Opticon H25

System Controls SDK Framework

H25 System Controls SDK Framework Documentation

Version: 2013.01.24_01
Edition: Standard (USA)
Date: January 1, 2013

Opticon, Inc.
2220 Lind Avenue SW Suite 100
Renton, WA 98057

Toll Free: (800) 636.0090
Local: (425) 651.2120
Fax: (425) 454.0865
Email: Sales@OpticonUSA.com
Email: Support@OpticonUSA.com

www.OpticonUSA.com

Overview

The System Controls SDK Framework provides the control functions of SCC hand-held terminal device.

Requirements

Header: syslib.h
DLL: syslib.dll
Link Library: syslib.lib

Information

SysGetModelName

This function retrieves the null-terminated string of model name.

```
BOOL SysGetModelName (LPCTSTR lpszModelName)
```

Parameter:

`lpszModelName`
[out] Pointer to a null-terminated string that specifies the model name.

Return values:

TRUE indicates success
FALSE indicates failure

SysGetFirmwareVersion

This function retrieves the null-terminated string of firmware version.

```
BOOL SysGetFirmwareVersion (LPCTSTR lpszFWVersion)
```

Parameter:

`lpszFWVersion`
[out] Pointer to a null-terminated string that specifies the firmware version.

Return values:

TRUE indicates success
FALSE indicates failure

SysGetSoftwareVersion

This function retrieves the null-terminated string of software version.

```
BOOL SysGetSoftwareVersion (LPCTSTR lpszSWVersion)
```

Parameter:

`lpszSWVersion`
[out] Pointer to a null-terminated string that specifies the software version.

Return values:

TRUE indicates success
FALSE indicates failure

SysGetSerialNumber

This function retrieves the null-terminated string of serial number.

```
BOOL SysGetSerialNumber (LPCTSTR lpszSerialNumber)
```

Parameter:

`lpszSerialNumber`
[out] Long pointer to a null-terminated string that specifies the serial number.

Return values:

TRUE indicates success
FALSE indicates failure

SysGetConfigNumber

This function retrieves the null-terminated string of configuration number.

```
BOOL SysGetConfigNumber (LPCTSTR lpszConfigNumber)
```

Parameter:

`lpszConfigNumber`
[out] Long pointer to a null-terminated string that specifies the configuration number.

Return values:

TRUE indicates success
FALSE indicates failure

SysGetLibraryVersion

This function retrieves the version of System Controls SDK Framework library.

```
BOOL SysGetLibraryVersion (LPCTSTR lpLibraryVersion)
```

Parameter:

`lpszSerialNumber`
[out] Long Pointer to a null-terminated string that specifies the version of System Controls SDK Framework library.

Return values:

TRUE indicates success
FALSE indicates failure

SysGetMemorySize

This function retrieves the size of system memories.

```
BOOL SysGetMemorySize (PDWORD lpROMSize,PDWORD lpRAMSize)
```

Parameter:

`lpROMSize`
[out] Long pointer to the size dedicated to the store memory.

`lpRAMSize`
[out] Long pointer to size dedicated to system memory.

Return values:

TRUE indicates success
FALSE indicates failure

Keypad Controls

SysGetKBDInputMode

This function retrieves the input mode of keypad.

```
BOOL SysGetKBDInputMode (PBYTE lpdwInputMode)
```

Parameter:

`lpdwInputMode`
[out] Pointer to the variable that specifies the input mode of keypad.

The following table shows the possible values.

Value	Description
NUMERICAL_KEY_MODE	Numerical character (1;2;3.....)
ALPHA_KEY_MODE	Alphabetical character (a;b;c...)
FUNC_KEY_MODE	Function keys

Return values:

TRUE indicates success
FALSE indicates failure

SysSetKBDInputMode
This function sets the input mode of keypad.

```
BOOL SysSetKBDInputMode (BYTE dwInputMode)
```

Parameter:

`dwInputMode`
[in] The variable specifies the input mode of keypad.

The following table shows the possible values.

Value	Description
NUMERICAL_KEY_MODE	Numerical character (1;2;3.....)
ALPHA_KEY_MODE	Alphabetical character (a;b;c...)
FUNC_KEY_MODE	Function keys

Return values:

TRUE indicates success
FALSE indicates failure

SysGetKeypadState
This function retrieves the state (Lock / Unlock) of key.

```
BOOL SysGetKeypadState (DWORD dwVKCode, PBOOL
lpdwLockState)
```

Parameter:

`dwVKCode`
[in] Specifies the virtual-key code to indicate the key

`lpdwLockState`
[out] Pointer to the Boolean value that specifies the state (Lock / Unlock) of key.

Return values:

TRUE indicates success
FALSE indicates failure

SysSetKeypadState
This function sets the state (Lock / Unlock) of key.

```
BOOL SysSetKeypadState(DWORD dwVKCode, BOOL dwLockState)
```

Parameter:

`dwVKCode`
[in] Specifies the virtual-key code to indicate the key

`dwLockState`
[in] The Boolean value specifies whether to lock or unlock the key.
Set it to TRUE to lock the key or FALSE to unlock it.

Return values:

TRUE indicates success
FALSE indicates failure

Function Key Controls

SysGetFxKeyPrograms
This function retrieves the settings of function keys (F1 ~ F4).

```
BOOL SysGetFxKeyPrograms (
        DWORD dwVKCode,
        LPCTSTR lpFileName,
        LPCTSTR lpParameters
        )
```

Parameter:

`dwVKCode`
[in] Specifies the virtual-key code to indicate the function key (F1 ~ F8).

`lpFileName`
[in] Long pointer to a null-terminated string that specifies the absolute name of the file to open.

`lpParameters`
[in] Long pointer to a null-terminated string that contains the application parameters.

Return values:

TRUE indicates success
FALSE indicates failure

SysSetFxKeyPrograms
This function sets the setting of function keys (F1 ~ F4).

```
BOOL SysSetFxKeyPrograms (
        DWORD dwVKCode,
        LPCTSTR lpFileName,
        LPCTSTR lpParameters
        )
```

Parameter:

`dwVKCode`
[in] Specifies the virtual-key code to indicate the function key (F1 ~ F8).

`lpFileName`
[in] Long pointer to a null-terminated string that specifies the absolute name of the file to open.

`lpParameters`
[in] Long pointer to a null-terminated string that contains the application parameters.

Return values:

TRUE indicates success
FALSE indicates failure

Backlight Controls

SysGetScreenBrightness
This function retrieves the level of LCD backlight brightness.

```
BOOL SysGetScreenBrightness (PBYTE lpdwLevel)
```

Parameter:

`lpdwLevel`
[out] Pointer to the variable that specifies the level of LCD backlight brightness..
The range of variable is from 0 to 100.

Return values:

TRUE indicates success
FALSE indicates failure

SysSetBacklight

This function sets the brightness level of LCD backlight.

```
BOOL SysSetScreenBrightness (BYTE dwLevel)
```

Parameter:

`dwLevel`
[out] Specifies the level of LCD backlight brightness..
The range of variable is from 0 to 100.

Return values:

TRUE indicates success
FALSE indicates failure

SysGetBrightnessTimer

This function retrieves the value of timer that specifies the time to turn off the LCD backlight.

```
BOOL SysGetBrightnessTimer (PDWORD lpdwTimeCount);
```

Parameter:

`lpdwTimeCount`
[out] Pointer to the variable that specifies the time-out value of LCD backlight.

The following table shows the possible values.

Value	Description
BKL_TIMEOUT_NEVER	Never turn off LCD backlight
BKL_TIMEOUT_10_SEC	Turn off the LCD backlight after 10 seconds
BKL_TIMEOUT_20_SEC	Turn off the LCD backlight after 20 seconds
BKL_TIMEOUT_30_SEC	Turn off the LCD backlight after 30 seconds
BKL_TIMEOUT_40_SEC	Turn off the LCD backlight after 40 seconds
BKL_TIMEOUT_1_MIN	Turn off the LCD backlight after 1 minute
BKL_TIMEOUT_3_MIN	Turn off the LCD backlight after 3 minutes
BKL_TIMEOUT_5_MIN	Turn off the LCD backlight after 5 minutes
BKL_TIMEOUT_10_MIN	Turn off the LCD backlight after 10 minutes

Return values:

TRUE indicates success
FALSE indicates failure

SysSetBrightnessTimer

This function sets the timer to turn off the LCD backlight.

```
BOOL SysSetBrightnessTimer(DWORD dwTimeCount)
```

Parameter:

`dwTimeCount`
[in] Specifies the time-out value of LCD backlight.

The following table shows the possible values.

Value	Description
BKL_TIMEOUT_NEVER	Never turn off LCD backlight
BKL_TIMEOUT_10_SEC	Turn off the LCD backlight after 10 seconds
BKL_TIMEOUT_20_SEC	Turn off the LCD backlight after 20 seconds
BKL_TIMEOUT_30_SEC	Turn off the LCD backlight after 30 seconds
BKL_TIMEOUT_40_SEC	Turn off the LCD backlight after 40 seconds
BKL_TIMEOUT_1_MIN	Turn off the LCD backlight after 1 minute
BKL_TIMEOUT_3_MIN	Turn off the LCD backlight after 3 minutes
BKL_TIMEOUT_5_MIN	Turn off the LCD backlight after 5 minutes
BKL_TIMEOUT_10_MIN	Turn off the LCD backlight after 10 minutes

Return values:

TRUE indicates success
FALSE indicates failure

SysGetKeypadBacklight
This function retrieves the mode of keypad backlight.

```
BOOL SysGetKeypadBacklight (PBYTE lpdwMode)
```

Parameter:
`lpdwMode`
[out] Pointer to the variable that specifies the mode of keypad backlight.

The following table shows the possible values.

Value	Description
KEYPAD_BL_OFF	Disable keypad backlight (always off)
KEYPAD_BL_ENABLE_TIMER	Enable the timer for keypad backlight
KEYPAD_BL_ALWAYS_ON	Keypad backlight is always on

Return values:
TRUE indicates success
FALSE indicates failure

SysSetKeypadBacklight
This function sets the mode of keypad backlight.

```
BOOL SysSetKeypadBacklight (BYTE dwMode)
```

Parameter:
`dwMode`
[in] Specifies the mode of keypad backlight.

The following table shows the possible values.

Value	Description
KEYPAD_BL_OFF	Disable keypad backlight (always off)
KEYPAD_BL_ENABLE_TIMER	Enable the timer for keypad backlight
KEYPAD_BL_ALWAYS_ON	Keypad backlight is always on

Return values:
TRUE indicates success
FALSE indicates failure

Wireless LAN Controls

SysGetWLANPower
This function retrieves the power of wireless LAN.

```
BOOL SysGetWLANPower (LPDWORD dwPowerStatus)
```

Parameter:
`lpdwPowerStatus`
[out] Pointer to the variable that specifies the power of WLAN.
If this parameter is 1, the WLAN is enabled.
If the parameter is 0, the WLAN is disabled.

Return values:
TRUE indicates success
FALSE indicates failure

SysSetWLANPower
This function sets the power of wireless LAN.

```
BOOL SysSetWLANPower (DWORD dwPowerStatus)
```

Parameter:
`dwPowerStatus`
[in] the variable that specifies whether to enable or disable the WLAN
If this parameter is 1, the WLAN is enabled.
If the parameter is 0, the WLAN is disabled.

Return values:
TRUE indicates success
FALSE indicates failure

SysGetWLANPHYAddress
This function retrieves the address of wireless LAN.

```
BOOL SysGetWLANPHYAddress (LPTSTR lpszPhyAddr)
```

Parameter:
`lpszPhyAddr`
[out] Pointer to a null-terminated string that specifies the address of wireless LAN.

Return values:
TRUE indicates success
FALSE indicates failure

Bluetooth Controls

SysGetBluetoothPower
This function retrieves the power of Bluetooth.

```
BOOL SysGetBluetoothPower (LPDWORD lpdwPowerStatus);
```

Parameter:
`lpdwPowerStatus`
[out] Pointer to the variable that specifies the power of Bluetooth.
If this parameter is 1, the Bluetooth is enabled.
If the parameter is 0, the Bluetooth is disabled.

Return values:
TRUE indicates success
FALSE indicates failure

SysSetBluetoothPower
This function sets the power of Bluetooth.

```
BOOL SysSetBluetoothPower (DWORD dwPowerStatus)
```

Parameter:
`dwPowerStatus`
[in] the variable that specifies whether to enable or disable the Bluetooth.
If this parameter is 1, the Bluetooth is enabled.
If the parameter is 0, the Bluetooth is disabled.

Return values:
TRUE indicates success
FALSE indicates failure

SysGetBluetoothPHYAddress
This function retrieves the address of Bluetooth.

```
BOOL SysGetBluetoothPHYAddress (LPTSTR lpszPhyAddr)
```

Parameter:
`lpszPhyAddr`
[out] Pointer to a null-terminated string that specifies the address of Bluetooth.

Return values:
TRUE indicates success
FALSE indicates failure

GPRS Controls

SysGetGPRSPower
This function retrieves the power of GPRS.

```
BOOL SysGetGPRSPower (LPDWORD lpdwPowerStatus);
```

Parameter:
`lpdwPowerStatus`
[out] Pointer to the variable that specifies the power of GPRS.
If this parameter is 1, the GPRS is enabled.
If the parameter is 0, the GPRS is disabled.

Return values:
TRUE indicates success
FALSE indicates failure

SysSetGPRSPower
This function sets the power of GPRS.

```
BOOL SysSetGPRSPower (DWORD dwPowerStatus)
```

Parameter:
`dwPowerStatus`
[in] the variable that specifies whether to enable or disable the GPRS.
If this parameter is 1, the GPRS is enabled.
If the parameter is 0, the GPRS is disabled.

Return values:
TRUE indicates success
FALSE indicates failure

GPS Controls

SysGetGPSPower
This function retrieves the power state of GPS.

```
BOOL SysGetGPSPower (LPDWORD lpdwPowerStatus)
```

Parameter:
`lpdwPowerStatus`
[out] Pointer to the variable that specifies the power of GPS.
If this parameter is 1, the GPS is enabled.
If the parameter is 0, the GPS is disabled.

Return values:
TRUE indicates success
FALSE indicates failure

SysSetGPSPower
This function sets the power state of GPS.

```
BOOL SysSetGPSPower (DWORD dwPowerStatus)
```

Parameter:
`dwPowerStatus`
[in] the variable that specifies whether to enable or disable the GPS.
If this parameter is 1, the GPS is enabled.
If the parameter is 0, the GPS is disabled.

Return values:
TRUE indicates success
FALSE indicates failure

System Controls

SysGetACPowerScheme
This function retrieves the settings of power schemes for AC power.

```
BOOL SysGetACPowerScheme(
        PDWORD lpUerIdleTime,
        PDWORD lpSystemIdleTime,
        PDWORD lpSuspendTime
        )
```

Parameter:
`lpUerIdleTime`
[out] Pointer to the variable that specifies the timeout setting of user idle for AC power.

`lpSystemIdleTime`
[out] Pointer to the variable that specifies the timeout setting of system idle for AC power.

`lpSuspendTime`
[out] Pointer to the variable that specifies the timeout setting of suspend time for AC power.

Return values:
TRUE indicates success
FALSE indicates failure

SysSetACPowerScheme
This function sets the settings of power schemes for AC power.

```
BOOL SysSetACPowerScheme(
        DWORD dwUerIdleTime,
        DWORD dwSystemIdleTime,
        DWORD dwSuspendTime
        )
```

Parameter:
`dwUerIdleTime`
[in] Specifies the timeout setting of user idle for AC power.

`dwSystemIdleTime`
[in] Specifies the timeout setting of system idle for AC power.

`dwSuspendTime`
[in] Specifies the timeout setting of suspend time for AC power.

Return values:
TRUE indicates success
FALSE indicates failure

SysGetBatteryPowerScheme

This function retrieves the settings of power schemes for main battery.

```
BOOL SysGetBatteryPowerScheme (
        PDWORD lpUerIdleTime,
        PDWORD lpSystemIdleTime,
        PDWORD lpSuspendTime
        )
```

Parameter:

`lpUerIdleTime`
[out] Pointer to the variable that specifies the timeout setting of user idle for main battery.

`lpSystemIdleTime`
[out] Pointer to the variable that specifies the timeout setting of system idle for main battery.

`lpSuspendTime`
[out] Pointer to the variable that specifies the timeout setting of suspend time for main battery.

Return values:

TRUE indicates success
FALSE indicates failure

SysSetBatteryPowerScheme

This function sets the settings of power schemes for main battery.

```
BOOL SysSetBatteryPowerScheme(
        DWORD dwUerIdleTime,
        DWORD dwSystemIdleTime,
        DWORD dwSuspendTime
        )
```

Parameter:

`dwUerIdleTime`
[in] The variable specifies the timeout setting of user idle for main battery.

`dwSystemIdleTime`
[in] The variable specifies the timeout setting of system idle for main battery.

`dwSuspendTime`
[in] The variable specifies the timeout setting of suspend time for main battery.

Return values:

TRUE indicates success
FALSE indicates failure

SysGetACLineState

This function retrieves status of AC power.

```
BOOL SysGetACLineState (PBOOL lpACState)
```

Parameter:

`lpACState`
[out] Pointer to the variable that specifies the status of AC power.
If this parameter is TRUE, AC power is online.
If the parameter is FALSE, AC power is offline.

Return values:

TRUE indicates success
FALSE indicates failure

SysGetMainBatteryState

This function retrieves status of main battery.

```
BOOL SysGetMainBatteryState (PBYTE lpPowerLevel,PBOOL
lpChargeState)
```

Parameter:

`lpPowerLevel`
[out] Pointer to the variable that specifies the level of main battery.

The following table shows the possible values.

Value	Description
SYSTEM_BATTERY_HIGH	High
SYSTEM_BATTERY_LOW	Low
SYSTEM_BATTERY_CRITICAL	Critical
SYSTEM_BATTERY_STATE_UNKNOWN	Unknown

`lpChargeState`
[out] Pointer to the variable that specifies the charging state.
If this parameter is TRUE, battery is charging. If the parameter is FALSE, battery power is discharged.

Return values:

TRUE indicates success
FALSE indicates failure

SysGetBackupBatteryState

This function retrieves status of backup battery.

```
BOOL SysGetBackupBatteryState (
        PBYTE lpPowerLevel,
        PBOOL lpChargeState
        )
```

Parameter:

`lpPowerLevel`
[out] Pointer to the variable that specifies the level of backup battery.

The following table shows the possible values.

Value	Description
SYSTEM_BATTERY_HIGH	High
SYSTEM_BATTERY_LOW	Low
SYSTEM_BATTERY_CRITICAL	Critical
SYSTEM_BATTERY_STATE_UNKNOWN	Unknown

`lpChargeState`
[out] Pointer to the variable that specifies the charging state.
If this parameter is TRUE, battery is charging. If the parameter is FALSE, AC power is discharged.

Return values:

TRUE indicates success
FALSE indicates failure

SysGetSIPState
This function receives the state of software-based input panel.

```
BOOL SysGetSIPState (PBOOL lpSIPState)
```

Parameter:
`lpSIPState`
[out] Pointer to the variable that specifies the state of software-based input panel.
If this parameter is TRUE, the software-based input panel is on, or visible.
If the parameter is FALSE, The software-based input panel is off, or not visible.

Return values:
TRUE indicates success
FALSE indicates failure

SysSetSIPState
This function sets the state of software-based input panel.

```
BOOL SysSetSIPState (BOOL dwSIPState)
```

Parameter:
`dwSIPState`
[in] Specifies the state of software-based input panel.
If this parameter is TRUE, the software-based input panel is on, or visible.
If the parameter is FALSE, The software-based input panel is off, or not visible.

Return values:
TRUE indicates success
FALSE indicates failure

SysStylusCalibration
This function opens the stylus calibration.

```
BOOL SysStylusCalibration (void)
```

Parameter:
None

Return values:
TRUE indicates success
FALSE indicates failure

SysLaunchCPLProgram
This function opens the programs of control panel.

```
BOOL SysLaunchCPLProgram (BYTE dwCPLProgram)
```

Parameter:
`dwCPLProgram`
[in] The variable specifies the programs of control panel.

The following table shows the possible values.

Value	Description
CPL_PC	PC Connection Properties
CPL_Dialing	Dialing Properties
CPL_Keyboard	Keyboard Properties
CPL_Password	Password Properties
CPL_Owner	Owner Properties
CPL_Power	Power Properties
CPL_System	System Properties
CPL_Display	Display Properties
CPL_Mouse	Mouse Properties
CPL_Stylus	Stylus Properties
CPL_Volume_Sounds	Volume & Sounds Properties
CPL_InputPanel	Input Panel Properties
CPL_Remove_Programs	Remove Programs
CPL_Date_Time	Date/Time Properties
CPL_Certificates	Certificates
CPL_Bluetooth	Bluetooth Device Properties
CPL_Net_Connections	Network and Dial-up Connections

Return values:
TRUE indicates success
FALSE indicates failure

SysWarmReset
This function supports a warm boot of the device.

```
BOOL SysWarmReset (void)
```

Return values:
TRUE indicates success
FALSE indicates failure

Bar Code Scanner Controls

BCGetPower
This function retrieves the state of barcode engine's power.

```
BOOL BCGetPower (PBOOL lpPowerState)
```

Parameter:
`lpPowerState`
[out] Pointer to the variable that specifies the state of barcode engine's power.
If this parameter is TRUE, the engine is enabled.
If the parameter is FALSE, the engine is disabled.

Return values:
TRUE indicates success
FALSE indicates failure

BCSetPower
This function sets the power of barcode engine.

```
BOOL BCSetPower (BOOL dwState)
```

Parameter:
`dwState`
[in] Specifies the state of barcode engine's power.
If this parameter is TRUE, the engine is enabled.
If the parameter is FALSE, the engine is disabled.

Return values:
TRUE indicates success
FALSE indicates failure

BCStartScan
This function starts the scan process of barcode engine.

```
BOOL BCStartScan (void)
```

Parameter:
None.

Return values:
TRUE indicates success
FALSE indicates failure

BCStopScan
This function stops the scan process of barcode engine.

```
BOOL BCStopScan (void)
```

Parameter:
None.

Return values:
TRUE indicates success
FALSE indicates failure

BCGetLastNotifyEvent
The WM_BC_NOTIFY is posted to the window to notify the state of scanning process when process is running. This function retrieves the state of scanning process.

```
BOOL ZBCRGetLastNotifyEvent (PDWORD lpNotifyEvent)
```

Parameter:
`lpNotifyEvent`
[out] Pointer to the variable that specifies the state of scan process.

The following table shows the possible values.

Value	Description
BC_NOTIFY_START_SCAN	Start to scan a barcode
BC_NOTIFY_STOP_SCAN	Stop scan
BC_NOTIFY_RECEIVE_BARCODE	Receive barcode
BC_NOTIFY_SCAN_FAILED	Scan is failed

Return values:
TRUE indicates success
FALSE indicates failure

BCGetLastBarcode
This function retrieves the string of decoded bar code.

```
BOOL BCGetLastBarcode (LPTSTR lpszBarcode)
```

Parameter:
`lpszBarcode`
[out] Pointer to a null-terminated string that specifies the bar code.

Return values:
TRUE indicates success
FALSE indicates failure

BCGetOutputMode
This function retrieves the setting of bar code output mode.

```
BOOL BCGetOutputMode (PBYTE lpOutputMode)
```

Parameter:
`lpOutputMode`
[out] Pointer to the variable that specifies the setting of bar code output mode.

The following table shows the possible values.

Value	Description
BC_CLIPBOARD_OUTPUT	Copy the bar code to clipboard and past to window
BC_KEYEVENT_OUTPUT	Simulate the keyboard event for each character
BC_DISABLE_OUTPUT	Only save the bar code to memory

Return values:
TRUE indicates success
FALSE indicates failure

BCSetOutputMode
This function sets the setting of bar code output mode.

```
BOOL BCSetOutputMode (BYTE dwMode)
```

Parameter:
`dwMode`
[in] the variable that specifies the setting of bar code output mode.

The following table shows the possible values.

Value	Description
BC_CLIPBOARD_OUTPUT	Copy the bar code to clipboard and past to window
BC_KEYEVENT_OUTPUT	Simulate the keyboard event for each character
BC_DISABLE_OUTPUT	Only save the bar code to memory

Return values:
TRUE indicates success
FALSE indicates failure

BCGetTerminalChar

This function retrieves the terminal character of decoded bar code.

```
BOOL BCGetTerminalChar (PBYTE lpTermChar)
```

Parameter:

`lpTermChar`
[out] Pointer to the variable that specifies the terminal character of bar code.

The following table shows the possible values.

Value	Description
BC_TERMINAL_CHAR_ENTER	Enter Key
BC_TERMINAL_CHAR_SPACE	Space Key
BC_TERMINAL_CHAR_TAB	TAB Key
BC_TERMINAL_CHAR_NONE	NONE

Return values:

TRUE indicates success
FALSE indicates failure

BCSetTerminalChar

This function sets the terminal character of decoded bar code.

```
BOOL BCSetTerminalChar (BYTE dwTermChar)
```

Parameter:

`dwTermChar`
[in] the variable that specifies the terminal character of decoded bar code.

The following table shows the possible values.

Value	Description
BC_TERMINAL_CHAR_ENTER	Enter Key
BC_TERMINAL_CHAR_SPACE	Space Key
BC_TERMINAL_CHAR_TAB	TAB Key
BC_TERMINAL_CHAR_NONE	NONE

Return values:

TRUE indicates success
FALSE indicates failure

BCGetPrefix

This function retrieves the prefix string of bar code.

```
BOOL BCGetPrefix(LPTSTR lpszPrefix)
```

Parameter:

`lpszPrefix`
[out] Pointer to a null-terminated string that specifies the prefix string of bar code.

Return values:

TRUE indicates success
FALSE indicates failure

BCSetPrefix

This function sets the prefix string of bar code.

```
BOOL BCSetPrefix(LPTSTR lpszPrefix)
```

Parameter:

`lpszPrefix`
[in] Pointer to a null-terminated string that specifies the prefix string of bar code.

Return values:

TRUE indicates success
FALSE indicates failure

BCGet Suffix

This function retrieves the suffix string of bar code.

```
BOOL BCGet Suffix (LPTSTR lpszSuffix)
```

Parameter:

`lpszSuffix`
[out] Pointer to a null-terminated string that specifies the suffix string of bar code.

Return values:

TRUE indicates success
FALSE indicates failure

BCSetSuffix

This function sets the suffixstring of bar code.

```
BOOL BCSetsuffix (LPTSTR lpszSuffix)
```

Parameter:

`lpszSuffix`
[in] Pointer to a null-terminated string that specifies the suffix string of bar code.

Return values:

TRUE indicates success
FALSE indicates failure

BCGetReadMode

This function retrieves the setting of bar code reading mode.

```
BOOL BCGetReadMode (LPDWORD lpReadMode)
```

Parameter:

`lpReadMode`
[out] Pointer to the variable that specifies the setting of bar code reading mode.

The following table shows the possible values.

Value	Description
BC_SINGLE_READ	When a bar code has been decoded, the scan engine will stop reading and output the decoded data. The scan engine must be triggered again to read another bar code.
BC_MULTIPLE_READ	When a bar code has been decoded, the decoded data will be output and the scan engine will keep on reading.
BC_READ_3LABELS_IF_POSSIBLE (Only for 2D Engine)	Read only 3 labels in an image
BC_READ_3LABELS_ONLY (Only for 2D Engine)	Read 3 labels in an image if possible

Return values:

TRUE indicates success
FALSE indicates failure

BCSetReadMode

This function sets the setting of bar code reading mode.

```
BOOL BCSetReadMode (DWORD dwReadMode)
```

Parameter:

`dwReadMode`
[in] the variable that specifies the setting of bar code reading mode.

The following table shows the possible values.

Value	Description
BC_SINGLE_READ	When a bar code has been decoded, the scan engine will stop reading and output the decoded data. The scan engine must be triggered again to read another bar code.
BC_MULTIPLE_READ	When a bar code has been decoded, the decoded data will be output and the scan engine will keep on reading.
BC_READ_3LABELS_IF_POSSIBLE (Only for 2D Engine)	Read only 3 labels in an image
BC_READ_3LABELS_ONLY (Only for 2D Engine)	Read 3 labels in an image if possible

Return values:

TRUE indicates success
FALSE indicates failure

BCGetBuzzerState

This function retrieves the state of the buzzer.

```
BOOL BCGetBuzzerState (PBOOL lpEnableState)
```

Parameter:

`lpEnableState`
[out] Pointer to the variable that specifies the state of the buzzer.
If this parameter is TRUE, the buzzer is enabled.
If the parameter is FALSE, the buzzer is disabled.

Return values:

TRUE indicates success
FALSE indicates failure

BCSetBuzzerState

This function sets the state of the buzzer.

```
BOOL BCSetBuzzerState (BOOL dwEnableState)
```

Parameter:

`dwEnableState`
[in] the variable that specifies the state of the buzzer.
If this parameter is TRUE, enable the buzzer.
If the parameter is FALSE, disable the buzzer.

Return values:

TRUE indicates success
FALSE indicates failure

BCGetVibratorState

This function retrieves the setting of vibrator for bar code decoding process.

```
BOOL BCGetVibratorState (PBYTE lpState)
```

Parameter:

`dwState`
[out] Pointer to the variable that specifies the setting of vibrator for bar code decoding process.

The following table shows the possible values.

Value	Description
BC_DIABLE_VIBRATOR	Disable vibrator
BC_SHORT_VIBRATION	Short vibration after decode a bar code
BC_TWO_SHORT_VIBRATION	Two times vibration after decode a bar code
BC_LONG_VIBRATION	Long vibration after decode a bar code

Return values:

TRUE indicates success
FALSE indicates failure

BCSetVibratorState

This function retrieves the setting of vibrator for bar code decoding process.

```
BOOL BCSetVibratorState (BYTE dwState)
```

Parameter:

dwState

[out] Pointer to the variable that specifies the setting of vibrator for bar code decoding process.

The following table shows the possible values.

Value	Description
BC_DIABLE_VIBRATOR	Disable vibrator
BC_SHORT_VIBRATION	Short vibration after decode a bar code
BC_TWO_SHORT_VIBRATION	Two times vibration after decode a bar code
BC_LONG_VIBRATION	Long vibration after decode a bar code

Return values:

TRUE indicates success
FALSE indicates failure

Functions & Supported Devices

Functions	PHL7000	PHL8000	H25
SysGetModelName	Supported	Supported	Supported
SysGetFirmwareVersion	Supported	Supported	Supported
SysGetSoftwareVersion	Supported	Supported	Supported
SysGetConfigNumber	Supported	Supported	Supported
SysGetSerialNumber	Supported	Supported	Supported
SysGetLibraryVersion	Supported	Supported	Supported
SysGetMemorySize	Supported	Supported	Supported
SysGetKBDInputMode	Supported	Supported	Supported
SysSetKBDInputMode	Supported	Supported	Supported
SysGetKeypadState			Supported
SysSetKeypadState			Supported
SysGetFxKeyPrograms	Supported	Supported	Supported
SysSetFxKeyPrograms	Supported	Supported	Supported
SysGetScreenBrightness	Supported	Supported	Supported
SysSetScreenBrightness	Supported	Supported	Supported
SysGetBrightnessTimer	Supported	Supported	Supported
SysSetBrightnessTimer	Supported	Supported	Supported
SysGetKeypadBacklight	Supported	Supported	Supported
SysSetKeypadBacklight	Supported	Supported	Supported
SysSetWLANPower	Supported	Supported	Supported
SysGetWLANPower	Supported	Supported	Supported
SysGetWLANPHYAddress	Supported	Supported	Supported
SysSetBluetoothPower	Supported	Supported	Supported
SysGetBluetoothPower	Supported	Supported	Supported
SysGetBluetoothPHYAddress	Supported	Supported	Supported
SysGetACPowerScheme	Supported	Supported	Supported
SysSetACPowerScheme	Supported	Supported	Supported
SysGetBatteryPowerScheme	Supported	Supported	Supported
SysSetBatteryPowerScheme	Supported	Supported	Supported
SysGetACLineState	Supported	Supported	Supported
SysGetMainBatteryState	Supported	Supported	Supported
SysGetBackupBatteryState	Supported	Supported	Supported
SysGetSIPState	Supported	Supported	Supported
SysSetSIPState	Supported	Supported	Supported
SysLaunchCPLProgram	Supported	Supported	Supported
SysWarmReset			Supported
BCGetPower	Supported	Supported	Supported
BCSetPower	Supported	Supported	Supported
BCStartScan	Supported	Supported	Supported
BCStopScan	Supported	Supported	Supported
BCGetLastNotifyEvent	Supported	Supported	Supported
BCGetLastBarcode	Supported	Supported	Supported
BCGetOutputMode	Supported	Supported	Supported
BCSetOutputMode	Supported	Supported	Supported
BCGetTerminalChar	Supported	Supported	Supported
BCSetTerminalChar	Supported	Supported	Supported
BCGetPrefix	Supported	Supported	Supported
BCSetPrefix	Supported	Supported	Supported
BCGetSuffix	Supported	Supported	Supported
BCSetSuffix	Supported	Supported	Supported
BCGetReadMode			Supported
BCSetReadMode			Supported
BCGetBuzzerState			Supported
BCSetBuzzerState			Supported
BCGetVibratorState			Supported
BCSetVibratorState			Supported

Load DLL & Call the Function

The programmer can use LoadLibrary to map syslib.dll module and return a handle that can be used in GetProcAddress to get the address of a DLL function.

Example:

```
BOOL GetWIFIPower (DWORD *lpPower)
{
        HMODULE hSDK = LoadLibrary(_T("sysctl.dll"));
        BOOL status=FALSE;
        typedef BOOL (*API_Export)(DWORD * lpPower);
        if(hSDK)
        {
                API_Export Export = (API_Export)GetProcAddress(hSDK, _T("SysGetWLANPower"));
                if(Export){ status=Export(lpPower); }
                FreeLibrary(hSDK);
        }
        return status;
}
```