise, it returns 0. ▸ 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
-133	E_READER_NO_SPEC_DC (Unknown error)

Remarks

This function must be called before any other functions. The barcode/RFID reader or readers will then be ready for use.

- If it fails to access the reader settings, it will automatically reset the reader.
- For error code "-102", check device configuration code for both barcode and RFID readers. It may be caused by incorrect configuration and both digits are non-zero.
- For error code "-103", check device configuration code for both barcode and RFID readers. It may be caused by incorrect configuration and both digits are zero.
- For error code "-111", check device configuration code – the 1st digit for barcode reader may be incorrect but the 2nd digit for RFID reader is correct!
- For error code "-112", check device configuration code – the 1st digit for barcode reader may be correct but the 2nd digit for RFID reader is incorrect!

See Also: GetActiveDevice, GetReaderType, SetActiveDevice

2.1.2 ACTIVE DEVICE

GetActiveDevice

Purpose	To find out the reader type currently active in use.
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 the barcode reader is active. (= RFID reader is inactive!)
32	DC_READER_RFID	Only the RFID reader is active. (= Barcode reader is inactive!)
255	ALL_DEVICE	All the readers are active.

Otherwise, it returns 0.

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

0	NO_ACTIVE_DEVICE
-101	E_READER_NOT_INIT

Remarks — The active reader type allows three possibilities:

- Only one barcode reader (1D or 2D) is active.
- Only the RFID reader is active.
- One barcode reader + RFID reader.

See Also — GetReaderType, InitReader, SetActiveDevice

SetActiveDevice

Purpose	To set the reader type ready for use.
Syntax	**int SetActiveDevice (int** *target***);**
Parameters	*target* [in] A value that specifies the active device.

0	NO_ACTIVE_DEVICE	Disable all active devices.
7	DC_READER_BC	Only the barcode reader is active. (= RFID reader is inactive!)
32	DC_READER_RFID	Only the RFID reader is active. (= Barcode reader is inactive!)
255	ALL_DEVICE	All the readers are active.

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 1.

Otherwise, it returns 0.

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

-101	E_READER_NOT_INIT

Remarks

The triggering behavior of a trigger key only applies to the active device. No event occurs with the inactive device.

An error does not occur if the specified reader type is not found!

See Also

GetReaderType, GetActiveDevice, InitReader

2.1.3 READER TYPE

GetReaderType

Purpose	To find out the reader type in hardware.
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:

1	ID_MOD_1D
64	ID_MOD_2D_4507
256	ID_MOD_MP_RFID
257	ID_MOD_1D + ID_MOD_MP_RFID
320	ID_MOD_2D_4507 + ID_MOD_MP_RFID

Otherwise, it returns 0.

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

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

Remarks

The reader type in hardware allows three possibilities:

- Only one barcode reader (1D or 2D) is present.
- Only the RFID reader is present.
- One barcode reader + RFID reader.

See Also: InitReader, GetActiveDevice, SetActiveDevice

2.2 OBTAIN DATA

2.2.1 DATA OUTPUT SETTINGS

Processed Data

Use **DataOutputSettings()** to configure how to process data for a barcode.

- When keyboard emulation is enabled, the barcode data will be processed to include [Code Type], [Code Length], [Prefix Code], and [Suffix Code].
- When keyboard emulation is disabled, you must use **GetOutputRecord()** to obtain processed data upon receiving the *WM_DECODEDATA* message. Refer to 1.3 Register Decode Message.

Barcode Data	Length (Byte)	Description
Code Type	1	Barcode Type. Refer to the parameter *showCodeType* of DataOutputSettings().
Code Length	4	Length of decoded barcode, prefix/suffix code is included if there is any. Refer to the parameter *showCodeLen* of DataOutputSettings().
Prefix Code	0~10	No prefix code if the value is 0. Refer to the parameter *prefixCode* of DataOutputSettings().
Decode Data	1~4096	Decoded barcode data
Suffix Code	0~10	No suffix code if the value is 0. Refer to the parameter *suffixCode* of DataOutputSettings().

Note: The fields of data can be as follows –
[Code Type][Code Length][Prefix Code][Decode Data][Suffix Code]

Use **DataOutputSettings()** and **RfidSettings()** to configure how to process data for an RFID tag.

- When keyboard emulation is enabled, the RFID tag data will be processed to include [Tag Type], [Tag Length], [Prefix Code], [Suffix Code], and [Delimiter].
- When keyboard emulation is disabled, you must use **GetOutputRecord()** to obtain processed data upon receiving the *WM_DECODEDATA* message. Refer to 1.3 Register Decode Message.

RFID Tag Data	Length (Byte)	Description
Tag Type	1	RFID Tag Type. Refer to the parameter *showCodeType* of DataOutputSettings().
Tag Length	4	Length of decoded tag, prefix/suffix code is included if there is any. Refer to the parameter *showCodeLen* of DataOutputSettings(). It may refer to - UID only - Data only - UID + Data
Prefix Code	0~10	No prefix code if the value is 0. Refer to the parameter *prefixCode* of DataOutputSettings().

Tag UID	1~20	Universal Identification (UID) of the tag. Refer to the parameter *readUid* of RfidSettings().
Suffix Code	0~10	No suffix code if the value is 0. Refer to the parameter *suffixCode* of DataOutputSettings().
Delimiter	1	The character used to separate UID and data. Refer to the parameter *useDelim* of RfidSettings(). No delimiter if the value is 0x00.
Tag Data	1~4096	Decoded RFID data. Refer to the parameter *readUserData* of RfidSettings().

Note: The fields of data can be as follows –
[Tag Type][Tag Length][Prefix Code][Tag UID][Suffix Code][Delimiter][Tag Data]
If Tag UID is disabled, the output will not include prefix code, UID, suffix code, and the delimiter.

Raw Data

Use the following functions to obtain raw data —

- **GetDecodeData()** and **GetDecodeType()** for barcode data.
- **GetImageContent()** and **GetImageSize()**for 2D image.
- **GetDecodeRfidData()**, **GetDecodeRfidUID()** and **GetDecodeTagType()** for RFID data.

DataOutputSettings

Purpose

To configure data output settings. The decoded data may be processed with 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

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 output settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set output settings

enableKeyboardEmulation

[in][out] A value that specifies whether data is emulated as typed text and sent to the active Window.

0*	Disable keyboard emulation ▸ All the rest parameters will not take effect unless GetOutputRecord() is called.
1	Emulate keyboard input on local machine
2	Emulate 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 automatically insert a character after decoding.

0	Disable
1*	Join to the end of data (= decoded data + Enter-character)
2	Join to the beginning of data (= Enter-character + decoded data)

autoEnterChar

[in][out] A value that specifies a character for Auto Enter.

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 insert barcode type or RFID tag type in data records…

0*	DO NOT transmit code type
1	Transmit code type

showCodeLen

[in][out] A value that specifies whether to insert length code for barcode or RFID tag in data records…

- If it is set to read only UID of a RFID tag, the code length will refer to the length of UID

0*	DO NOT transmit code length
1	Transmit code length

prefixCode

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

- For RFID decoding, Tag UID must be enabled so that the output will include prefix code, UID, and suffix code.

suffixCode

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

- For RFID decoding, Tag UID must be enabled so that the output will include prefix code, UID, and suffix code.

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

Remarks

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

When data comes from the barcode reader, the fields of data can be as follows:

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

[Code Type]:	This field is output only when the value of *showCodeType* is non-zero.
[Code Length]:	This field is output only when the value of *showCodeLen* is non-zero. It covers the prefix/suffix code if any has been specified.
[Prefix Code]:	This field is output only when the value of *prefixCode* is non-zero.
[Decode Data]:	This field is output when the data has been decoded.
[Suffix Code]:	This field is output only when the value of *suffiixCode* is non-zero.

When data comes from the RFID reader, the fields of data can be as follows:

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

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

[Tag Type]:	This field is output only when the value of *showCodeType* is non-zero.
[Tag Length]:	This field is output only when the value of *sShowCodeLen* is non-zero. When specified to read UID only, it refers to the length of UID. It covers the prefix/suffix code if any has been specified.
[Prefix Code]:	This field is output only when the value of *prefixCode* is non-zero.
[Tag UID]:	This field is output only when the value of *readUID* of RfidSettings() is non-zero.
[Suffix Code]:	This field is output only when the value of *suffiixCode* is non-zero.
[Delimiter]:	This field is output only when the value of *useDelim* of RfidSettings() is non-zero.
[Tag Data]:	This field is output only when the value of *readData* of RfidSettings() is non-zero.

See Also

GetDecodeData, GetDecodeType,

GetDecodeRfidData, GetDecodeRfidUID, GetDecodeTagType,

GetOutputRecord, RfidSettings

GetOutputRecord

Purpose

To obtain the data record, of which the decoded data has been processed with code/tag type, data length, prefix code, suffix code, etc.

Syntax

int GetOutputRecord (int *target***,**

ref string *buf***);**

Parameters

target

[in] A value that specifies the reader type.

7	DC_READER_BC	Read by the barcode reader only.
32	DC_READER_RFID	Read by the RFID reader only.

buf

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

Example for C#

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

Example for VB

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

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-101	E_READER_NOT_INIT
-300	E_BUFF_IS_TOO_SMALL

Remarks

Before another successful decoding, this function is called to obtain the current data record. Depending on the reader type and associated reader settings, the fields of output record may differ.

When data comes from the barcode reader, the fields of data can be as follows:

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

[Code Type]:	This field is output only when the value of *showCodeType* of DataOutputSettings() is non-zero.
[Code Length]:	This field is output only when the value of *showCodeLen* of DataOutputSettings() is non-zero. It covers the prefix/suffix code if any has been specified.
[Prefix Code]:	This field is output only when the value of *prefixCode* of DataOutputSettings() is non-zero.
[Decode Data]:	This field is output when the data has been decoded.
[Suffix Code]:	This field is output only when the value of *suffiixCode* of DataOutputSettings() is non-zero.

When data comes from the RFID reader, the fields of data can be as follows:

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

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

[Tag Type]:	This field is output only when the value of *showCodeType* of DataOutputSettings() is non-zero.
[Tag Length]:	This field is output only when the value of *sShowCodeLen* of DataOutputSettings() is non-zero. When specified to read UID only, it refers to the length of UID. It covers the prefix/suffix code if any has been specified.
[Prefix Code]:	This field is output only when the value of *prefixCode* of DataOutputSettings() is non-zero.
[Tag UID]:	This field is output only when the value of *readUID* of RfidSettings() is non-zero.
[Suffix Code]:	This field is output only when the value of *suffiixCode* of DataOutputSettings() is non-zero.
[Delimiter]:	This field is output only when the value of *useDelim* of RfidSettings() is non-zero.
[Tag Data]:	This field is output only when the value of *readData* of RfidSettings() is non-zero.

See Also

GetDecodeData, GetDecodeType,

GetDecodeRfidData, GetDecodeRfidUID, GetDecodeTagType,

DataOutputSettings, RfidSettings

2.2.2 RFID SETTINGS

RfidSettings

Purpose

To configure data output settings. The decoded data may be processed with code/tag type, data length, prefix code, suffix code, etc.

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

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 RFID settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set RFID settings

readUID

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

0	DO NOT decode Tag UID
1*	Decode Tag UID

readData

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

0*	DO NOT decode Tag data
1	Decode Tag data

userDataStartByte

[in][out] A value that specifies the start position for decoding Tag data. As the capacity may vary by different tags, when you pass a value greater than allowed, it will be automatically adjusted to the maximum value allowed on a tag.

-1*	A value that specifies the start byte or start page (for a specific tag type). "-1" means that it starts from the default page (see below) for all tags:

4	Mifare	(ISO 14443A)
4	SR176	(ISO 14443B)
3	ICODE SLI	(ISO 15693)

0	LRI512	(ISO 15693)
3	SRF55VxxP	(ISO 15693)
0	EM4135	(ISO 15693)
0	Tag-it	(ISO 15693)
0	Others	(ISO 15693)
5	ICODE	(Phillips)

readUserDataLen

[in][out] A value that specifies the maximum length for decoding Tag data. It is set to 128 bytes by default. As the capacity may vary by different tags, when you pass a value greater than allowed, it will be automatically adjusted to the maximum value allowed on a tag.

userDelim

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

0*	DO NOT add delimiter
1~127	Add 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.

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

LoginKey Example	*Key String*
0	(Keep the current string)
"F1F2F3F4F5F6"	F1F2F3F4F5F6

loginKeyType

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

0*	Key A
1	Key B

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

Remarks

When data comes from the RFID reader, the fields of data can be as follows:

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

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

[Tag Type]:	This field is output only when the value of *showCodeType* of DataOutputSettings() is non-zero.
[Tag Length]:	This field is output only when the value of *sShowCodeLen* of DataOutputSettings() is non-zero. When specified to read UID only, it refers to the length of UID. It covers the prefix/suffix code if any has been specified.
[Prefix Code]:	This field is output only when the value of *prefixCode* of DataOutputSettings() is non-zero.
[Tag UID]:	This field is output only when the value of *readUID* of RfidSettings() is non-zero.
[Suffix Code]:	This field is output only when the value of *suffiixCode* of DataOutputSettings() is non-zero.
[Delimiter]:	This field is output only when the value of *useDelim* of RfidSettings() is non-zero.
[Tag Data]:	This field is output only when the value of *readData* of RfidSettings() is non-zero.

See Also

ChangeMifareKey, GetDecodeRfidData, GetDecodeRfidUID GetDecodeTagType, DataOuputSettings, GetOutputRecord

Note: (1) A scanning session starts when the trigger key is pressed.
(2) The specified value for 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	To modify the security key to a Mifare card — 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 target sector that you want to modify. *loginKey* [in] Pass Key A of the target sector. *newKeyA* [in] New Key A of the target sector that will be used to login. If not, pass the old Key A instead. This parameter must be a character array of 6 bytes length. *newKeyB* [in] New Key B of the target sector that will be used to login. If not, pass the 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 1.

Otherwise, it returns 0.

Remarks

Due to security concerns, it does not allow reading the security key. Please make sure the key values are correct.

See Also

RfidSettings

2.2.3 BARCODE DATA

Use **GetDecodeData()** and **GetDecodeType()** to obtain raw data of a barcode.

GetDecodeData	
Purpose	To get the decoded data of a barcode upon pressing the trigger.
Syntax	**int GetDecodeData (ref string** *buf***);**
Parameters	*buf* [out] Pointer to a buffer where the decoded data is stored.

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

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

Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered: -101 E_READER_NOT_INIT -300 E_BUFF_IS_TOO_SMALL
Remarks	Before another successful decoding, this function is called to obtain the current decoded data. If Prefix Code or Suffix Code is specified, it will be inserted into the decoded data accordingly.
See Also	DataOuputSettings, GetDecodeType, GetOutputRecord

GetDecodeType

Purpose	To obtain the code type of the decoded barcode upon pressing the 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 0. ▸ 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 current 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	To read data from the barcode without intercepting WM_DECODEDATA.
Syntax	**int ReadBarcodeData (ref int** *codeType*, **ref string** *outBuf*, **int** *timeout*);
Parameters	*codeType* [out] Pointer to a variable that stores the barcode type. *outBuf* [out] Pointer to a buffer where the data is stored. *timeout* Reserved [in] Pass 0 to ignore this parameter. It is decided by UserPreferences_1D() or UserPreferences_2D_4507().

Example for C#	`int codeType = 0;`
	`string outBuf = string.Empty;`
	`b1 = Reader.ReaderEngineAPI.ReadBarcodeData(`
	`ref codeType, ref outBuf, 3);`

Example for VB	`Dim b1 As Integer`
	`Dim codeType As Integer`
	`Dim outBuf As String = ""`
	`b1 = Reader.ReaderEngineAPI.ReadBarcodeData(codeType, outBuf, 3)`

Return Value	If successful, it returns the number of bytes obtained (not including the terminating null character). Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered: 0 — E_READER_UNKNOWN_ERROR -101 — E_READER_NOT_INIT
Remarks	This function is provided for users who want to scan and retrieve data without intercepting WM_DECODEDATA. For example, if users want to scan data by clicking 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()** to obtain raw data of an RFID tag. Some RFID tags support both read/write operations, on a page-by-page basis. You may find it necessary to define your own read/write operation. If so, use **ReadRfidData()** and **WriteRfidData()** to read and write to the tag. Due to the length of page for each tag is different, these functions will automatically truncate data into pages. For reference only, the table below lists the start page for read/write operation on a number of RFID tags.

Start Page	Tag Type	
4	Mifare	(ISO 14443A)
4	SR176	(ISO 14443B)
3	ICODE SLI	(ISO 15693)
0	LRI512	(ISO 15693)
3	SRF55VxxP	(ISO 15693)
0	EM4135	(ISO 15693)
0	Tag-it	(ISO 15693)
0	Others	(ISO 15693)
5	ICODE	(Phillips)

Note: The start page and the length of page may vary by tag type. Please refer to the specifications of your RFID tags for memory organization.

GetDecodeRfidData

Purpose	To get the decoded data of an RFID tag upon pressing the trigger.
Syntax	**int GetDecodeRfidData (ref string** *buf***);**
Parameters	*buf* [out] Pointer to a buffer where the 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 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

-101	E_READER_NOT_INIT
-300	E_BUFF_IS_TOO_SMALL

Remarks	Before another successful decoding, this function is called to obtain the current decoded data. If Prefix Code or Suffix Code is specified, it will be inserted into the decoded data accordingly.
See Also	GetDecodeRfidUID, GetDecodeTagType, DataOutputSettings, GetOutputRecord, RfidSettings

GetDecodeRfidUID

Purpose	To obtain the UID of the decoded RFID tag upon pressing the trigger.
Syntax	**int GetDecodeRfidUID (ref string** *buf***);**
Parameters	*buf* [out] Pointer to a buffer where the 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 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

-101	E_READER_NOT_INIT
-300	E_BUFF_IS_TOO_SMALL

Remarks	Before another successful decoding, this function is called to obtain the current decoded UID. If Prefix Code or Suffix Code is specified, it will be inserted into the decoded data accordingly.
See Also	GetDecodeRfidData, GetDecodeTagType, DataOutputSettings, GetOutputRecord, RfidSettings

GetDecodeTagType

Purpose	To obtain the tag type of the decoded RFID tag upon pressing the 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.

Tag Type		*RFID Tag / Standard*
0x49	73 (I)	Icode
0x4D	77 (M)	Mifare
0x53	83 (S)	SR176
0x54	84 (T)	Tagit
0x56	86 (V)	ISO 15693
0x5A	90 (Z)	ISO 14443B

Otherwise, it returns 0.

- 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 tag type of the current decoded RFID tag.
See Also	GetDecodeRfidData, GetDecodeRfidUID, DataOutputSettings, GetOutputRecord, RfidSettings

ReadRfidData

Purpose: To read data from the RFID tag 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 part of RFID data is to be read.

1	READ_UID	Read UID only.
2	READ_DATA	Read data only.

szReadData

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

nStartByte

[in] A value that specifies the start byte or start page (for a specific tag type).

-1	Start from the default page (see below) for all tags		
	4	Mifare	(ISO 14443A)
	4	SR176	(ISO 14443B)
	3	ICODE SLI	(ISO 15693)
	0	LRI512	(ISO 15693)
	3	SRF55VxxP	(ISO 15693)
	0	EM4135	(ISO 15693)
	0	Tag-it	(ISO 15693)
	0	Others	(ISO 15693)
	5	ICODE	(Phillips)

nReadLen

[in] Actual number of bytes to read. As the number of pages that allow for read operation may vary by different tags, when you pass a value greater than allowed, it will be automatically adjusted to the maximum value allowed on a tag.

nTimeout Reserved

loginKey

[in] A string variable that stores the login key. The security key for accessing a specific 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 will be 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 specifies the login key type.

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 1.

Otherwise, it returns 0.

- 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 is provided for users who want to scan and retrieve data without intercepting WM_DECODEDATA. For example, if users want to scan data by clicking a GUI button, all they need to do is to invoke this function in the clicking event handler function. The specified RFID data is collected after decoding.

See Also

DataOutputSettings, GetOutputRecord, RfidSettings, WriteRfidData

WriteRfidData

Purpose: To write data to the RFID tag upon pressing the trigger.

Syntax:

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

Parameters

szWriteData

[in] A string variable that stores the data.

nStartByte

[in] A value that specifies the start byte or start page (for a specific tag type).

-1	Start from the default page (see below) for all tags

4	Mifare	(ISO 14443A)
4	SR176	(ISO 14443B)
3	ICODE SLI	(ISO 15693)
0	LRI512	(ISO 15693)
3	SRF55VxxP	(ISO 15693)
0	EM4135	(ISO 15693)
0	Tag-it	(ISO 15693)
0	Others	(ISO 15693)
5	ICODE	(Phillips)

nWriteLen

[in] Maximum number of bytes to write. As the number of pages that allow for write operation may vary by different tags, when you pass a value greater than allowed, it will be automatically adjusted to the maximum value allowed on a tag.

nWritten

[in] Pointer to a buffer where the actual number of bytes (of data written to tag) is stored.

nTimeout Reserved

nMode

[in] A value that specifies the operation mode.

0	Start to write immediately.
Non-zero	Press the trigger to start the write operation.

loginKey

[in] A string variable that stores the login key. The security key for accessing a specific 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 will be 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 specifies the login key type.

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 1.

Otherwise, it returns 0.

- 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 is provided for users to define their own write operation on the RFID reader. The specified RFID data is written to the tag.

See Also

DataOutputSettings, GetOutputRecord, ReadRfidData, RfidSettings

2.3 OBTAIN IMAGE

2.3.1 IMAGE OUTPUT SETTINGS

ImageOptions_2D_4507

Purpose: To configure 2D scan engine for image output.

Syntax:

```
int ImageOptions_2D_4507 (int rw,
                          ref int enable,
                          ref int enableWaitCursor,
                          ref int decodingAutoexposure,
                          ref int decodingIllumination,
                          ref int decodeAimingPattern,
                          ref int ledIllumination,
                          ref int imageAutoexposure,
                          ref int imageIllumination,
                          ref int gain,
                          ref int exposureTime,
                          ref int snapshotModeTimeout,
                          ref int snapshotAimingPattern,
                          ref int imageResolution,
                          ref int jpgImageOptions,
                          ref int qualityValue,
                          ref int jpgSizeValue,
                          ref int imageFileFormat,
                          ref int bitsPerPixel);
```

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 2D image settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set 2D image settings

enable

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

0*	Disable Image Capture
1	Enable Image Capture (= Snapshot Mode)

enableWaitCursor

[in][out] A value that specifies whether to enable the wait cursor while processing an image.

0	Disable Wait Cursor
1*	Enable Wait Cursor

decodingAutoexposure

[in][out] A value that specifies whether to allow 2D scan engine to control gain settings and exposure time to best capture a 2D barcode.

0	Disable Decoding Autoexposure
1*	Enable Decoding Autoexposure

- When Decoding Autoexposure is disabled, specify the gain and exposure time. This parameter is only recommended for advanced users with difficult image capture situations.

decodingIllumination

[in][out] A value that specifies whether to flash illumination on every barcode capture to aid decoding.

0	Disable Decoding Illumination
1*	Enable Decoding Illumination

decodeAimingPattern

[in][out] A value that specifies whether to project the aiming pattern on every barcode capture.

0	Disable Decode Aiming Pattern
2*	Enable Decode Aiming Pattern

ledIllumination

[in][out] Always 0 (Internal LED illumination)

imageAutoexposure

[in][out] A value that specifies whether to allow 2D scan engine to control gain settings and exposure time to best capture an image.

0	Disable Image Capture Autoexposure
1*	Enable Image Capture Autoexposure

- When Image Autoexposure is disabled, specify the gain and exposure time. This parameter is only recommended for advanced users with difficult image capture situations.

imageIllumination

[in][out] A value that specifies whether to flash illumination on every image capture to aid decoding.

0	Disable Image Capture Illumination
1*	Enable Image Capture Illumination

gain Reserved

exposureTime

[in][out] A value that specifies exposure time when Decoding Autoexposure or Image Autoexposure is disabled.

1~1000	10* (in units of 100 µs)

- Exposure Time controls the amount of time the CCD is allowed to collect light, mush like the shutter speed for a camera, Generally, the brighter the environment, the lower the exposure time. Increasing the exposure time past 20 ms in a handheld application increases the risk of blurring the image due to hand jitter.

snapshotModeTimeout

[in][out] A value that specifies the amount of time it remains in Snapshot Mode. It exits Snapshot Mode upon a trigger event, or when the Snapshot Mode Timeout elapses.

0~9	0* (= 30 seconds) - Values increment by 30. For example, 1 = 60 seconds, 2 = 90 seconds, etc.

snapshotAimingPattern

[in][out] A value that specifies whether to project the aiming pattern on every image capture.

0	Disable Snapshot Aiming Pattern
1*	Enable Snapshot Aiming Pattern

imageResolution

[in][out] A value that specifies how to alter image resolution before compression. Multiple pixels are combined to one pixel, resulting in a smaller image containing the original content with reduced resolution.

0*	752×480 (= Full resolution)
1	376×240 (= Half resolution)
3	188×120 (= 1/4 resolution)

jpgImageOptions

[in][out] A value that specifies whether to optimize JPEG images for either size or quality.

0	Optimized for JPEG size
1*	Optimized for JPEG quality

qualityValue

[in][out] A value that specifies the image quality. Set a value from 5 to 100, where "100" represents the highest quality image.

5~100	65*

jpgSizeValue

[in][out] A value that specifies the image size. Set a value from 5 to 150, which represents the file size in multiples of 1024 bytes (1K). For example, setting this value to 8 permits the file size to be as large as 8192 bytes.

5~150	40*

imageFileFormat

[in][out] A value that specifies the image format, in which captured images are stored.

1*	JPEG file format
3	BMP file format

bitsPerPixel

[in][out] A value that specifies the number of significant bits per pixel (BPP) to use when capturing an image. For JPEG files, these BPP settings are ignored for it always uses 8 bits per pixel!

0	1 bit per pixel (for black and white images)
1	4 BPP (to assign 1 of 16 levels of grey to each pixel)
2*	8 BPP (to assign 1 of 256 levels of grey to each pixel)

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

See Also

GetImageContent, GetImageSize, ReadImage

2.3.2 IMAGE DATA

Use **GetImageContent()** and **GetImageSize()** to obtain raw data of a 2D image. These functions are only available on a device with 2D scan engine when the image capture feature is enabled.

GetImageContent	
Purpose	To obtain the contents of a captured image upon pressing the trigger.
Syntax	**int GetImageContent (ref byte[]** *buf*, **int** *bufSize*);
Parameters	*buf* [out] A byte array that stores the image contents. *bufSize* [in] The size of the array, in characters.

Example for C#	`int b1 = 0;`
	`byte[] buftmp = new byte[100];`
	`b1 = Reader.ReaderEngineAPI.GetImageContent(ref buftmp, 100);`

Example for VB	`Dim b1 As Integer`
	`Dim buftem(100) As Byte`
	`b1 = Reader.ReaderEngineAPI.GetImageContent(buftmp, 100)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.
Remarks	When receiving a WM_DECODEIMAGE message, you may call this function to get the contents of a captured image, in bytes.
See Also	GetImageSize, ImageOptions_2D_4507

GetImageSize	
Purpose	To obtain the size of a captured image upon pressing the trigger.
Syntax	**int GetImageSize ();**
Example for C#	`int size = 0;` `size = Reader.ReaderEngineAPI.GetImageSize();`
Example for VB	`Dim size As Integer` `size = Reader.ReaderEngineAPI.GetImageSize()`
Return Value	If successful, it returns the image size (in bytes). Otherwise, it returns 0.
Remarks	When receiving a WM_DECODEIMAGE message, you may call this function to get the image size, in bytes.
See Also	GetImageContent, ImageOptions_2D_4507

ReadImage	
Purpose	To capture an image without intercepting WM_DECODEIMAGE.
Syntax	**int ReadImage (ref ArrayList** *buf*, **ref int** *capturedImgSize*);
Parameters	*pbuf* [out] An ArrayList variable that stores the image contents. *capturedImgSize* [out] An integer variable that stores the size of a captured image.

Example for C#	`int b1 = 0;`
	`ArrayList buftmp = new ArrayList();`
	`int capturedImgSize = 0;`
	`b1 = Reader.ReaderEngineAPI.ReadImage(ref buftmp, ref capturedImgSize);`

Example for VB	`Dim b1 As Integer`
	`Dim buftmp As New ArrayList()`
	`Dim capturedImgSize As Integer`
	`b1 = Reader.ReaderEngineAPI.ReadImage(buftmp, capturedImgSize)`

Return Value	If successful, it returns 1. Otherwise, it returns 0.
Remarks	This function is provided for users who want to scan and retrieve data without intercepting WM_DECODEIMAGE. For example, if users want to scan data by clicking a GUI button, all they need to do is to invoke this function in the clicking event handler function. The 2D scan engine must be initialized successfully before calling this function.
See Also	ImageOptions_2D_4507

2.4 MANIPULATE STATUS INDICATION

According to the settings of **NotificationSettings()**, the ReaderDLL will automatically signal whether a successful decoding is obtained by playing a sound or turning on/off the vibrator. You may call **Beeper()** to signal for receiving other events except decoding.

2.4.1 NOTIFICATION SETTINGS

NotificationSettings

Purpose: To configure notification settings.

Syntax:

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

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

goodRead

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

- If buzzer is enabled at the same time, time delay occurs when playing sound via speaker.

0	Mute
1~9	Sound 2*

enableVibrator

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

0*	Disable Vibrator
1	Enable Vibrator

vibrationTime

[in][out] A value that specifies how long it vibrates.

0	No vibration
1~9	2* (second)

ledDuration

[in][out] A value that specifies whether to enable Good Read LED (green).

0*	Disable
Non-zero	Good Read LED duration (millisecond)

buzzerDuration

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

0*	Disable
1~255	(0.1 second)

buzzerFreq

[in][out] A value that specifies the buzzer frequency (+ 500 Hz per increment) when it beeps to indicate a successful reading.

1*	500 Hz
2~10	1000 Hz ~5000 Hz

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

See Also

Beeper

2.4.2 BEEPER (SPEAKER)

Beeper

Purpose: To play a sound via speaker.

Syntax:

```
int Beeper (int mode,
            string path);
```

Parameters:

mode

[in] A value that specifies the sound to be played.

0	Mute
1~9	Sound 1 ~ Sound 9
-1	User-defined path

path

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

- This only applies 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 1.

Otherwise, it returns 0.

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

-241	E_SOUND_INVALID_INDEX
-242	E_SOUND_INVALID_PATH
-243	E_SOUND_PLAY_FAILED

Remarks: This function plays a sound or WAVE file in asynchronous mode. To stop playing, use Beeper(0, “”).

See Also: NotificationSettings

2.5 OBTAIN ESSENTIAL INFORMATION

2.5.1 READER DLL VERSION

GetDllVer	
Purpose	To obtain the current C/C++ library version for Reader DLL.
Syntax	**int GetDllVer (ref string** *buf***);**
Parameters	*buf* [out] Pointer to a buffer where the 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 1. Otherwise, it returns 0.

GetNETDLLVersion	
Purpose	To obtain the current .NET library version for Reader DLL.
Syntax	**int GetNETDLLVersion (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 = 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 1. Otherwise, it returns 0.

2.5.2 DECODER VERSION

GetDecoderVersion

Purpose: To obtain the decoder version.

Syntax: **int GetDecoderVersion (int** *target*,
ref string *buf*);

Parameters: *target*

[in] A value that specifies the reader type.

7	DC_READER_BC	Read decoder version for the barcode reader.
32	DC_READER_RFID	Read decoder version for the RFID reader.

buf

[out] Pointer to a buffer where the decoded data 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 (not including the terminating null character).

Otherwise, it returns 0.

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

0	E_READER_UNKNOWN_ERROR
-101	E_READER_NOT_INIT

2.5.3 ERROR INFORMATION

GetErrorCode	
Purpose	To find the current error condition encountered.
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.

2.6 CONFIGURE SCAN ENGINE – 1D: CCD/LASER SCAN ENGINE

UserPreferences_1D() is provided for configuring reader settings. The rest functions are provided for changing symbology settings.

- CCD: CCD scan engine
- Laser: Laser scan engine

2.6.1 PREFERENCES

UserPreferences_1D	**CCD, Laser**

Purpose: To configure reader settings.

Syntax:

int UserPreferences_1D (int *rw*,
ref int *scanMode*,
ref int *readRedundancy*,
ref int *timeout*);

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

scanMode

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

0	Auto Off mode
1	Continuous mode
2	N/A
3	Alternate mode
4	N/A
5	N/A
6*	Laser mode
7	Test mode
8	N/A

readRedundancy

[in][out] A value that specifies the number of times of supplementary decoding of the same barcode that makes a valid reading.

0*	No redundancy
1	One more decoding is required
2	Two more decoding is required
3	Three more decoding is required

timeout

[in][out] A value that specifies the elapsed time before a scanning session ends. This setting only applies to Laser mode and Auto Off mode.

0~255	3* (second)

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.2 CODABAR_1D

CodaBar_1D	**CCD, Laser**

Purpose: To configure symbology settings — Codabar.

Syntax:

int CodaBar_1D (int *rw*,
ref int *enable*,
ref int *startStopChar*,
ref int *transmitStartStopChar*);

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

enable

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

0	Disable
1*	Enable

startStopChar

[in][out] A value that specifies the start/stop characters.

0*	abcd/abcd
1	abcd/tn*e
2	ABCD/ABCD
3	ABCD/TN*E

transmitStartStopChar

[in][out] A value that specifies whether to include the start/stop characters in the data being transmitted.

0*	Do not transmit
1	Transmit

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.3 INDUSTRIAL25_1D

Industrial25_1D	**CCD, Laser**

Purpose: To configure symbology settings — Industrial 25.

Syntax:

```
int Industrial25_1D (int rw,
                     ref int enable,
                     ref int checkDigitVerification,
                     ref int transmitCheckDigit,
                     ref int startStopPattern,
                     ref int isMaxMinLengthFormat,
                     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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

checkDigitVerification

[in][out] A value that specifies whether to perform check digit verification when decoding barcodes.

0*	Do not verify
1	Verify

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit in the data being transmitted.

0	Do not transmit
1*	Transmit

startStopPattern

[in][out] A value that specifies which start/stop pattern in use.

0*	Use Industrial 25 Start/Stop pattern
1	Use Interleaved 25 Start/Stop pattern
2	Use Matrix 25 Start/Stop pattern

isMaxMinLengthFormat

[in][out] A value that specifies whether to use the Max/Mix length format.

0	Fixed length format
1*	Max/Min length format

length1

[in][out] A value that specifies the maximum length or the 1st fixed length.

0~127	127*

length2

[in][out] A value that specifies the minimum length or the 2nd fixed length.

0~127	4*

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.4 INTERLEAVED25_1D

Interleaved25_1D	**CCD, Laser**

Purpose: To configure symbology settings — Interleaved 25.

Syntax:

```
int Interleaved25_1D (int rw,
                      ref int enable,
                      ref int checkDigitVerification,
                      ref int transmitCheckDigit,
                      ref int startStopPattern,
                      ref int isMaxMinLengthFormat,
                      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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

checkDigitVerification

[in][out] A value that specifies whether to perform check digit verification when decoding barcodes.

0*	Do not verify
1	Verify

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit in the data being transmitted.

0	Do not transmit
1*	Transmit

startStopPattern

[in][out] A value that specifies which start/stop pattern in use.

0	Use Industrial 25 Start/Stop pattern
1*	Use Interleaved 25 Start/Stop pattern
2	Use Matrix 25 Start/Stop pattern

isMaxMinLengthFormat

[in][out] A value that specifies whether to use the Max/Mix length format.

0	Fixed length format
1*	Max/Min length format

length1

[in][out] A value that specifies the maximum length or the 1st fixed length.

0~127	127*

length2

[in][out] A value that specifies the minimum length or the 2nd fixed length.

0~127	4*

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.5 MATRIX25_1D

Matrix25_1D	**CCD, Laser**

Purpose: To configure symbology settings — Matrix 25.

Syntax:

```
int Matrix25_1D (int rw,
                 ref int enable,
                 ref int checkDigitVerification,
                 ref int transmitCheckDigit,
                 ref int startStopPattern,
                 ref int isMaxMinLengthFormat,
                 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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

checkDigitVerification

[in][out] A value that specifies whether to perform check digit verification when decoding barcodes.

0*	Do not verify
1	Verify

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit in the data being transmitted.

0	Do not transmit
1*	Transmit

startStopPattern

[in][out] A value that specifies which start/stop pattern in use.

0	Use Industrial 25 Start/Stop pattern
1	Use Interleaved 25 Start/Stop pattern
2*	Use Matrix 25 Start/Stop pattern

isMaxMinLengthFormat

[in][out] A value that specifies whether to use the Max/Mix length format.

0	Fixed length format
1*	Max/Min length format

length1

[in][out] A value that specifies the maximum length or the 1st fixed length.

0~127	127*

length2

[in][out] A value that specifies the minimum length or the 2nd fixed length.

0~127	4*

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.6 CODE39_1D

Code39_1D	**CCD, Laser**

Purpose: To configure symbology settings — Code 39.

Syntax:

```
int Code39_1D (int rw,
               ref int enable,
               ref int checkDigitVerification,
               ref int transmitCheckDigit,
               ref int fullASCII,
               ref int transmitStartStopChar);
```

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

enable

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

0	Disable
1*	Enable

checkDigitVerification

[in][out] A value that specifies whether to perform check digit verification when decoding barcodes.

0*	Do not verify
1	Verify

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit in the data being transmitted.

0	Do not transmit
1*	Transmit

fullASCII

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

0*	Standard Code 39
1	Code 39 Full ASCII

transmitStartStopChar

[in][out] A value that specifies whether to include the start/stop characters in the data being transmitted.

0*	Do not transmit
1	Transmit

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.7 CODE93_1D

Code93_1D	**CCD, Laser**

Purpose: To configure symbology settings — Code 93.

Syntax:

int Code93_1D (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.8 CODE128_1D

Code128_1D	**CCD, Laser**

Purpose: To configure symbology settings — Code 128.

Syntax:

int Code128_1D (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.9 GS1_128_1D

GS1_128_1D	**CCD, Laser**

Purpose: To configure symbology settings — GS1-128 (EAN-128).

Syntax:

```
int GS1_128_1D (int rw,
                ref int enable,
                ref int wantCodeID,
                ref int enableGS1_128FieldSeparator);
```

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

enable

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

0	Disable
1*	Enable

wantCodeID

[in][out] A value that specifies whether to strip GS1-128 Code ID "]C1".

0*	Do not transmit
1	Transmit

enableGS1_128FieldSeparator

[in][out] Pass 0 to ignore this parameter.

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.10 ISBT128_1D

Isbt128_1D	**CCD, Laser**

Purpose: To configure symbology settings — ISBT 128.

Syntax:

int Isbt128_1D (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.11 I_F_PHARMACODE_1D

I_F_Pharmacode_1D	**CCD, Laser**

Purpose — To configure symbology settings — Italian and French Pharmacode.

Syntax

```
int I_F_Pharmacode_1D (int rw,
                       ref int enableItalianPharmacode,
                       ref int enableFrenchPharmacode,
                       ref int transmitCheckDigitItalian,
                       ref int transmitCheckDigitFrench);
```

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

enableItalianPharmacode

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

0*	Disable
1	Enable

enableItalianPharmacode

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

0*	Disable
1	Enable

transmitCheckDigitItalian

[in][out] A value that specifies whether to include the check digit in the Italian Pharmacode data being transmitted.

0	Do not transmit
1*	Transmit

transmitCheckDigitFrench

[in][out] A value that specifies whether to include the check digit in the French Pharmacode data being transmitted.

0	Do not transmit
1*	Transmit

Return Value — If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.12 MSI_1D

Msi_1D	**CCD, Laser**

Purpose: To configure symbology settings — MSI.

Syntax:

```
int Msi_1D (int rw,
            ref int enable,
            ref int checkDigitVerification,
            ref int transmitCheckDigit,
            ref int isMaxMinLengthFormat,
            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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0*	Disable
1	Enable

checkDigitVerification

[in][out] A value that specifies whether to perform check digit verification when decoding barcodes.

0*	Single Modulo 10
1	Double Modulo 10
2	Modulo 11 and Modulo 10

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit(s) in the data being transmitted.

0	Last check digit is NOT transmitted
1*	Both check digits are transmitted
2	Both check digits are NOT transmitted

isMaxMinLengthFormat

[in][out] A value that specifies whether to use the Max/Mix length format.

0	Fixed length format
1*	Max/Min length format

length1

[in][out] A value that specifies the maximum length or the 1st fixed length.

0~127	127*

length2

[in][out] A value that specifies the minimum length or the 2nd fixed length.

0~127	4*

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.13 NEGATIVBARCODE_1D

NegativeBarcode_1D	**CCD, Laser**

Purpose: To configure negative barcode settings.

Syntax:

int NegativeBarcode_1D (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0*	Disable
1	Enable

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.14 PLESSEY_1D

Plessey_1D	**CCD, Laser**

Purpose: To configure symbology settings — Plessey.

Syntax:

int Plessey_1D (int *rw*,
ref int *enable*,
ref int *transmitCheckDigit*,
ref int *convertToUKPlessey*);

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

enable

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

0*	Disable
1	Enable

transmitCheckDigit

[in][out] A value that specifies whether to include the two check digits in the data being transmitted.

0	Do not transmit
1*	Transmit

convertToUKPlessey

[in][out] A value that specifies whether to use the Max/Mix length format.

0*	Do not convert
1	Convert

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.15 GS1_DATABAR_1D

GS1_DataBar_1D	**CCD, Laser**

Purpose: To configure symbology settings — GS1 DataBar (RSS family).

Syntax:

int GS1_DataBar_1D (int *rw*,
ref int *enableGS1_DataBarLimited*,
ref int *enableGS1_DataBarOmnidirectionalAndExpanded*,
ref int *transmitGS1_DataBarOmnidirectionalCodeID*,
ref int *transmitGS1_DataBarOmnidirectionalApplicationID*,
ref int *transmitGS1_DataBarOmnidirectionalCheckDigit*,
ref int *transmitGS1_DataBarLimitedCodeID*,
ref int *transmitGS1_DataBarLimitedApplicationID*,
ref int *transmitGS1_DataBarLimitedCheckDigit*,
ref int *transmitGS1_DataBarExpandedCodeID*);

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

enableGS1_DataBarLimited

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

0*	Disable
1	Enable

enableGS1_DataBarOmnidirectionalAndExpanded

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

0*	Disable
1	Enable

transmitGS1_DataBarOmnidirectionalCodeID

[in][out] A value that specifies whether to include RSS-14 Code ID "]e0" in the data being transmitted.

0	Do not transmit
1*	Transmit

transmitGS1_DataBarOmnidirectionalApplicationID

[in][out] A value that specifies whether to include RSS-14 Application ID in the data being transmitted.

0	Do not transmit
1*	Transmit

transmitGS1_DataBarOmnidirectionalCheckDigit

[in][out] A value that specifies whether to include the check digit in the RSS-14 data being transmitted.

0	Do not transmit
1*	Transmit

transmitGS1_DataBarLimitedCodeID

[in][out] A value that specifies whether to include RSS Limited Code ID "]e0" in the data being transmitted.

0	Do not transmit
1*	Transmit

transmitGS1_DataBarLimitedApplicationID

[in][out] A value that specifies whether to include RSS Limited Application ID in the data being transmitted.

0	Do not transmit
1*	Transmit

transmitGS1_DataBarLimitedCheckDigit

[in][out] A value that specifies whether to include the check digit in the RSS Limited data being transmitted.

0	Do not transmit
1*	Transmit

transmitGS1_DataBarExpandedCodeID

[in][out] A value that specifies whether to include RSS Expanded Code ID "]e0" in the data being transmitted.

0	Do not transmit
1*	Transmit

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.16 TELEPEN_1D

Telepen_1D	**CCD, Laser**

Purpose: To configure symbology settings — Telepen.

Syntax:

int Telepen_1D (int *rw*,
ref int *enable*,
ref int *enableAIM*);

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

enable

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

0*	Disable
1	Enable

enableOriginal

[in][out] A value that specifies whether to support Telepen in numeric format.

0	Enable AIM Telepen (Full ASCII)
1*	Enable original Telepen (Numeric)

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.17 EAN8_1D

Ean8_1D	**CCD, Laser**

Purpose: To configure symbology settings — EAN-8.

Syntax:

```
int Ean8_1D (int rw,
             ref int enable,
             ref int enableAddon2,
             ref int enableAddon5,
             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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

enableAddon2

[in][out] A value that specifies whether to support EAN-8 Addon 2.

0*	Disable
1	Enable

enableAddon5

[in][out] A value that specifies whether to support EAN-8 Addon 5.

0*	Disable
1	Enable

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit in the data being transmitted.

0	Do not transmit
1*	Transmit

convertToEAN13

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

0*	Do not convert
1	Convert

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.18 EAN13UPCA_1D

Ean13UpcA_1D	**CCD, Laser**

Purpose: To configure symbology settings — EAN-13 and UPC-A.

Syntax:

```
int Ean13UpcA_1D (int rw,
                  ref int enable,
                  ref int enableAddon2,
                  ref int enableAddon5,
                  ref int convertToISBN,
                  ref int convertToISSN,
                  ref int transmitEan13CheckDigit,
                  ref int convertToEAN13,
                  ref int transmitUpcACheckDigit,
                  ref int transmitUpcASystemNumber);
```

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

enable

[in][out] A value that specifies whether to enable EAN-13 and UPC-A.

0	Disable
1*	Enable

enableAddon2

[in][out] A value that specifies whether to support EAN-13 and UPC-A Addon 2.

0*	Disable
1	Enable

enableAddon5

[in][out] A value that specifies whether to support EAN-13 and UPC-A Addon 5.

0*	Disable
1	Enable

convertToISBN

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

0*	Do not convert
1	Convert

convertToISSN

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

0*	Do not convert
1	Convert

transmitEan13CheckDigit

[in][out] A value that specifies whether to include the check digit in the EAN-13 data being transmitted.

0	Do not transmit
1*	Transmit

convertToEAN13

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

0	Do not convert
1*	Convert

transmitUpcACheckDigit

[in][out] A value that specifies whether to include the check digit in the UPC-A data being transmitted.

0	Do not transmit
1*	Transmit

transmitUpcASystemNumber

[in][out] A value that specifies whether to include the system number in the UPC-A data being transmitted.

0	Do not transmit
1*	Transmit

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.19 GTIN_1D

Gtin_1D	**CCD, Laser**

Purpose: To configure GTIN settings.

Syntax:

int Gtin_1D (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0*	Disable
1	Enable

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.6.20 UPCE_1D

UpcE_1D	**CCD, Laser**

Purpose: To configure symbology settings — UPC-E.

Syntax:

```
int UpcE_1D (int rw,
             ref int enable,
             ref int enableAddon2,
             ref int enableAddon5,
             ref int enableUpcE1,
             ref int convertToUpcA,
             ref int transmitCheckDigit,
             ref int transmitSystemNumber,
             ref int enableUpcE1TripleCheck);
```

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

enable

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

0	Disable
1*	Enable

enableAddon2

[in][out] A value that specifies whether to support UPC-E Addon 2.

0*	Disable
1	Enable

enableAddon5

[in][out] A value that specifies whether to support UPC-E Addon 5.

0*	Disable
1	Enable

enableUpcE1

[in][out] A value that specifies whether to support UPC-E1 as well.

- Decoder version must be 1.02 or later.

0*	Disable
1	Enable

convertToUpcA

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

0*	Do not convert
1	Convert

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit in the data being transmitted.

0	Do not transmit
1*	Transmit

transmitSystemNumber

[in][out] A value that specifies whether to include the system number in the UPC-A data being transmitted.

0*	Do not transmit
1	Transmit

enableUpcE1TripleCheck

[in][out] A value that specifies whether to apply read redundancy (= triple check) on UPC-E1.

- Decoder version must be 1.02 or later.

0*	Do not apply
1	Apply read redundancy

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7 CONFIGURE SCAN ENGINE – 2D SCAN ENGINE

UserPreferences_2D_4507() and **MiscellaneousOption_2D_4507** are provided for configuring reader settings. The rest functions are provided for changing symbology settings.

2.7.1 PREFERENCES

UserPreferences_2D_4507	**2D**

Purpose: To configure reader settings.

Syntax:

int UserPreferences_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

laserOnTime

[in][out] A value that specifies the maximum time decode processing continues during a scan attempt.

5~99	30* (0.1 second)

timeoutBetweenSameBarcode Reserved

triggerMode

[in][out] A value that specifies the trigger mode.

8*	Laser mode

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

SymbologySecurityLevel_2D_4507	**2D**

Purpose — To configure decode redundancy settings.

Syntax

```
int SymbologySecurityLevel _2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

redundancyLevel

[in][out] A value that specifies the decode redundancy. Higher redundancy levels are selected for decreasing levels of 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 except for the following barcodes must be successfully read twice before being decoded. The following barcodes 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 the decode security level, which is appropriate for the barcode quality when reading delta barcodes such as Code 128, Code 93, UPC/EAN.

0*	Security Level 0 – This default setting allows the scan engine to operate in its most aggressive state, providing sufficient security in decoding most "in-spec" barcodes.
1	Security Level 1 – Select this option if misdecodes occur. This level should eliminate most misdecodes.
2	Security Level 2 – Select this option if Security Level 1 fails to eliminate misdecodes.
3	Security Level 3 – Select this option if Security Level 2 also fails to eliminate 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, preventing 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 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

MiscellaneousOption_2D_4507	**2D**

Purpose

To configure more reader settings.

Syntax

int MiscellaneousOption _2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

transmitCodeIdChar

[in][out] A value that specifies whether to transmit a specific set of 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 1.

Otherwise, it returns 0.

- 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, 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`"). ▸ In GS1-128 emulation mode, RSS is transmitted using Code 128 rules (i.e. "`]C1`"). ▸ UPC portion of composite is transmitted using UPC rules.
PDF417, Micro PDF417	0	Scan engine is set to conform to protocol defined in 1994 PDF417 symbology specifications. ▸ When this option is transmitted, the receiver cannot reliably determine whether ECIs have been invoked or whether data byte 92_{DEC} has been doubled in transmission.
	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. ▸ When decoders are set to this mode, unbuffered Macro symbols and symbols requiring the decoder to convey ECI escape sequences cannot be transmitted.
	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 `]L2`ABCD.	
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_4507

Upc_2D_4507	**2D**

Purpose: To configure symbology settings — UPC-A and UPC-E.

Syntax:

```
int Upc_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enableUpcA

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

0	Disable
1*	Enable

enableUpcE

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

0	Disable
1*	Enable

enableUpcE1

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

0*	Disable
1	Enable

enableAddons

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

0*	Ignore Addons
1	Decode only with Addons
2	Decode with Addons (= Auto-discriminate)

addonsRedundancy

[in][out] A value that specifies the 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 include the check digit in the UPC-A data being transmitted.

0	Do not transmit
1*	Transmit

transmitUpcECheckDigit

[in][out] A value that specifies whether to include the check digit in the UPC-E0 data being transmitted.

0	Do not transmit
1*	Transmit

transmitUpcE1CheckDigit

[in][out] A value that specifies whether to include the check digit in the UPC-E1 data being transmitted.

0	Do not transmit
1*	Transmit

preambleUpcA

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

0	No preamble
1*	Transmit System Number
2	Transmit System Number & Country Code

preambleUpcE

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

0	No preamble
1*	Transmit System Number
2	Transmit System Number & Country Code

preambleUpcE1

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

0	No preamble
1*	Transmit System Number
2	Transmit System Number & Country Code

convertUpcEtoA

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

0*	Do not convert
1	Convert

convertUpcE1toA

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

0*	Do not convert
1	Convert

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*	Disable
1	Enable

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.3 EANJAN_2D_4507

EanJan_2D_4507	**2D**

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

Syntax:

int EanJan_2D_4507 (int *rw*,
ref int *enableEanJan8*,
ref int *enableEanJan13*,
ref int *enableBooklandEan*,
ref int *enableAddons*,
ref int *addonsRedundancy*,
ref int *enableEanJan8Extended*,
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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enableEanJan8

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

0	Disable
1*	Enable

enableEanJan13

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

0	Disable
1*	Enable

enableBooklandEan

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

- EAN/JAN-13 must be enabled first!

0	Disable
1*	Enable

enableAddons

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

0*	Ignore Addons
1	Decode only with Addons
2	Decode with Addons (= Auto-discriminate)

addonsRedundancy

[in][out] A value that specifies the 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*	Do not convert
1	Convert

uccCouponExtendedCode

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

0*	Disable
1	Enable

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.4 CODE39_2D_4507

Code39_2D_4507	**2D**

Purpose: To configure symbology settings — Code 39.

Syntax:

```
int Code39_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

enableTrioptic

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

0*	Disable
1	Enable

convertToCode32

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

0*	Do not convert
1	Convert

prefixCode32

[in][out] A value that specifies whether to include the prefix in the Code 32 data being transmitted.

0*	Do not transmit
1	Transmit

checkDigitVerification

[in][out] A value that specifies whether to perform check digit verification when decoding barcodes.

0*	Do not verify
1	Verify

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit in the data being transmitted.

- Check digit verification must be enabled first.

0*	Do not transmit
1	Transmit

fullASCII

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

0*	Standard Code 39
1	Code 39 Full ASCII

length1

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

- If "length2" is greater than "length1", it applies "Max/Mix length format"; otherwise, it applies "Fixed length format".

0~255	4*

length2

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

0~255	55*

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.5 CODE93_2D_4507

Code93_2D_4507	**2D**

Purpose: To configure symbology settings — Code 93.

Syntax:

int Code93_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

length1

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

- If "length2" is greater than "length1", it applies "Max/Mix length format"; otherwise, it applies "Fixed length format".

0~255	4*

length2

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

0~255	55*

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.6 INTERLEAVED2OF5_2D_4507

Interleaved2Of5_2D_4507	**2D**

Purpose: To configure symbology settings — Interleaved 25.

Syntax:

```
int Interleaved2Of5_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

length1

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

- If "length2" is greater than "length1", it applies "Max/Mix length format"; otherwise, it applies "Fixed length format".

0~255	4*

length2

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

0~255	55*

checkDigitVerification

[in][out] A value that specifies whether to perform check digit verification when decoding barcodes.

0*	Disable
1	USS Check Digit
2	OPCC Check Digit

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit in the data being transmitted.

0*	Do not transmit
1	Transmit

convertToEAN13

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

- Check digit verification must be disabled first! This only works when the barcode has a leading zero and a valid EAN-13 check digit.

0*	Do not convert
1	Convert

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.7 INDUSTRIAL2OF5_2D_4507

Industrial2Of5_2D_4507	**2D**

Purpose

To configure symbology settings — Industrial 25 (= Discrete 25).

Syntax

int Industrial2Of5_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

length1

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

- If "length2" is greater than "length1", it applies "Max/Mix length format"; otherwise, it applies "Fixed length format".

0~255	4*

length2

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

0~255	55*

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.8 MATRIX2OF5_2D_4507

Matrix2Of5_2D_4507	**2D**

Purpose: To configure symbology settings — Matrix 25.

Syntax:

```
int Matrix2Of5_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

length1

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

- If "length2" is greater than "length1", it applies "Max/Mix length format"; otherwise, it applies "Fixed length format".

0~255	4*

length2

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

0~255	55*

redundancy

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

0*	Disable
1	Enable

checkDigitVerification

[in][out] A value that specifies whether to perform check digit verification when decoding barcodes.

0*	Do not verify
1	Verify

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit in the data being transmitted.

0*	Do not transmit
1	Transmit

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.9 CHINESE2OF5_2D_4507

Chinese2Of5_2D_4507	**2D**

Purpose: To configure symbology settings — Chinese 25.

Syntax:

int Chinese2Of5_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.10 CODABAR_2D_4507

Codabar_2D_4507	**2D**

Purpose: To configure symbology settings — Codabar.

Syntax:

```
int Codabar_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

length1

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

- If "length2" is greater than "length1", it applies "Max/Mix length format"; otherwise, it applies "Fixed length format".

0~255	4*

length2

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

0~255	55*

enableCLSI_Editing

[in][out] A value that specifies whether to perform CLSI editing when decoding barcodes.

0*	Disable
1	Enable

- When applied, the 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 perform NOTIS editing when decoding barcodes.

0*	Disable
1	Enable

- NOTIS Editing is to strip the start/stop characters, which equals to "Disable Transmit Start/Stop Characters".

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.11 MSI_2D_4507

Msi_2D_4507	**2D**

Purpose: To configure symbology settings — MSI.

Syntax:

```
int Msi_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

length1

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

- If "length2" is greater than "length1", it applies "Max/Mix length format"; otherwise, it applies "Fixed length format".

0~255	4*

length2

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

0~255	55*

checkDigitVerification

[in][out] A value that specifies how to perform check digit verification when decoding barcodes.

0*	Verify one check digit
1	Verify two check digits

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit in the data being transmitted.

0*	Do not transmit
1	Transmit

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 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.12 GS1_DATABAR_2D_4507

GS1_DataBar_2D_4507	**2D**

Purpose: To configure symbology settings — GS1 DataBar (= RSS family).

Syntax:

int GS1_DataBar_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enableGS1_DataBarOmnidirectional

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

0	Disable
1*	Enable

enableGS1_DataBarLimited

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

0	Disable
1*	Enable

enableGS1_DataBarExpanded

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

0	Disable
1*	Enable

convertToUpcEan

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

0*	Do not convert
1	Convert

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.13 CODE128_2D_4507

Code128_2D_4507	**2D**

Purpose: To configure symbology settings — Code 128.

Syntax:

```
int Code128_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enableCode128

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

0	Disable
1*	Enable

enableGS1_128

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

0	Disable
1*	Enable

enableISBT128

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

0	Disable
1*	Enable

enableIsbtConcatenation

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

0*	Disable (= It will not concatenate pairs of ISBT barcodes it encounters.)
1	Enable (= There must be two ISBT barcodes in order for the scanner to decode and perform concatenation. It does not decode single ISBT barcodes.)
2	Auto-discriminate (=It decodes and concatenates pairs of ISBT barcodes immediately. If only a single ISBT barcode is present, the scanner must decode 10 times before transmitting its data to confirm that there is no additional ISBT barcode.)

isbtConcatenationRedundancy

[in][out] A value that specifies the 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 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.14 CODE11_2D_4507

Code11_2D_4507	**2D**

Purpose: To configure symbology settings — Code 11.

Syntax:

```
int Code11_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enable

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

0	Disable
1*	Enable

numberOfCheckDigits

[in][out] A value that specifies whether to perform check digit verification when decoding barcodes.

0*	None
1	One check digit
2	Two check digits

transmitCheckDigit

[in][out] A value that specifies whether to include the check digit in the data being transmitted.

0*	Do not transmit
1	Transmit

length1

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

- If "length2" is greater than "length1", it applies "Max/Mix length format"; otherwise, it applies "Fixed length format".

0~255	4*

length2

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

0~255	55*

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

-253	E_WRONG_READER_TYPE

2.7.15 POSTALCODE_2D_4507

PostalCode_2D_4507	**2D**

Purpose: To configure symbology settings — Postal Codes.

Syntax:

int PostalCode_2D_4507 (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	Disable
1*	Enable

enableUSPlanet

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

0	Disable
1*	Enable

enableUKPostal

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

0	Disable
1*	Enable

transmitUKCheckDigit

[in][out] A value that specifies whether to include the check digit in the UK Postal data being transmitted.

0	Do not transmit
1*	Transmit

enableJapanPostal

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

0	Disable
1*	Enable

enableAustralianPostal

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

0	Disable
1*	Enable

enableDutchPostal

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

0	Disable
1*	Enable

transmitUSCheckDigit

[in][out] A value that specifies whether to include the check digit in the US Postal data being transmitted.

0	Do not transmit
1*	Transmit

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.16 COMPOSITE_2D_4507

Composite_2D_4507	**2D**

Purpose: To configure symbology settings — Composite barcodes.

Syntax:

int Composite_2D_4507 (int *rw*,
ref int *enableCC_C*,
ref int *enableCC_AB*,
ref int *enableTLC39*,
ref int *enableUpcMode*,
ref int *enableBeepMode*,
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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enableCC_C

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

0	Disable
1*	Enable

enableCC_AB

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

0*	Disable
1	Enable

enableTLC39

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

0*	Disable
1	Enable

enableUpcMode

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

- UPC barcodes can be "linked" with a 2D barcode during transmission as if they were one barcode.

0	UPC Never Linked (= discard 2D portion if there is any)
1*	UPC Always Linked (= discard whole barcode if 2D is not present)
2	Auto-discriminate UPC Composites

enableBeepMode

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

- One or more beeps are given to indicate a composite barcode is decoded.

0	Single Beep after both are decoded
1*	Beep as each code type is decoded
2	Double Beep after both are decoded

enableEmulationModel

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

0*	Disable
1	Enable

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.17 SYMBOLOGIES_2D_4507

Symbologies_2D_4507	**2D**

Purpose: To configure 2D symbology settings.

Syntax:

int Symbologies_2D_4507 (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**	Get settings
'w'	**Reader.ReaderEngineAPI.WRITE_PARAM**	Set settings

enablePDF417

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

0	Disable
1*	Enable

enableMicroPDF417

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

0*	Disable
1	Enable

enableCode128Emulation

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

0*	Disable
1	Enable

- AIM Code Identifier must be enabled first!
- When applied, the MicroPDF417 barcodes are transmitted as if they were encoded in Code 128 barcodes with one of these prefixes:

The first codeword of MicroPDF417 is 903-907, 912, 914, 915:
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	Disable
1*	Enable

enableDataMatrixInverse

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

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

decodeMirrorImage Reserved

enableMaxicode

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

0	Disable
1*	Enable

enableQRCode

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

0	Disable
1*	Enable

enableQRCodeInverse

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

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

enableMicroQR

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

0	Disable
1*	Enable

enableAztec

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

0	Disable
1*	Enable

enableAztecInverse

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

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

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

2.7.18 MACROPDF_2D_4507

MacroPDF_2D_4507	**2D**

Purpose: To configure symbology settings — Macro PDF.

Syntax:

```
int MacroPDF_2D_4507 (int rw,
                      ref int modeTransmitDecode,
                      ref int transmitControlHeader,
                      ref int enableEscCharacter);
```

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

modeTransmitDecode

[in][out] A value that specifies how to handle Macro PDF decoding.

- Macro PDF is a special feature for concatenating multiple PDF barcodes into one file, known as Macro PDF417 or Macro MicroPDF417.

0	Buffer All Symbols / Transmit Macro PDF When Complete ▸ Transmit all decoded data from an entire Macro PDF sequence only when the entire sequence is scanned and decoded. If the decoded data exceeds the limit of 50 symbols, no transmission because the entire sequence was not scanned!
1	Transmit Any Symbol in Set / No Particular Order ▸ Transmit data from each Macro PDF symbol as decoded, regardless of the sequence.
4*	Passthrough All Symbols ▸ Transmit and decode all Macro PDF symbols and perform no processing. In this mode, the host is responsible for detecting and parsing the Macro PDF sequences.

transmitControlHeader

[in][out] A value that specifies whether to transmit the control header, which contains the segment index and the file ID.

0*	Disable
1	Enable

- This parameter has no effect when it is set to "Passthrough All Symbols".
- Enable this when it is set to "Transmit Any Symbol in Set / No Particular Order".
- Disable this when it is set to "Buffer All Symbols / Transmit Macro PDF When Complete".

enableEscCharacter

[in][out] A value that specifies whether to enable the backslash "\" as an Escape character for systems that can process transmissions containing special data sequences. It will format special data according to the Global Label Identifier (GLI) protocol, which only affects the data portion of a Macro PDF symbol transmission. The Control Header, if enabled, is always sent with GLI formatting.

0*	Disable
1	Enable

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

-253	E_WRONG_READER_TYPE

Note: When printing barcodes, keep each Macro PDF sequence separate, as each has a unique identifier. Do not mix barcodes from several Macro PDF sequences, even if they encode the same data. When you scan Macro PDF sequences, scan the entire Macro PDF sequence without interruption!

2.8 RESET READER

ResetReaderToDefault

Purpose: To reset the barcode reader.

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

Parameters: *target*

[in] A value that specifies the reader type.

7	DC_READER_BC	Reset the barcode reader only.
32	DC_READER_RFID	Reset the RFID reader only.
255	ALL_DEVICE	Reset both readers.

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 1.

Otherwise, it returns 0.

- 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 the specified reader to defaults.

See Also: InitReader

Chapter 3

SYSTEM API

For data communications via the RS-232 cable, open COM 7 on the mobile computer.

Find the necessary files on the CD-ROM before you start to write your application.

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

Note: For signature capture, it does not require SystemApi_Ce_Net.dll.

IN THIS CHAPTER

3.1 SYSTEM SETTINGS

3.1.1 API VERSION

GetAPIVersion	
Purpose	To get information about system APIs.
Syntax	**int GetAPIVersion (ref string** *apiVer***);**
Parameters	*apiVer* [out] Pointer to a buffer where the version information is stored.

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

Example for VB	`Dim b1 As Integer`
	`Dim apiver AS String = ""`
	`b1 = Cipherlab.SystemAPI.Member.GetAPIVersion(apiver)`

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

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.

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.

0x01	UUID inaccessible

3.1.3 DEVICE NAME

GetSysDevName

Purpose	To get the device name that is identical to the one displayed in **Start | Settings | Control Panel | 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 = "CPT9600CE";`
	`b1 = Cipherlab.SystemAPI.Member.SetSysDevName(setname);`

Example for VB	`Dim b1 As Integer`
	`Dim setname as String = "CPT9600CE"`
	`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)

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 open registry
2	Fail to set registry

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)

3.1.7 CIPHERLAB ORIGINAL DEVICE

GetCipherlabDeviceID

Purpose	To get the CipherLab original device ID.
Syntax	**int GetCipherlabDeviceID (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.GetCipherlabDeviceID(ref deviceName);`

Example for VB	`Dim b1 As Integer`
	`Dim deviceName As String = ""`
	`b1 = Cipherlab.SystemAPI.Member.GetCipherlabDeviceID(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, this device ID consists of a 2-digit prefix "96" and 14-digit serial number.

GetCipherlabPlatformName

Purpose	To get the CipherLab platform.
Syntax	**int GetCipherlabPlatformName (ref int** *platformInfo***);**
Parameters	*platformInfo* [out] Pointer to a buffer where the information is stored.

Example for C#	`int b1 = 0;`
	`int deviceName = 0;`
	`b1 = Cipherlab.SystemAPI.Member.GetCipherlabPlatformName(`
	`ref deviceName);`

Example for VB	`Dim b1 As Integer`
	`Dim deviceName As Integer = 0`
	`b1 = Cipherlab.SystemAPI.Member.GetCipherlabPlatformName(deviceName)`

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

4	ERROR_PARAMETER (Wrong parameter)

Remarks	For 9600, the platform information is always "`0x43505496`".

3.2 TASKBAR

3.2.1 ALWAYS HIDE

SetTaskBarAlwaysHide

Purpose: To set the Taskbar hidden all the time.

Syntax: **int SetTaskBarAlwaysHide (byte** *hide***);**

Parameters: *hide*

[in] Byte variable

0	Taskbar always available (= Start | Settings | Taskbar and Start Menu: "Always on top" enabled + "Auto hide" disabled)
1	Taskbar always hide in application programs (= Start | Settings | Taskbar and Start Menu: "Always on top" disabled + "Auto hide" enabled)

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetTaskBarAlwaysHide(0);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetTaskBarAlwaysHide(0)
```

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

3.2.2 AUTO HIDE

SetTaskBarAutoHide

Purpose: To set the Taskbar hidden automatically.

Syntax: **int SetTaskBarAutoHide (byte** *hide***);**

Parameters: *hide*

[in] Byte variable

0	Taskbar availability depends on "Always on top" setting (= Start | Settings | Taskbar and Start Menu: "Auto hide" disabled)
1	Taskbar hide automatically (= Start | Settings | Taskbar and Start Menu: "Auto hide" enabled)

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetTaskBarAutoHide(1);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetTaskBarAutoHide(1)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

2	Fail to open registry
4	Fail to set registry

3.3 STATUS INDICATION

3.3.1 BUZZER

SetBuzzerOFF

Purpose	To turn off the buzzer.
Syntax	**int SetBuzzerOFF ();**
Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.SetBuzzerOFF();`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.SetBuzzerOFF()`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	DeviceIOControl Error
2	CreateFile Error

SetBuzzerON

Purpose	To turn on the buzzer and set its frequency.
Syntax	**int SetBuzzerON (uint** *hz***);**
Parameters	*hz* [in] Unsigned integer variable

0x01	Audio frequency 500Hz
0x02	Audio frequency 1kHz
0x03	Audio frequency 1.5kHz
…	…
0x0A	Audio frequency 5kHz

Example for C#	`int b1 = 0;` `b1 = Cipherlab.SystemAPI.Member.SetBuzzerON(0x01);`
Example for VB	`Dim b1 As Integer` `b1 = Cipherlab.SystemAPI.Member.SetBuzzerON(1)`
Return Value	If successful, it returns 1. Otherwise, it returns 0. ▸ Call GetErrorCode() to find the error condition encountered:

1	DeviceIOControl Error
2	CreateFile Error
4	ERROR_PARAMETER (Wrong parameter)

3.3.2 LED LIGHT

SetLEDLight

Purpose	To manipulate the status light (LED).
Syntax	**int SetLEDLight (byte** *color*, **int** *onOff*);
Parameters	*color* [in] Byte variable

0x01	Red LED
0x02	Green LED
0x03	Blue 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	DeviceIOControl Error
2	CreateFile Error

Remarks

The time interval required between on and off must be greater than 400 milliseconds.

3.3.3 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:

4	ERROR_PARAMETER (Wrong parameter)

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:

4	ERROR_PARAMETER (Wrong parameter)

3.4 LCD BACKLIGHT

3.4.1 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 = 60;`
	`bklctl.actimeout = 60;`
	`bklctl.backlightontap = 0;`
	`bklctl.lightlevel = 3;`
	`bklctl.acbacklightontap = 1;`
	`bklctl.aclightlevel = 6;`
	`bklctl.usebattery = 1;`
	`bklctl.useext = 1;`
	`b1 = Cipherlab.SystemAPI.Member.SetBacklightCTL(bklctl);`

Example for VB	`Dim b1 As Integer`
	`Dim bklctl As New Member.BklCtl`
	`bklctl.batttimeout = 60`
	`bklctl.actimeout = 60`
	`bklctl.backlightontap = 0`
	`bklctl.lightlevel = 3`
	`bklctl.acbacklightontap = 1`
	`bklctl.aclightlevel = 6`
	`bklctl.usebattery = 1`
	`bklctl.useext = 1`
	`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)
16	Out of range, backlight level in battery mode.
32	Out of range, backlight level in AC mode.

3.4.2 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 6 (bright).

System defaults are level 3 in battery mode and level 6 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 6 (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.

3.4.3 BACKLIGHT STATE

GetBacklightST

Purpose: To find out the current backlight state.

Syntax: **int GetBacklightST (ref byte** *state***);**

Parameters: *state*

[out] A byte variable that stores the information.

0	Backlight OFF
1	Backlight ON

Example for C#

```
int b1 = 0;
byte bstate = new byte();
b1 = Cipherlab.SystemAPI.Member.GetBacklightST(ref bstate);
```

Example for VB

```
Dim b1 As Integer
Dim bstate As New Byte
b1 = Cipherlab.SystemAPI.Member.GetBacklightST(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)

SetBacklightST

Purpose: To set the backlight state.

Syntax: **int SetBacklightST (byte** *state***);**

Parameters: *state*

[in] Byte variable

0	Backlight OFF
1	Backlight ON

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetBacklightST(1);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetBacklightST(1)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

3.4.4 RESET BACKLIGHT TO DEFAULT

SetBacklightDefault

Purpose	To reset the settings of backlight control to defaults.
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

3.4.5 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=60;
	public uint lightlevel=6;
	public uint actimeout=60;
	public uint aclightlevel=6;
	public uint backlightontap=1;
	public uint acbacklightontap=1;
	public uint usebattery=1;
	public uint useext=1;

See Also	SetBacklightDefault, BklCtl

3.4.6 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=15;
	public uint lightlevel=1;
	public uint actimeout=15;
	public uint aclightlevel=1;
	public uint backlightontap=0;
	public uint acbacklightontap=1;
	public uint usebattery=1;
	public uint useext=1;

See Also	SetBacklightDefault, BklCtl

3.5 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)

3.6 TOUCH PANEL

3.6.1 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

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 locked

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
2	Fail to access touch panel
4	Full key locked
8	Fail to set touch panel

Note: You cannot lock both the touch panel and keypad at the same time!

3.6.2 DISPLAY TYPE

GetDisplayType

Purpose: To get the current resolution of touch panel.

Syntax: **int GetDisplayType (ref byte** *panelMode*);

Parameters: *panelMode*

[out] A byte variable that stores the information.

0	QVGA for 320 x 240 pixels
1	VGA for 640 x 480 pixels

Example for C#

```
int b1 = 0;
byte paneltype = new byte();
b1 = Cipherlab.SystemAPI.Member.GetDisplayType(ref paneltype);
```

Example for VB

```
Dim b1 As Integer
Dim paneltype As New Byte
b1 = Cipherlab.SystemAPI.Member.GetDisplayType(paneltype)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

1	(Read Error)

SetDisplayType

Purpose: To change the resolution of touch panel.

Syntax: **int SetDisplayType (byte** *panelMode*);

Parameters: *panelMode*

[in] Byte variable

0	QVGA for 320 x 240 pixels
1	VGA for 640 x 480 pixels

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetDisplayType(0);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetDisplayType(0)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

2	Registry Open Error
3	VGA mode not supported on QVGA panel.

Remarks: For the change to take effect, it will automatically warm boot the system.

3.6.3 SCREEN LOCK

StartScreenLock

Purpose	To apply device lock.
Syntax	**int StartScreenLock (uint** *timeIdle***);**
Parameters	*timeIdle* [in] Unsigned integer variable A value that specifies the time interval before the device lock is applied.

0	Do not apply
1~10000	(in units of second)

Example for C#

```
int b1 = 0;
uint timeIdle = 0;
b1 = Cipherlab.SystemAPI.Member.StartScreenLock(timeIdle);
```

Example for VB

```
Dim b1 As Integer
Dim timeIdle As UInteger = 0
b1 = Cipherlab.SystemAPI.Member.StartScreenLock(timeIdle)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

1	Fail to get handle
4	ERROR_PARAMETER (Wrong parameter)

Remarks

When device lock is applied and the device is idle for the specified period of time, there will be a lock displayed on the Start Screen. It requires pressing "F5" to unlock. That is, press the function key and then the number key "5".

3.7 KEYPAD

3.7.1 KEYPAD TYPE

GetKeypadType

Purpose: To find out the keypad type.

Syntax: **int GetKeypadType (ref byte** *keypadLayout***);**

Parameters: *keypadLayout*

[out] A byte variable that stores the information.

0	29-key
1	Reserved
2	QWERTY

Example for C#

```
int b1 = 0;
byte keypadtype = new byte();
b1 = Cipherlab.SystemAPI.Member.GetKeypadType(ref keypadtype);
```

Example for VB

```
Dim b1 As Integer
Dim keypadtype As New Byte
b1 = Cipherlab.SystemAPI.Member.GetKeypadType(keypadtype)
```

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

3.7.2 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
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 (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
Remarks	For 29-key 9600, such repeat rate will be applied to black-coded keys only. That is, it is not applicable to blue-coded characters when in Alpha mode and orange-coded F1~F12 when in Function mode.

3.7.3 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

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

Remarks: KEY_ALL = KEY_GENERAL + KEY_ALPHA + KEY_FN + KEY_SCAN + KEY_POWER + KEY_LSTRIGGER + KEY_RSTRIGGER + KEY_SHIFT

3.7.4 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

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)

3.7.5 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	Upper-case	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

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	Upper-case	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
4	ERROR_PARAMETER (Wrong parameter)

3.8 AUDIO & HEADSET

3.8.1 AUDIO TYPE

SetAudioPath

Purpose	To set the audio path — for general use, GSM application, etc.
Syntax	**int SetAudioPath (uint** *uiPath***);**
Parameters	*uiPath* [in] Unsigned integer variable

0x00	System sounds, such as Good Read feedback, music, etc.
0x10	GSM
0x20	Reserved

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetAudioPath(0x00);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetAudioPath(0)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

1	DeviceIOControl Fail
2	CreateFile Error

See Also: GetHeadsetState, SetBTAudioState

3.8.2 WIRED HEADSET

GetHeadsetState

Purpose	To check whether a wired headset is inserted to the headset jack.
Syntax	**int GetHeadsetState (ref byte** *state***);**
Parameters	*state* [out] A byte variable that stores the information.

0	No wired headset is found.
1	A wired headset is present.

Example for C#

```
int b1 = 0;
byte headsetState = new byte();
b1 = Cipherlab.SystemAPI.Member.GetHeadsetState(ref headsetState);
```

Example for VB

```
Dim b1 As Integer
Dim headsetState As New Byte
b1 = Cipherlab.SystemAPI.Member.GetHeadsetState(headsetState)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

1	DeviceIOControl Fail
2	CreateFile Error

See Also

SetAudioPath

3.8.3 BLUETOOTH HEADSET

SetBTAudioState

Purpose	To set the state of Bluetooth headset.
Syntax	**int SetBTAudioState (int** *onOff***);**
Parameters	*onOff* [in] Integer variable

0	Disable Bluetooth headset
1	Enable Bluetooth headset

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetBTAudioState(0);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetBTAudioState(0)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

1	DeviceIOControl Fail
2	CreateFile Error

See Also: SetAudioPath, StartBluetooth

3.9 WIRELESS LAN

3.9.1 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

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

3.9.2 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)
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)

3.9.3 DOMAIN, SDK & MAC INFORMATION

GetCurrentDomain

Purpose	To find out the regulatory domain or domains for which the radio is configured.
Syntax	**int 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)

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 1.

Otherwise, it returns 0.

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

3.9.4 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.

3.9.5 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.

3.9.6 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.

GetWiFiBSSIDList	
Purpose	To obtain the BSSID list. (The maximum arrary size is 50.)
Syntax	**int GetWiFiBSSIDList (ref BSSIDInfo[]** *bssid***);**
Parameters	*bssid* [out] BSSIDInfo structure that stores the information.

Example for C#	`int b1 = 0;`
	`Member.BSSIDInfo[] bssid = new Member.BSSIDInfo[50];`
	`b1 = Cipherlab.SystemAPI.Member.GetWifiBSSIDList(ref bssid);`

Example for VB	`Dim b1 As Integer`
	`Dim bssid(50) As Member.BSSIDInfo`
	`b1 = Cipherlab.SystemAPI.Member.GetWifiBSSIDList(bssid)`

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

3.9.7 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
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.

3.9.8 EDITING 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 = 5;`
	`sdccfg.authType = Member.AUTH.AUTH_NETWORK_EAP;`
	`sdccfg.eapType = Member.EAPTYPE.EAP_PEAPMSCHAP;`
	`sdccfg.powerSave = Member.POWERSAVE.POWERSAVE_MAX;`
	`sdccfg.wepType = Member.WEPTYPE.WEP_CKIP;`
	`sdccfg.bitRate = Member.BITRATE.BITRATE_48;`
	`sdccfg.radioMode = Member.RADIOMODE.RADIOMODE_BGA;`
	`sdccfg.userName.buffer = "";`
	`sdccfg.userName.offset = 0;`
	`sdccfg.userName.size = 0;`
	`sdccfg.userPwd.buffer = "";`
	`sdccfg.userPwd.offset = 0;`
	`sdccfg.userPwd.size = 0;`
	`sdccfg.PSK.buffer = "";`
	`sdccfg.PSK.offset = 0;`
	`sdccfg.PSK.size = 0;`
	`sdccfg.WEPKeys.buffer = "";`
	`sdccfg.WEPKeys.offset = 0;`
	`sdccfg.WEPKeys.size = 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 = 5`
	`sdccfg.authType = Member.AUTH.AUTH_NETWORK_EAP`
	`sdccfg.eapType = Member.EAPTYPE.EAP_PEAPMSCHAP`

```
sdccfg.powerSave = Member.POWERSAVE.POWERSAVE_MAX
sdccfg.wepType = Member.WEPTYPE.WEP_CKIP
sdccfg.bitRate = Member.BITRATE.BITRATE_48
sdccfg.radioMode = Member.RADIOMODE.RADIOMODE_BGA
sdccfg.userName.buffer = ""
sdccfg.userName.offset = 0
sdccfg.userName.size = 0
sdccfg.userPwd.buffer = ""
sdccfg.userPwd.offset = 0
sdccfg.userPwd.size = 0
sdccfg.PSK.buffer = ""
sdccfg.PSK.offset = 0
sdccfg.PSK.size = 0
sdccfg.WEPKeys.buffer = ""
sdccfg.WEPKeys.offset = 0
sdccfg.WEPKeys.size = 0
b1 = Cipherlab.SystemAPI.Member.AddConfig(sdccfg)
```

Return Value	If successful, it returns 1. Otherwise, it returns 0.
Remarks	This function cannot be used to delete ThirdPartyConfig.

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	This function cannot be used to delete ThirdPartyConfig.

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.

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.

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.
Remarks	This function cannot be used to modify ThirdPartyConfig; you must use Set3rdPartyConfig() instead.

3.9.9 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.

3.9.10 3RD PARTY CONFIG SETTINGS

Get3rdPartyConfig

Purpose	To retrieve the settings of a third party configuration.
Syntax	**int Get3rdPartyConfig (ref SDC3rdPartyConfig** *config3rd***);**
Parameters	*config3rd* [out] SDC3rdPartyConfig structure that stores the information.

Example for C#	`int b1 = 0;`
	`Member.SDC3rdpartyconfig sdc3rdcfg = new Member.SDC3rdpartyconfig();`
	`b1 = Cipherlab.SystemAPI.Member.Get3rdPartyConfig(ref sdc3rdcfg);`

Example for VB	`Dim b1 As Integer`
	`Dim sdc3rdcfg As New Member.SDC3rdpartyconfig`
	`b1 = Cipherlab.SystemAPI.Member.Get3rdPartyConfig(sdc3rdcfg)`

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

Set3rdPartyConfig

Purpose	To overwrite the settings of a third party configuration.
Syntax	**int Set3rdPartyConfig (SDC3rdPartyConfig** *config3rd***);**
Parameters	*config3rd* [in] SDC3rdPartyConfig structure that stores the information.

Example for C#	`int b1 = 0;`
	`Member.SDC3rdpartyconfig sdc3rdcfg = new Member.SDC3rdpartyconfig();`
	`Cipherlab.SystemAPI.Member.Get3rdPartyConfig(ref sdc3rdcfg);`
	`sdc3rdcfg.txPower = 5;`
	`b1 = Cipherlab.SystemAPI.Member.Set3rdPartyConfig(sdc3rdcfg);`

Example for VB	`Dim b1 As Integer`
	`Dim sdc3rdcfg As New Member.SDC3rdpartyconfig`
	`Cipherlab.SystemAPI.Member.Get3rdPartyConfig(sdc3rdcfg)`
	`sdc3rdcfg.txPower = 5`
	`b1 = Cipherlab.SystemAPI.Member.Set3rdPartyConfig(sdc3rdcfg)`

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

3.9.11 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.

3.9.12 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
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.

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.

3.9.13 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] CONFIG_FILE_INFO 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, global settings, and third party configuration to a specified file.
Syntax	**SDCERR 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, global settings, and third party configuration from a specified file.
Syntax	**SDCERR 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.

3.9.14 WEP KEY

GetWEPKey

Purpose: To retrieve a WEP key.

Syntax:

int GetWEPKey (ref 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.

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(
ref 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(
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
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(

ref 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.

3.9.15 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(ref 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.

3.9.16 LEAP CREDENTIALS

GetLEAPCred	
Purpose	To retrieve the LEAP creditials.
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 creditials.
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(`
	`ref 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.

3.9.17 EAP-FAST CREDENTIALS

GetEAPFASTCred	
Purpose	To retrieve the EAP-FAST creditials.
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 creditials.
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(`
	`ref 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.

3.9.18 PEAP-GTC CREDENTIALS

GetPEAPGTCCred

Purpose: To retrieve the PEAP-GTC creditials.

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 creditials.

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(
ref 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.

3.9.19 PEAP-MSCHAP CREDENTIALS

GetPEAPMSCHAPCred

Purpose: To retrieve the PEAP-MSCHAP creditials.

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 creditials.

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(
ref 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.

3.9.20 EAP-TLS CREDENTIALS

GetEAPTLSCred

Purpose: To retrieve the EAP-TLS creditials.

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 creditials.

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(
ref 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)
```

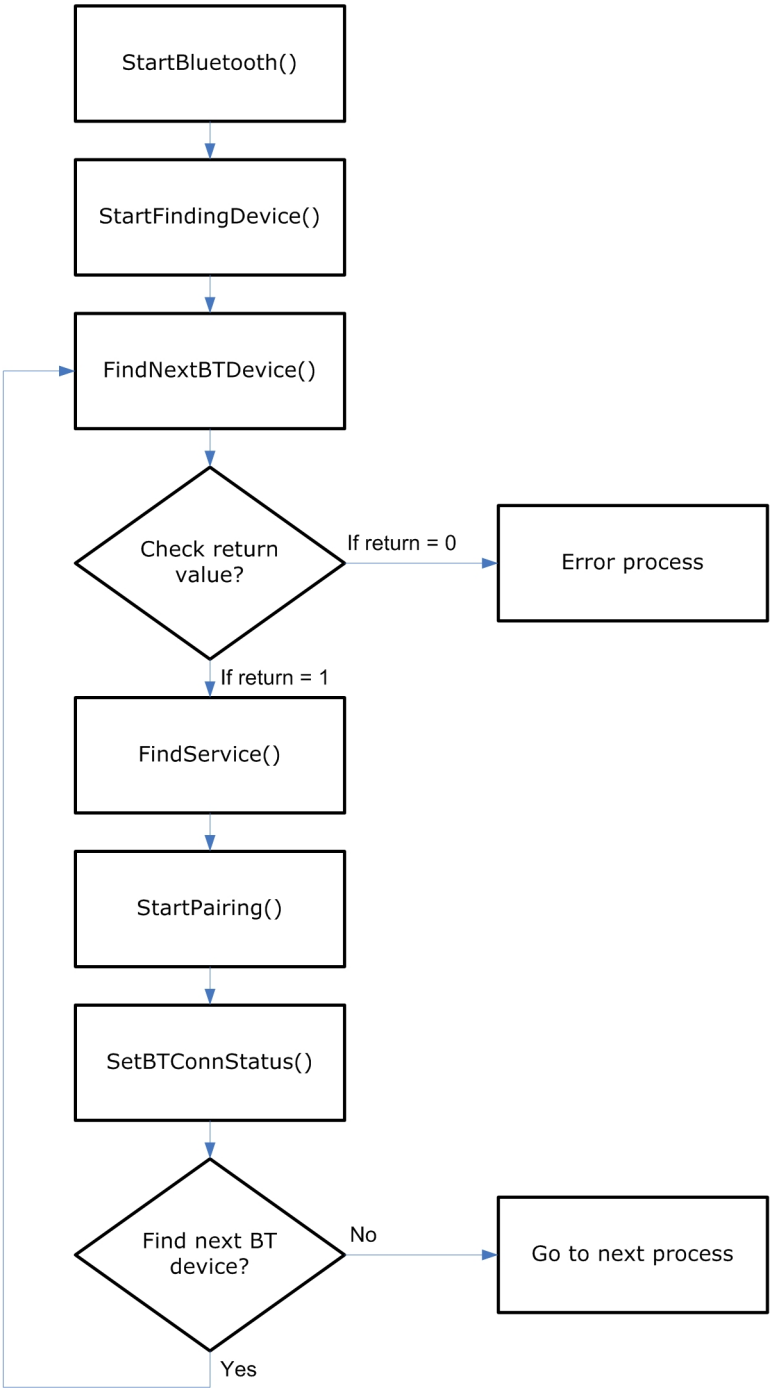
Return Value

If successful, it returns 1.

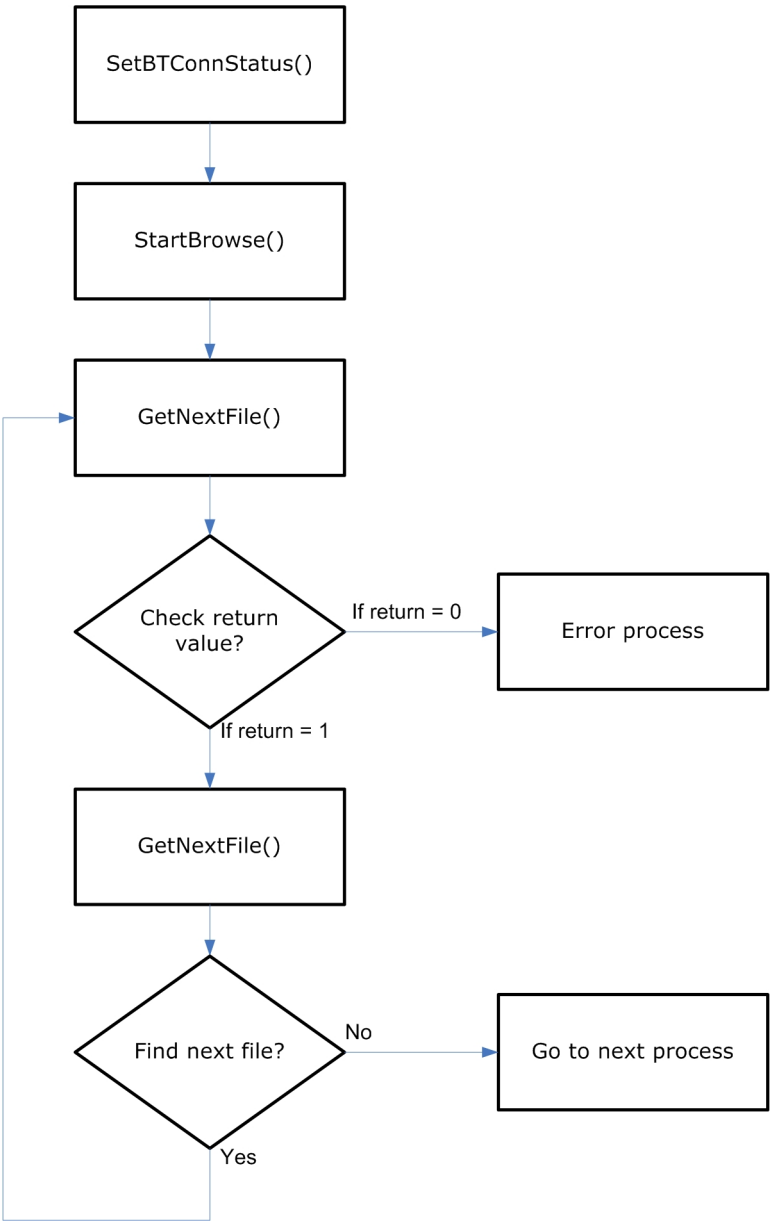
Otherwise, it returns 0.

3.10 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).



3.10.1 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

3.10.2 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

3.10.3 PAIR DEVICES

StartPairing

Purpose	To pair with a spcified 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 spcified 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

3.10.4 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 6 (client)
3	BT_SPP_SERVICE	COM 6 (client/server)
4	BT_OPP_SERVICE	
5	BT_PAN_SERVICE	
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:

21	ERROR_NO_DEVICE_INFO
25	ERROR_NOT_DEVICEINFO_OBJ
31	ERROR_WRONG_ARG

See Also	FindNextBTDevice

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:
See Also	FindNextBTDevice

21	ERROR_NO_DEVICE_INFO
25	ERROR_NOT_DEVICEINFO_OBJ
31	ERROR_WRONG_ARG

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:

21	ERROR_NO_DEVICE_INFO
26	ERROR_NOT_SERVICEINFO_OBJ
31	ERROR_WRONG_ARG

See Also FindService

3.10.5 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 6 (client)
3	BT_SPP_SERVICE	COM 6 (client/server)
4	(Reserved)	
5	BT_PAN_SERVICE	
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 6 (client)
3	BT_SPP_SERVICE	COM 6 (client/server)
4	(Reserved)	
5	BT_PAN_SERVICE	
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

3.10.6 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:

22	ERROR_NO_SERVICE_INFO
26	ERROR_NOT_SERVICEINFO_OBJ
27	ERROR_WRONG_SERVICE_TYPE
29	ERROR_SERVICE_NOT_CONNECTED
32	ERROR_OPERATION_FAIL

See Also	FindService

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

3.10.7 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

3.10.8 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 comPort);`

Example for VB	`Dim b1 As Integer`
	`Dim serviceInfo As Integer`
	`Dim comPort As Integer`
	`b1 = Cipherlab.SystemAPI.Member.FindService(`
	`Member.BT_SPP_SERVICE, deviceInfo, serviceInfo)`
	`b1 = Cipherlab.SystemAPI.Member.GetSppComPort(`
	`serviceInfo, comPort)`

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] A value that specifies the serial port number.

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.SetSppComPort(serviceInfo, comPort);`

Example for VB	`Dim b1 As Integer`
	`Dim serviceInfo As Integer`
	`Dim comPort As Integer`
	`b1 = Cipherlab.SystemAPI.Member.FindService(`
	`Member.BT_SPP_SERVICE, deviceInfo, serviceInfo)`
	`b1 = Cipherlab.SystemAPI.Member.SetSppComPort(`
	`serviceInfo, comPort)`

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, COM 6 is the serial port used for connecting SPP service. Currently, the option is COM 7, which is for external port use. This function must be called before SetBTConnStatus().

3.10.9 CIPHERLAB BT-CONNECT

GetBTService

Purpose	To find out the current state of the CipherLab BT-Connect utility.
Syntax	**int GetBTService (ref int** *onOff***);**
Parameters	*onOff* [out] Pointer to a buffer where the information is stored.

0	Bluetooth power is off and no BT-Connect service
1	Bluetooth power is on and BT-Connect service is available

Example for C#

```
int b1 = 0;
int onOff = 0;
b1 = Cipherlab.SystemAPI.Member.GetBTService(ref onOff);
```

Example for VB

```
Dim b1 As Integer
Dim onOff As Integer = 1
b1 = Cipherlab.SystemAPI.Member.GetBTService(onOff)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

4	ERROR_PARAMETER (Wrong parameter)

Remarks

GetBTService() and SetBTService() are provided for launching the CipherLab BT-Connect service quickly. They cannot be used with other Bluetooth functions.

SetBTService

Purpose: To decide whether to launch the CipherLab BT-Connect utility.

Syntax: **int SetBTService (int** *onOff***);**

Parameters: *onOff*

[in] Integer variable

0	Turn off Bluetooth power and exit BT-Connect service
1	Turn on Bluetooth power and launch BT-Connect service

Example for C#

```
int b1 = 0;
int onOff = 1;
b1 = Cipherlab.SystemAPI.Member.SetBTService(onOff);
```

Example for VB

```
Dim b1 As Integer
Dim onOff As Integer = 1
b1 = Cipherlab.SystemAPI.Member.SetBTService(onOff)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

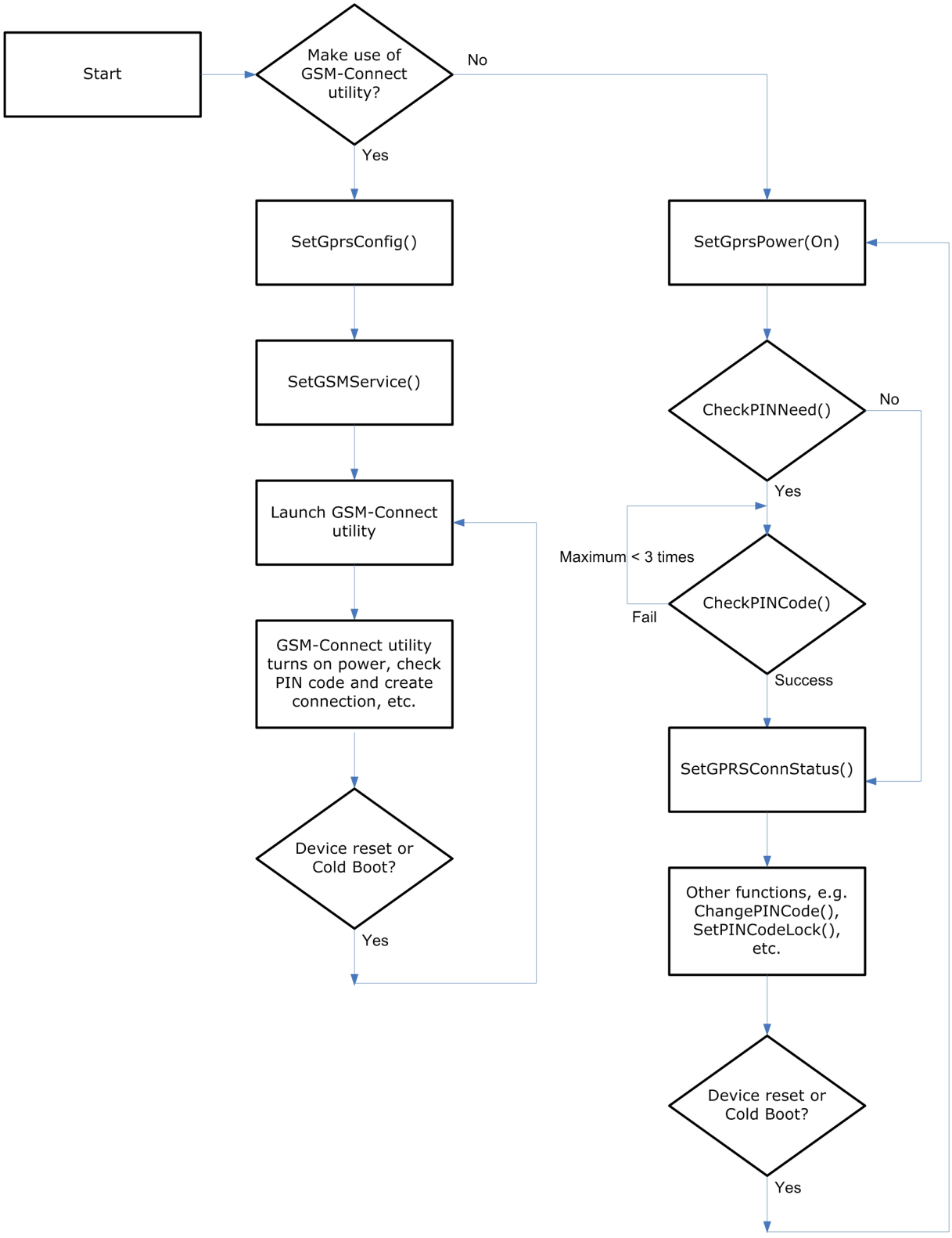
- Call GetErrorCode() to find the error condition encountered:

1	ERROR_NORESOURCE (Service is running)
4	ERROR_PARAMETER (Wrong parameter)
21	ERROR_NO_DEVICE_INFO (No device)

Remarks: GetBTService() and SetBTService() are provided for launching the CipherLab BT-Connect service quickly. They cannot be used with other Bluetooth functions.

3.11 GSM/GPRS

Here is the programming flow for GPRS initialization.



3.11.1 GPRS POWER

GetGPRSPower

Purpose	To find out the current power state of GPRS card.
Syntax	**int GetGPRSPower (ref byte** *onOff***);**
Parameters	*onOff* [out] A byte variable that stores the information.

0	Power to GPRS card is OFF
1	Power to GPRS card is ON

Example for C#

```
int b1 = 0;
byte gPoweron = new byte();
b1 = Cipherlab.SystemAPI.Member.GetGPRSPower(ref gPoweron);
```

Example for VB

```
Dim b1 As Integer
Dim gPoweron As New Byte
b1 = Cipherlab.SystemAPI.Member.GetGPRSPower(gPoweron)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

1	(CreateFile Fail; Device Occupied)
2	Fail to get card handle

Remarks

This function cannot be used to obtain the GPRS power state until the GPRS connection has been disconnected. Please use GetGPRSConnStatus() before calling this function.

See Also

GetGPRSConnStatus

SetGPRSPower

Purpose	To set the power state of GPRS card.
Syntax	**int SetGPRSPower (byte** *onOff***);**
Parameters	*onOff* [in] Byte variable

0	Turn off the power to GPRS card
1	Turn on the power to GPRS card

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetGPRSPower(1);
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetGPRSPower(1)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

1	Fail to enable card power
2	Fail to disable card power

See Also: CheckPINCode, CheckPINNeed, SetGPRSConnStatus

3.11.2 GPRS STATUS

GetGPRSConnStatus

Purpose	To find out the status of the current GPRS connection.
Syntax	**int GetGPRSConnStatus (ref int** *status***);**
Parameters	*status* [out] An integer variable that stores the information.

0	GPRS network disconnected
1	GPRS network connected

Example for C#

```
int b1 = 0;
int status = 0;
b1 = Cipherlab.SystemAPI.Member.GetGPRSConnStatus(ref connStatus);
```

Example for VB

```
Dim b1 As Integer
Dim connStatus As Integer
b1 = Cipherlab.SystemAPI.Member.GetGPRSConnStatus(connStatus)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

33	ERROR_MACHINE_NOT_SUPPORTED

See Also GetGPRSPower

SetGPRSConnStatus

Purpose: To set GPRS connection.

Syntax:

int SetGPRSConnStatus (int *onOff*,
string *entryName*,
string *phoneNumber*,
string *userName*,
string *userPassword*,
string *apName*,
string *callbackNumber*,
string *domainName*);

Parameters:

onOff

[in] Integer variable

0	Disconnect
1	Connect to GPRS network

entryName

[in] A string variable that stores the entry name. If passing "", it uses "GPRS" as a default value.

phoneNumber

[in] A string variable that stores the phone number to dial. This parameter is ignored and should be set to "".

userName

[in] A string variable that stores the user name. If passing "", it uses "" as a default value.

userPassword

[in] A string variable that stores the password. If passing "", it uses "" as a default value.

apName

[in] A string variable that stores the GPRS access point name. If passing "", it uses "INTERNET" as a default value.

callbackNumber

[in] A string variable that stores the phone number to call back. This parameter is ignored and should be set to "".

domainName

[in] A string variable that stores the domain name. If passing "", it uses "" as a default value.

Example for C#

```
int b1 = 0;

b1 = Cipherlab.SystemAPI.Member.SetGPRSConnStatus(1, "GPRS", "", "", "", "", "", "");
```

Example for VB

```
Dim b1 As Integer

b1 = Cipherlab.SystemAPI.Member.SetGPRSConnStatus(1, "GPRS", "", "", "", "", "", "")
```

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

33	ERROR_MACHINE_NOT_SUPPORTED
It may return error codes defined by Microsoft. Please refer to http://msdn.microsoft.com/en-us/library/bb530704(VS.85).aspx.	

Remarks	Please make sure the GPRS power is turned on before calling this function.
See Also	SetGPRSPower

3.11.3 GSM SIGNAL STRENGTH

GetGSMSignalStrength

Purpose	To find out the current GSM signal strength.
Syntax	**int GetGSMSignalStrength (ref uint** *strength***);**
Parameters	*strength* [out] Pointer to a buffer where the information is stored.

0	≤ −113 dBm
1	−111 dBm
2	−109 dBm
...	…
30	− 53 dBm
31	≥ -51 dBm
99	Not detectable or unknown

Example for C#

```
int b1 = 0;
uint strength = 0;
b1 = Cipherlab.SystemAPI.Member.GetGSMSignalStrength(ref strength);
```

Example for VB

```
Dim b1 As Integer
Dim strength As UInteger = 0
b1 = Cipherlab.SystemAPI.Member.GetGSMSignalStrength(strength)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

1	Fail to open GPRS device handle
2	Fail to send command
33	ERROR_MACHINE_NOT_SUPPORTED

See Also: GetGPRSPower

3.11.4 PIN CODE

CheckPINCode

Purpose: To check whether PIN code is matching.

Syntax: **int CheckPINCode (string** *pinCode*,
ref byte *result*);

Parameters: *pinCode*

[in] A string variable that stores the old PIN code.

result

[out] A byte variable that stores the result.

0	PIN code is found mismatching
1	PIN code is found matching

Example for C#

```
int b1 = 0;
string Pin = "0000";
byte Result = 0;
b1 = Cipherlab.SystemAPI.Member.CheckPINCode(Pin, ref Result);
```

Example for VB

```
Dim b1 As Integer
Dim OldPin As String = "0000"
Dim Result As Byte = 0
b1 = Cipherlab.SystemAPI.Member.CheckPINCode(Pin, Result)
```

Return Value: If successful, it returns 1.

Otherwise, it returns 0.

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

1	Fail to open GPRS device handle
2	Fail to send command

CheckPINNeed

Purpose	To check whether PIN code is required.
Syntax	**int CheckPINNeed (ref byte** *state***);**
Parameters	*state* [out] A byte variable that stores the information.

0	SIM card PIN disabled (= PIN code not required)
1	SIM card PIN enabled (= PIN code required)

Example for C#

```
int b1 = 0;
byte Stateflags = 0;
b1 = Cipherlab.SystemAPI.Member.CheckPINNeed(ref Stateflags);
```

Example for VB

```
Dim b1 As Integer
Dim Stateflags As Byte = 0
b1 = Cipherlab.SystemAPI.Member.CheckPINNeed(Stateflags)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

1	Fail to open GPRS device handle
2	Fail to send command

ChangePINCode

Purpose	To change PIN code.
Syntax	**int ChangePinCode (string** *oldPinCode*, **string** *newPinCode*);
Parameters	*oldPinCode* [in] A string variable that stores the old PIN code. *newPinCode* [in] A string variable that stores the new PIN code.

Example for C#	`int b1 = 0;`
	`string OldPin = "0000";`
	`string NewPin = "1234";`
	`b1 = Cipherlab.SystemAPI.Member.ChangePINCode(OldPin, NewPin);`

Example for VB	`Dim b1 As Integer`
	`Dim OldPin As String = "0000"`
	`Dim NewPin As String = "1234"`
	`b1 = Cipherlab.SystemAPI.Member.ChangePINCode(OldPin, NewPin)`

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

1	Fail to get card handle
2	Fail to send command
4	PIN code error

Remarks	SIM card PIN must be enabled first.
See Also	SetPINCodeLock

SetPINCodeLock

Purpose	To decide whether to apply PIN code for security.
Syntax	**int SetPINCodeLock (string** *pinCode***,** **byte** *onOff***);**
Parameters	*pinCode* [in] A string variable that stores the PIN code. *onOff* [in] Byte variable

0	Disable SIM card PIN (= PIN code not required)
1	Enable SIM card PIN (= PIN code required)

Example for C#

```
int b1 = 0;
string pin = "0000"
byte OnOff = 0;
b1 = Cipherlab.SystemAPI.Member.SetPINCodeLock(pin, OnOff);
```

Example for VB

```
Dim b1 As Integer
Dim pin As String = "0000"
Dim OnOff As Byte = 0
b1 = Cipherlab.SystemAPI.Member.SetPINCodeLock(pin, OnOff)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

1	Fail to get card handle
2	Fail to send command
4	PIN code error

3.11.5 CIPHERLAB GSM-CONNECT

SetGPRSConfig	
Purpose	To set connection settings for launching the CipherLab GSM-Connect utility.
Syntax	**int SetGPRSConnStatus (string** *phoneNumber*, **string** *userName*, **string** *userPassword*, **string** *apName*,**);**
Parameters	*phoneNumber* [in] A string variable that stores the phone number to dial. This parameter is ignored and should be set to "". *userName* [in] A string variable that stores the user name. If passing "", it uses "" as a default value. *userPassword* [in] A string variable that stores the password. If passing "", it uses "" as a default value. *apName* [in] A string variable that stores the GPRS access point name. If passing "", it uses "INTERNET" as a default value.

Example for C#	`int b1 = 0;`
	`string phoneNumber = "*99***1#";`
	`string userName = "";`
	`string password = "";`
	`string apName = "INTERNET";`
	`b1 = Cipherlab.SystemAPI.Member.SetGPRSConfig(phoneNumber, username, password, apName);`

Example for VB	`Dim b1 As Integer`
	`Dim phoneNumber As String = "*99***1#"`
	`Dim userName As String = ""`
	`Dim password As String = ""`
	`Dim apName As String = "INTERNET"`
	`b1 = Cipherlab.SystemAPI.Member.SetGPRSConfig(phoneNumber, userName, password, apName)`

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
33	ERROR_MACHINE_NOT_SUPPORTED
It may return error codes defined by Microsoft. Please refer to http://msdn.microsoft.com/en-us/library/bb530704(VS.85).aspx.	

Remarks	After calling this function, call SetGSMService() to launch the CipherLab GSM-Connect utility and turn on the GPRS power.

GetGSMService

Purpose	To find out the current state of the CipherLab GSM-Connect utility.
Syntax	**int GetGSMService (ref int** *onOff***);**
Parameters	*onOff* [out] Pointer to a buffer where the information is stored.

0	GSM/GPRS power is off and no GSM-Connect service
1	GSM/GPRS power is on and GSM-Connect service is available

Example for C#

```
int b1 = 0;
int onOff = 0;
b1 = Cipherlab.SystemAPI.Member.GetGSMService(ref onOff);
```

Example for VB

```
Dim b1 As Integer
Dim onOff As Integer = 1
b1 = Cipherlab.SystemAPI.Member.GetGSMService(onOff)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

4	PIN code error

Remarks

After calling this function, call SetGSMService() to launch the CipherLab GSM-Connect utility and turn on the GPRS power.

SetGSMService

Purpose

To decide whether to launch the CipherLab GSM-Connect utility.

Syntax

int SetGSMService (int *onOff***);**

Parameters

onOff

[in] Integer variable

0	Turn off GSM/GPRS power and exit GSM-Connect service
1	Turn on GSM/GPRS power and launch GSM-Connect service

Example for C#

```
int b1 = 0;
int onOff = 1;
b1 = Cipherlab.SystemAPI.Member.SetGSMService(onOff);
```

Example for VB

```
Dim b1 As Integer
Dim onOff As Integer = 1
b1 = Cipherlab.SystemAPI.Member.SetGSMService(onOff)
```

Return Value

If successful, it returns 1.

Otherwise, it returns 0.

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

1	ERROR_NORESOURCE (Service is running)
4	ERROR_PARAMETER (Wrong parameter)
21	ERROR_NO_DEVICE_INFO (No device)

Remarks

SetGSMService() is provided for launching the CipherLab GSM-Connect service quickly. It cannot be used with other GSM/GPRS functions.

Call SetGPRSConfig() first, and then call this function to launch the utility and turn on the GPRS power.

3.11.6 INFORMATION

GetGPRSManufacturer

Purpose	To find out the manufacturer of the GPRS card.
Syntax	**int GetGPRSManufacturer (ref string** *manuInfo***);**
Parameters	*manuInfo* [out] Pointer to a buffer where the information is stored.

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

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

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

1	Fail to open GPRS device handle
2	Fail to send command

Remarks	Please make sure the GPRS power is turned on before calling this function.

3.12 CAMERA

Here is the programming flow for camera operation.

3.12.1 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

3.12.2 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	800 x 600 pixels
1	1600 x 1200 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)

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: To get the camera flashlight setting.

Syntax: **int GetFlashLight (ref Flash** *flashLight***);**

Parameters: *flashLight*

[out] Flash structure that stores the flashlight setting.

Example for C#:

```
int b1 = 0;
Member.Flash flashlight = new Member.Flash();
b1 = Cipherlab.SystemAPI.Member.GetFlashLight(ref flashlight);
```

Example for VB:

```
Dim flashlight As New Member.Flash
b1 = Cipherlab.SystemAPI.Member.GetFlashLight(flashlight)
```

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: To control the camera flashlight for the preview or still capture mode.

Syntax: **int SetFlashLight (Flash** *flashLight***);**

Parameters: *flashLight*

[in] Flash structure that stores the flashlight setting.

Example for C#:

```
int b1 = 0;
Cipherlab.SystemAPI.Member.Flash flashlight =
new Cipherlab.SystemAPI.Member.Flash();
flashlight.PreviewFlash = 0;
flashlight.CaptureFlash = 1;
b1 = Cipherlab.SystemAPI.Member.SetFlashLight(flashlight);
```

Example for VB:

```
Dim b1 As Integer
Dim flashlight As New Member.Flash
flashlight.PreviewFlash = 0
flashlight.CaptureFlash = 1
b1 = Cipherlab.SystemAPI.Member.SetFlashLight(flashlight)
```

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: EnablePreview, StartStillCapture

3.12.3 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** *posY***);**
Parameters	*posX* Reserved [out] An integer variable that stores the information. *posY* [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 posY);`

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** *posY***);**
Parameters	*posX* Reserved [in] Integer variable *posY* [in] Integer variable
Example for C#	`int posX = 0, posy = 0;` `Cipherlab.SystemAPI.Member.SetPreviewLocation(posX, posY);`

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	Sepia
3	Negative
4	Solarize with unmodified UV
5	Solarize with UV

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)

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

3.12.4 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(ref 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: 1 — ERROR_NORESOURCE (Fail to get resource) 4 — ERROR_PARAMETER (Wrong parameter)
Remarks	This function must be called before StartStillCapture().
See Also	StartStillCapture

StartStillCapture

Purpose	To start with still image capture.
Syntax	**int StartStillCapture (int** *playSound*, **int** *addTimeStamp*);
Parameters	*playSound* [in] Integer variable

0	No notification.
1	Play a sound to notify the image has been captured.

addTimeStamp

[in] Integer variable

0	Do not apply time stamp
1	Apply time stamp

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

3.13 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	57600
Data Bits	8
Parity	None
Stop bits	1

The GPS receiver hosted on the mobile computer uses COM 9 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

3.14 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	Scan	(default: Scan)
4		(Reserved)
5	Send	(default: Send)
6	End	(default: End)

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

SetButtonAssignment

Purpose

To assign a user-definable key to act as another key or serve as a shortcut key for launching a specific program.

Syntax

int SetButtonAssignment (int *buttonID*,
int *keyCode*,
string *buffer*);

Parameters

buttonID

[in] Integer variable

1	Side-Trigger-Left	(default: Scan)
2	Side-Trigger-Right	(default: Scan)
3	Scan	(default: Scan)
4		(Reserved)
5	Send	(default: Send)
6	End	(default: End)

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	F1
10	F2
...	…
20	F12
21	Start Menu
22	Alt
23	OEM_Key1 (0xE9) = Send
24	OEM_Key2 (0xEA) = End
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)

buffer

[in] A string variable that stores user-defined key code or the full file path of the program.

Example for C#

```
int b1 = 0;
b1 = Cipherlab.SystemAPI.Member.SetButtonAssignment(
1, 1, "\\DiskOnChip\\CE_ReaderConfig.exe");
```

Example for VB

```
Dim b1 As Integer
b1 = Cipherlab.SystemAPI.Member.SetButtonAssignment(
1, 1, "\DiskOnChip\CE_ReaderConfig.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

3.15 SIGNATURE CAPTURE

3.15.1 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.

3.15.2 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)
```

3.15.3 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)`

3.15.4 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)
```

3.15.5 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:

1	SIGERROR_UNKNOWN
3	SIGERROR_SIG_WINDOW_NOT_CREATED
4	SIGERROR_FILE_NOT_FOUND
6	SIGERROR_HIDDEN_WINDOW

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.

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.

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

3.15.6 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

3.16 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

IN THIS CHAPTER

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 via **Start | Settings | Control Panel | System –** tap the Device Name tab.

```
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	Device Configuration – 7-digit code. Refer to section 3.2.2 for Device Configuration Code in the 9600 Reference Manual.
string	SerialNum	Serial Number
string	SerialNumReadOnly	Serial number of 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 via **Start | Settings | Control Panel | Display - Backlight tab**.

- 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 – 0~6 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 – 0~6 from dark to bright
uint	backlightontap	When backlight is off in battery mode, it will be automatically turned on when you tap the touch

<table>
<tr><td></td><td></td><td>screen or press a key –<table><tr><td>0</td><td>Disable backlight on tap in battery mode</td></tr><tr><td>1</td><td>Enable backlight on tap in battery mode</td></tr></table></td></tr>
<tr><td>uint</td><td>acbacklightontap</td><td>When backlight is off in AC mode, it will be automatically turned on when you tap the touch screen or press a key –<table><tr><td>0</td><td>Disable backlight on tap in AC mode</td></tr><tr><td>1</td><td>Enable backlight on tap in AC mode</td></tr></table></td></tr>
<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 information about the program to be executed upon system initialization.

```
public struct KeyPadBind

        {

        public string        szClass;

        }
```

Data Type	Member Name	Description
string	szClass	Name of the program that is to be executed.

4.4 WI-FI STRUCTURES

4.4.1 WlanAdptInfo

This structure stores information about wireless networking, which is identical to the settings displayed on the mobile computer via **Start | Settings | Network and Dial-up Connections –** double-tap **WLAN1**.

```
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 rest IP-related information will be set 0.

4.4.2 CF10G_STATUS

This structure stores information about the connection status.

```
typedef 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;
        } CF10G_STATUS;
```

Data Type	Member Name	Description
CARDSTATE	cardState	The association status — 0 CARDSTATE_NOT_INSERTED 1 CARDSTATE_NOT_ASSOCIATED 2 CARDSTATE_ASSOCIATED 3 CARDSTATE_AUTHENTICATED 4 CARDSTATE_FCCTEST

		5 CARDSTATE_NOT_SDC
string	configName	Name of active profile No more than 33 characters. See CONFIG_NAME_SZ defined in Cipherlab.SystemAPI.Member.
string	client_MAC	MAC address of Summit radio
string	client_IP	IP address of Summit radio
string	clientName	Name of Summit radio No more than 33 characters. See CLIENT_NAME_SZ defined in Cipherlab.SystemAPI.Member.
string	AP_MAC	MAC address of the access point
string	AP_IP	IP address of the access point (= Null, if not supported by AP)
string	APName	Name of the access point (= Null, if not supported by AP) No more than 33 characters. See CLIENT_NAME_SZ defined in Cipherlab.SystemAPI.Member.
EAPTYPE	eapType	The EAP type in use — 0 EAP_NONE 1 EAP_LEAP 2 EAP_EAPFAST 3 PEAPMSCHAP 4 PEAPGTC 5 EAPTLS
uint	channel	Channel in use — information on WLAN connection between Summit radio and AP.
int	rssi	Signal strength — information on WLAN connection between Summit radio and AP.
BITRATE	bitRate	Data rate — information on WLAN connection between Summit radio and 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 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
uint	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).
uint	beaconPeriod	Amount of time between access point beacons in kilomicroseconds, where one Kμsec equals 1024 microseconds.
uint	beaconsReceived	Beacons received.

4.4.3 CONFIG_FILE_INFO

This structure stores information about the configuration files and SDK version.

```
typedef struct _CONFIG_FILE_INFO

	{

	public uint		numConfigs;

	public byte		globalConfigPresent;

	public byte		thirdPartyConfigPresent;

	public uint		sdkVersion;

	} CONFIG_FILE_INFO;
```

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	Whether 3rd party configuration is present — 0 No 1 Yes
uint	sdkVersion	SDK version for Summit radio

4.4.4 SDC_ALL

This structure stores information about all the configurations.

```
typedef struct _SDC_ALL
        {
        public uint                numConfigs;
        public SDCConfig           *configs;
        public SDC3rdPartyConfig   *configThirdParty;
        public SDCGlobalConfig     *configGlobal;
        } SDC_ALL;
```

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	Refer to SDC3rdPartyConfig structure.
SDCGlobalConfig	*configGlobal	Refer to SDCGlobalConfig structure.

4.4.5 SDCCONFIG

This structure stores detailed information of local configuration.

```
typedef 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;
        } SDCConfig;
```

Data Type	Member Name	Description
string	configName	Name of profile No more than 33 characters. See CONFIG_NAME_SZ defined in Cipherlab.SystemAPI.Member.
string	SSID	Service Set Identifier (SSID) which Summit radio will connect to. No more than 33 characters. See SSID_SZ defined in Cipherlab.SystemAPI.Member.
string	clientName	Name of Summit radio No more than 33 characters. See CLIENT_NAME_SZ defined in Cipherlab.SystemAPI.Member.
int	txPower	The TX power in use —

		0 — Maximum defined for the current regulatory domain 1 — 1 mW 5 — 5 mW 10 — 10 mW 20 — 20 mW 30 — 30 mW 50 — 50 mW
AUTH	authType	The authentication type in use — 0 — Open 1 — Shared-key 2 — LEAP (Network-EAP)
EAPTYPE	eapType	The EAP type in use — 0 — EAP_NONE 1 — EAP_LEAP 2 — EAP_EAPFAST 3 — PEAPMSCHAP 4 — PEAPGTC 5 — EAPTLS
POWERSAVE	powerSave	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></td><td></td><td><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><tr><td>6</td><td>WEP2_AES</td></tr><tr><td>7</td><td>CCKM_TKIP</td></tr><tr><td>8</td><td>WEP_CKIP</td></tr><tr><td>9</td><td>WEP_AUTO_CKIP</td></tr><tr><td>10</td><td>CCKM_AES</td></tr></table></td></tr>
<tr><td>BITRATE</td><td>bitRate</td><td>Data rate used by Summit radio when interacting with 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><tr><td>12</td><td>BITRATE_6</td></tr><tr><td>18</td><td>BITRATE_9</td></tr><tr><td>22</td><td>BITRATE_11</td></tr><tr><td>24</td><td>BITRATE_12</td></tr><tr><td>36</td><td>BITRATE_18</td></tr><tr><td>48</td><td>BITRATE_24</td></tr><tr><td>72</td><td>BITRATE_36</td></tr></table></td></tr>
<tr><td>RADIOMODE</td><td>radioMode</td><td>The radio mode in use —<table><tr><td>0</td><td>RADIOMODE_B_ONLY</td></tr><tr><td>1</td><td>RADIOMODE_BG</td></tr><tr><td>2</td><td>RADIOMODE_G_ONLY</td></tr><tr><td>3</td><td>RADIOMODE_BG_LRS</td></tr><tr><td>4</td><td>RADIOMODE_A_ONLY</td></tr><tr><td>5</td><td>RADIOMODE_ABG</td></tr><tr><td>6</td><td>RADIOMODE_BGA</td></tr></table></td></tr>
</table>

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

4.4.6 SDC3RDPARTYCONFIG

This structure stores detailed information of the 3rd party configuration.

```
typedef struct _SDC3rdPartyConfig

        {

        public string        clientName;

        POWERSAVE            powerSave;

        public int           txPower;

        BITRATE              bitRate;

        RADIOMODE            radioMode;

        } SDC3rdPartyConfig;
```

Data Type	Member Name	Description
string	clientName	Name of Summit radio No more than 33 characters. See CLIENT_NAME_SZ defined in Cipherlab.SystemAPI.Member.
POWERSAVE	powerSave	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></td><td></td><td><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>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>
<tr><td>BITRATE</td><td>bitRate</td><td>Data rate used by Summit radio when interacting with 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>
<tr><td>12</td><td>BITRATE_6</td></tr>
<tr><td>18</td><td>BITRATE_9</td></tr>
<tr><td>22</td><td>BITRATE_11</td></tr>
<tr><td>24</td><td>BITRATE_12</td></tr>
<tr><td>36</td><td>BITRATE_18</td></tr>
<tr><td>48</td><td>BITRATE_24</td></tr>
<tr><td>72</td><td>BITRATE_36</td></tr>
</table></td></tr>
<tr><td>RADIOMODE</td><td>radioMode</td><td>The radio mode in use —<table>
<tr><td>0</td><td>RADIOMODE_B_ONLY</td></tr>
<tr><td>1</td><td>RADIOMODE_BG</td></tr>
<tr><td>2</td><td>RADIOMODE_G_ONLY</td></tr>
<tr><td>3</td><td>RADIOMODE_BG_LRS</td></tr>
<tr><td>4</td><td>RADIOMODE_A_ONLY</td></tr>
<tr><td>5</td><td>RADIOMODE_ABG</td></tr>
<tr><td>6</td><td>RADIOMODE_BGA</td></tr>
</table></td></tr>
</table>

4.4.7 SDCGLOBALCONFIG

This structure stores detailed information of global configuration.

```
typedef 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;
} SDCGlobalConfig;
```

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 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
PREAMBLE	preamble	(Not implemented)
GSHORTSLOT	g_shortslot	(Not implemented)
BT_COEXIST	BTcoexist	(Not implemented)
PING_PAYLOAD	pingPayload	The amount of data to be transmitted on a pin — 32 — 32 bytes 64 — 64 bytes 128 — 128 bytes 256 — 256 bytes 512 — 512 bytes 1024 — 1024 bytes
uint	pingTimeout	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 Cipherlab.SystemAPI.Member.
uint	pingDelay	The amount of time that elapses between successive ping requests — ▸ 0 ~ 7200000 (milliseconds) See PDELAY_LOW and PDELAY_HIGH defined in Cipherlab.SystemAPI.Member.
uint	radioState	Whether to enable radio or not — 0 — Disable

		1 — Enable
uint	displayPasswords	(Not implemented)
uint	adminOverride	(Not implemented)
uint	txMax	(Not implemented)
FCC_TEST	FCCtest	(Not implemented)
uint	testChannel	(Not implemented)
BITRATE	testRate	(Not implemented)
int	testPower	(Not implemented)
uint	regDomain	Check the regulatory domain in use — 0 — FCC (Americas) 1 — ETSI (Europe) 2 — TELEC (Japan) 3 — Worldwide
uint	ledUsed	(Not implemented)
uint	txTestTimeout	(Not implemented)
uint	WMEenabled	Whether to allow the use of Wi-Fi Multimedia Extensions (WME) or not — 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 (type 1) 1 SBR (type 2)
uint[]	Reserved1	Reserved

4.4.8 BSSIDINFO

```
public struct BSSIDInfo
        {
        public byte[]       macAddress;
        public string       ssid;
        public int          privacy;
        public int          rssi;
        public int          networkTypeInUse;
        public int          configFreq;
        public int          infrastructureMode;
        public byte[]       supportedRates;
        }
```

Data Type	Member Name	Description
byte[]	macAddress	MAC address of each access point that broadcasts its SSID.
string	ssid	Service Set Identifier (SSID) of each access point which Summit radio will connect to.
int	privacy	Specifies a WEP/WPA/WPA2 encryption requirement — 0 Privacy disabled 1 Privacy enabled
int	rssi	The received signal strength indication (RSSI), in dBm. Typical values range from -10 through -200 dBm.
int	networkTypeInUse	The network type as defined in the NDIS_802_11_NETWORK_TYPE enumeration. For access points or IBSS nodes that are IEEE 802.11g-capable, the driver must set **NetworkTypeInUse** to **Ndis802_11OFDM24**. This will also imply that the access point or IBSS node supports **Ndis802_11DS**. 0 Ndis802_11FH 1 Ndis802_11DS 2 Ndis802_11OFDM5 3 Ndis802_11OFDM24 4 Ndis802_11NetworkTypeMax

<table>
<tr><td>int</td><td>configFreq</td><td>The frequency, in kHz, of the selected channel.

For a set operation, configFreq specifies the frequency for ad hoc mode. The driver must ignore a setting of configFreq for infrastructure mode.

For a query, DSConfig contains the current radio frequency.

For 2.4-GHz DSSS, 2.4-GHz OFDM, and 5-GHz OFDM radios, the miniport driver must return the current frequency as defined in the 802.11a specification.

The valid frequency ranges for configFreq are from 2,412,000 through 2,484,000 for 2.4-GHz radios and from 5,000,000 through 6,000,000 for 5-GHz radios.

For example, the valid frequency subset in the United States for 5-GHz radios is from 5,180,000 through 5,240,000; from 5,260,000 through 5,320,000; and from 5,745,000 through 5,805,000.</td></tr>
<tr><td>int</td><td>infrastructureMode</td><td>The network mode as defined in the NDIS_802_11_NETWORK_INFRASTRUCTURE enumeration.
<table><tr><td>0</td><td>Ndis802_11IBSS</td></tr><tr><td>1</td><td>Ndis802_11Infrastructure</td></tr><tr><td>2</td><td>Ndis802_11AutoUnknown</td></tr><tr><td>3</td><td>Ndis802_11InfrastructureMax</td></tr></table></td></tr>
<tr><td>byte[]</td><td>supportedRates</td><td>This is an array containing 16 bytes. Each byte contains a data rate in units of 0.5 Mbps. Set unused entries, if any, at the end of the array to 0.</td></tr>
</table>

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	File format of the captured image – 0 JPEG 1 BMP
string	lpPathName	The directory under which the captured image is stored.
int	PathSize	Size of the path name.
string	lpFileName	File name of the saved image.
int	FileSize	Size of the file name.

4.6.2 Flash

```
public struct Flash
	{
	public int		PreviewFlash;
	public int		CaptureFlash;
	}
```

Data Type	Member Name	Description
int	PreviewFlash	Whether to turn on the flashlight (for example, when previweing an image) – 0 – Turn off the flashlight 1 – Turn on the flashlight
int	CaptureFlash	Whether to turn on the flashlight when capturing an image – 0 – Turn off the flashlight 1 – Turn on the flashlight 2 – Auto flash

Appendix I

SCAN ENGINE SETTINGS

The 9600 Series Mobile Computers provide flexible choices on the readers (or scan engines):

- 1D CCD scan engine
- 1D Laser scan engine
- 2D scan engine
- RFID reader — ID_MOD_MP_RFID

Options of different reader combination are allowed, such as 1D+RFID and 2D+RFID. For each combination, both readers can be initialized and ready for scanning at the same time (dual mode operation). For example, if you press the **SCAN** button while running your application on the mobile computer, it will read a barcode in position or an RFID tag in proximity depending on which one comes first.

Note: The mobile computer allows the co-existence of one integrated scan engine and the RFID reader. You cannot have 1D+2D scan engines installed on the mobile computer because they are both barcode readers!

IN THIS CHAPTER

SYMBOLOGIES SUPPORTED

Varying by the scan engine installed, the supported symbologies or tag types are listed below. For details on configuring associated settings, please refer to each Appendix separately.

		CCD/Laser	2D
Codabar		✓	✓
Code 11		×	✓
Code 93		✓	✓
Composite Code		×	✓
MSI		✓	✓
Plessey		✓	×
Postal Codes		×	✓
Telepen		✓	×
Code 128	Code 128	✓	✓
	GS1-128 (EAN-128)	✓	✓
	ISBT 128	✓	✓
Code 2 of 5	Industrial 25 (Discrete 25)	✓	✓
	Interleaved 25	✓	✓
	Matrix 25	✓	✓
	Chinese 25	×	✓
Code 3 of 9	Code 39	✓	✓
	Trioptic Code 39	×	✓
	Italian Pharmacode (Code 32)	✓	✓
	French Pharmacode	✓	×
EAN/UPC	EAN-8	✓	✓
	EAN-13	✓	✓
	Bookland EAN (ISBN)	✓	✓
	UPC-E0	✓	✓
	UPC-E1	×	✓
	UPC-A	✓	✓
GS1 DataBar (RSS)	GS1 DataBar Omnidirectional (RSS-14)	✓	✓
	GS1 DataBar Limited (RSS Limited)	✓	✓
	GS1 DataBar Expanded (RSS Expanded)	✓	✓
2D Symbologies	PDF417	×	✓
	MicroPDF417	×	✓
	Data Matrix	×	✓

	Maxicode	×	✓
	QR Code	×	✓
	MicroQR	×	✓
	Aztec	×	✓

RFID TAGS SUPPORTED

The RFID reader supports read/write operations depending on the tags. The supported labels include ISO 15693, ISO 14443A, and ISO 14443B.

Currently, the performance of some tags has been confirmed, and the results are listed below for your reference.

Note: You should study the specifications of RFID tags before use.

ID_MOD_MP_RFID		UID Only	Read Page	Write Page
ISO 14443A	Mifare Standard 1K	✓	✓	✓
	Mifare Standard 4K	✓	✓	✓
	Mifare Ultralight	✓	✓	✓
	Mifare DESFire	✓	---	---
	Mifare S50	✓	✓	✓
	SLE44R35	✓	✓	✓
	SLE66R35	✓	✓	✓
ISO 14443B	SR176	✓	✓	✓
ISO 15693	ICODE SLI	✓	✓	✓
	SRF55V02P	✓	✓	✓
	SRF55V02S	✓	✓	✓
	SRF55V10P	✓	✓	✓
	TI Tag-it HF-I	✓	✓	✓
	ST LRI512	✓	✓	✓

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

T

U

W