

# [ESC Windows SDK]

[打印机 ESC 指令开发手册 v1.6]

# 目录

|                                                       |    |
|-------------------------------------------------------|----|
| 1. 手册信息 .....                                         | 5  |
| 2. 操作系统 .....                                         | 5  |
| 3. 备注 .....                                           | 5  |
| 4. 方法 .....                                           | 6  |
| 4.1 PrinterCreator .....                              | 6  |
| 4.2 PrinterCreatorS .....                             | 7  |
| 4.3 PrinterDestroy .....                              | 8  |
| 4.4 PortOpen .....                                    | 9  |
| 4.5 PortClose .....                                   | 11 |
| 4.6 PrinterInitialize .....                           | 12 |
| 4.7 SetTextLineSpace .....                            | 13 |
| 4.8 CancelPrintDataInPageMode .....                   | 14 |
| 4.9 GetPrinterState .....                             | 15 |
| 4.10 SetCodePage .....                                | 16 |
| 4.11 SetInternationalCharacter .....                  | 18 |
| 4.12 CutPaper .....                                   | 20 |
| 4.13 FeedLine .....                                   | 21 |
| 4.14 OpenCashDrawer .....                             | 22 |
| 4.15 PrintText .....                                  | 23 |
| 4.16 PrintTextS .....                                 | 25 |
| 4.17 PrintBarCode .....                               | 26 |
| 4.18 PrintSymbol .....                                | 29 |
| 4.19 PrintTwoQRCode .....                             | 31 |
| 4.20 PrintImage .....                                 | 34 |
| 4.21 PrintBitMapData .....                            | 35 |
| 4.22 DefineNVImageCompatible .....                    | 37 |
| 4.23 PrintNVImageCompatible .....                     | 38 |
| 4.24 DefineDownloadedImageCompatible .....            | 39 |
| 4.25 PrintDownloadedImageCompatible .....             | 40 |
| 4.26 GetFirmwareVersion .....                         | 41 |
| 4.27 SelectPageMode .....                             | 42 |
| 4.28 SelectStandardMode .....                         | 43 |
| 4.29 SelectPrintDirectionInPageMode .....             | 44 |
| 4.30 SetAbsoluteVerticalPrintPositionInPageMode ..... | 45 |
| 4.31 PrintAndReturnStandardMode .....                 | 46 |
| 4.32 SetPrintAreaInPageMode .....                     | 47 |
| 4.33 PrintDataInPageMode .....                        | 48 |
| 4.34 DirectIO .....                                   | 49 |
| 4.35 SetAbsolutePrintPosition .....                   | 51 |
| 4.36 PositionNextLabel .....                          | 52 |
| 4.37 DefineNVImage .....                              | 53 |
| 4.38 PrintNVImage .....                               | 54 |
| 4.39 DefineDownloadedImage .....                      | 55 |
| 4.40 PrintDownloadedImage .....                       | 56 |
| 4.41 DefineBufferedImage .....                        | 57 |
| 4.42 PrintBufferedImage .....                         | 58 |
| 4.43 DeleteAllNVImages .....                          | 59 |
| 4.44 GetCashDrawerState .....                         | 60 |
| 4.45 ClearBuffer .....                                | 61 |
| 4.46 FormatError .....                                | 62 |
| 4.47 SetAlign .....                                   | 63 |
| 4.48 SetTextBold .....                                | 64 |
| 4.49 SetTextFont .....                                | 65 |
| 4.50 SetBuzzer .....                                  | 66 |
| 4.51 SetLog .....                                     | 67 |
| 4.52 SetHorizontalAndVerticalMotionUnits .....        | 68 |

|                          |    |
|--------------------------|----|
| 4.53 DrawLine .....      | 69 |
| 4.54 DrawRectangle ..... | 70 |

## 1. 手册信息

本 SDK 手册提供了 Windows 应用程序开发所需的 dll 文件信息。

我们在不断地努力提高和升级我们所有产品的功能与质量。之后，产品规格和用户手册的内容可能会更改，将不再另行通知。

## 2. 操作系统

Windows 2003/XP/7/8/10

## 3. 备注

- 错误代码返回值大于0时，属于 Windows 系统内部错误，请查阅相关帮助文档。
- 本软件开发包区分 Ansi 字符和 Unicode 字符，请根据您所使用的开发环境选择对应的 dll 文件。

## 4. 方法

### 4.1 PrinterCreator

此函数功能为创建指定机型的打印机对象（在进行任何打印机操作之前必须先创建打印机对象）。

```
int PrinterCreator(  
    void* handle,  
    const TCHAR* model  
);
```

参数：

*void\* handle*  
[in,out] 创建的目标打印机对象。

*const TCHAR\* model*  
[in] 指定目标打印机型号。

返回值：

| 错误代码              | 值  | 描述      |
|-------------------|----|---------|
| E_SUCCESS         | 0  | 正常      |
| E_NO_ENOUGHMEMORY | -2 | 计算机内存不足 |
| E_INVALID_MODEL   | -8 | 机型名称无效  |

## 4.2 PrinterCreatorS

此函数功能与 `PrinterCreator` 相同，即创建指定机型的目标打印机（使用任何功能之前必须创建目标打印机）。

```
void* PrinterCreatorS(  
    const TCHAR* model  
);
```

参数：

```
const TCHAR* model  
    [in] 指定目标打印机型号。
```

返回：

成功: 返回打印机对象的句柄。

失败: 返回无效。

## 4.3 PrinterDestroy

此函数功能为释放已创建指定机型打印机对象的资源（在操作结束后且不再进行打印机操作时必须释放创建的打印机对象）。

```
int PrinterDestroy(  
    void* handle  
);
```

参数：

*void\* handle*  
[in] 需要释放的目标打印机对象的句柄。

返回值：

| 错误代码         | 值  | 描述   |
|--------------|----|------|
| E_SUCCESS    | 0  | 正常   |
| E_BAD_HANDLE | -6 | 句柄无效 |

## 4.4 PortOpen

此函数功能为打开通讯端口，与打印建立连接。连接成功后才能正常使用其它功能。连接失败时，请查看函数返回的错误信息。目前支持 USB、网络通信、串口通信、LPT 口通信。

```
int PortOpen(  
    void* handle,  
    const TCHAR* ioSettings  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*const TCHAR\* ioSettings*

[in] 设置连接目标打印机的通讯端口参数。具体内容查看下表：

配置列表：

| 类别  | 配置                                                | 描述                                                                                                                     | 示例                                                        |
|-----|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------|
| USB | <b>USB</b> , <i>[Position/Model/PortNum]</i>      | USB：连接任一本公司 USB 打印机。<br><b>USB</b> , <i>[Position]</i> ：当同时连接本公司多台打印机时，可以通过 USB 位置信息(Position 参数)来指定连接某一特定 USB 端口的打印机。 | USB<br>USB,Port_#0004.Hub_#0003<br>USB,LPG4<br>USB,USB001 |
| NET | <b>NET</b> , <i>IP Add (IPV4)</i> , <i>[Port]</i> | 指定网络打印机的 IP 地址和端口。如果不指定端口，默认端口是9100。                                                                                   | NET,192.168.0.36<br>NET,192.168.0.36,9100                 |
| COM | <b>COM</b> <i>n</i> , <i>BAUDRATE _rate</i>       | 指定连接的串口端口号和波特率。                                                                                                        | COM5,BAUDRATE_19200                                       |
| LPT | <b>LPT</b> <i>n</i>                               | 指定连接的并口端口号。                                                                                                            | LPT1                                                      |

注：[ ]表示可选参数。

**如何查看 USB 打印机的位置信息 (Position 参数)：**

在“Windows 设备管理器”中，展开“通用串行总线控制器”，选中指定机型的“USB 打印支持”设备，右键菜单中选择“属性”，点击“详细信息”。属性“总线关系”包含机型名称与虚拟 USB 端口号。

\* 如果同时连接本公司多台不同型号的打印机，建议采用“USB, 机型”的方式连接。

**返回值：**

| 错误代码                  | 值    | 描述     |
|-----------------------|------|--------|
| E_SUCCESS             | 0    | 正常     |
| E_INVALID_PARAMETER   | -1   | 无效的参数  |
| E_BAD_HANDLE          | -6   | 句柄无效   |
| E_IO_PORT_OPEN_FAILED | -311 | 端口打开失败 |

## 4.5 PortClose

此函数功能为关闭通讯。当不使用端口通讯时，请关闭端口。

```
int PortClose(  
    void* handle  
);
```

参数：

*void\* handle*  
[in,out] 创建的目标打印机对象。

返回值：

| 错误代码         | 值  | 描述   |
|--------------|----|------|
| E_SUCCESS    | 0  | 正常   |
| E_BAD_HANDLE | -6 | 句柄无效 |

## 4.6 PrinterInitialize

此函数功能为清除打印缓冲数据并重置打印机模式为开机有效模式。

不清除任何宏函数。

不清除脱机回应选择。

不清除用户 NV 存储内容。

不清除 NV 图像（NV 位图）和 NV 用户内存。

本指令不影响维护对应值。

不清除指定的脱机回应。

```
int PrinterInitialize(  
    void* handle  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

返回值：

| 错误代码               | 值    | 描述     |
|--------------------|------|--------|
| E_SUCCESS          | 0    | 正常     |
| E_BAD_HANDLE       | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN | -309 | 端口未打开  |
| E_IO_WRITE_FAILED  | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT | -322 | 写入数据超时 |

## 4.7 SetTextLineSpace

此函数功能为设置行间距为行间距 ×（垂直或水平移动单元）。

选择标准模式时，行间距使用的是垂直移动单元。

选择页模式时，根据打印方向确认使用垂直或水平移动单元。当打印方向为从左至右时，行间距使用的是垂直移动单元；当打印方向为从上至下时，行间距使用的是水平移动单元。

```
int SetTextLineSpace(  
    void* handle,  
    int lineSpace  
);
```

**参数：**

*void\* handle*

[in,out] 创建的目标打印机对象。

*int lineSpace*

[in] 设置字符行间距  $0 \leq \text{行间距} \leq 255$ 。

**返回值：**

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.8 CancelPrintDataInPageMode

此函数功能为在页模式下，清除当前打印区域的所有打印数据。

```
int CancelPrintDataInPageMode(  
    void* handle  
);
```

参数：

*void\* handle*  
[in,out] 创建的目标打印机对象。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.9 GetPrinterState

此函数功能为获取打印机实时状态操作。

```
int GetPrinterState(  
    void* handle,  
    unsigned int* printerStatus  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*unsigned int\* printerStatus*

[in,out] 打印机实时状态,返回多种状态时，数值累加表示，返回状态参照以下表格。

| 错误代码             | 值   | 描述      |
|------------------|-----|---------|
| STS_NORMAL       | 0   | 正常      |
| STS_PAPEREMPTY   | 1   | 缺纸      |
| STS_COVEROPEN    | 2   | 上盖打开    |
| STS_PAPERNEAREND | 4   | 纸将尽     |
| STS_ERROR        | 32  | 获取状态时出错 |
| STS_NOT_OPEN     | 64  | 端口未打开   |
| STS_OFFLINE      | 128 | 打印机离线   |

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |
| E_IO_READ_FAILED     | -331 | 读取数据失败    |
| E_IO_READ_TIMEOUT    | -332 | 读取数据超时    |

## 4.10 SetCodePage

此函数功能为设置字符集。

```
int SetCodePage(  
    void* handle,  
    int characterSet,  
    int type  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int characterSet*

[in] 选择字符集设置。

默认：中文字符集

| 值    | 描述               |
|------|------------------|
| 0    | 默认               |
| 437  | 美国字符集            |
| 720  | 阿拉伯字符集           |
| 737  | 希腊字符集            |
| 775  | 波罗的海字符集          |
| 850  | 多语言字符集           |
| 852  | 拉丁语字符集           |
| 855  | 西里尔语字符集          |
| 857  | 土耳其语字符集          |
| 858  | 欧洲字符集            |
| 860  | 葡萄牙字符集           |
| 862  | 希伯来语字符集          |
| 863  | 加拿大法语字符集         |
| 864  | 阿拉伯语字符集          |
| 865  | 北欧字符集            |
| 866  | 西里尔语字符集          |
| 1250 | 中欧字符集（windows）   |
| 1251 | 西里尔字符集（windows）  |
| 1252 | 西欧字符集（windows）   |
| 1253 | 希腊字符集（windows）   |
| 1254 | 土耳其字符集（windows）  |
| 1255 | 希伯来字符集（windows）  |
| 1256 | 阿拉伯字符集（windows）  |
| 1257 | 波罗的海字符集（windows） |
| 1258 | 越南字符集（windows）   |

*int type*

[in] 选择是否在打印机存储器里保存设置。

0: 不写进存储器，关机时设置将不保存。

1: 写进存储器，关机时设置将会保存。

返回值:

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.11 SetInternationalCharacter

此函数功能为选择国际字符集。

```
int SetInternationalCharacter(  
    void* handle,  
    int characterSet  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int characterSet*

[in] 国际字符集。

默认: U.S.A

| 值  | 描述                 |
|----|--------------------|
| 0  | U.S.A              |
| 1  | France             |
| 2  | Germany            |
| 3  | U.K.               |
| 4  | Denmark I          |
| 5  | Sweden             |
| 6  | Italy              |
| 7  | Spain              |
| 8  | Japan              |
| 9  | Norway             |
| 10 | Denmark II         |
| 11 | Spain II           |
| 12 | Latin America      |
| 13 | Korean             |
| 14 | Slovenia / Croatia |
| 15 | Chinese            |

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.12 CutPaper

此函数功能为走纸至（裁切位置+ 间距 × 垂直移动单元），执行全切（完全切纸）或者半切（留一点不切），然后走纸至打印起始位置。

```
int CutPaper(  
    void* handle,  
    int cutMode,  
    int distance  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int cutMode*

[in] 切纸模式，全切或者半切。

| 切纸模式        | 值 | 描述 |
|-------------|---|----|
| FULL_CUT    | 0 | 全切 |
| PARTIAL_CUT | 1 | 半切 |

*int distance*

[in] 指定切纸范围  $0 \leq \text{间距} \leq 255$ 。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.13 FeedLine

此函数功能为打印打印缓冲区的数据并走纸，页模式下，只有打印位置移动，打印机实际上不执行打印。

```
int FeedLine(  
    void* handle,  
    int lines  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int lines*

[in] 设置走纸行数  $0 \leq \text{行数} \leq 255$ 。

返回值：

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.14 OpenCashDrawer

此函数功能为打开钱箱（打印机必须连接钱箱）。

```
int OpenCashDrawer(  
    void* handle,  
    int pinMode,  
    int onTime,  
    int offTime  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int pinMode*

[in] 选择钱箱连接的引脚。

| 引脚          | 值 | 描述   |
|-------------|---|------|
| CASDRAWER_1 | 0 | 引脚 2 |
| CASDRAWER_2 | 1 | 引脚 5 |

*int onTime*

[in] 设置脉冲开始时间，onTime\*2毫秒。

*int offTime*

[in] 设置脉冲结束时间，offTime\*2毫秒。

备注：当结束时间设置值小于开始时间时，结束时间等于开始时间。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.15 PrintText

此函数功能为打印文本数据。

```
int PrintText(  
    void* handle,  
    const TCHAR* data,  
    int alignment,  
    int attribute,  
    int textSize  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*const TCHAR\* data*,

[in] 所需打印的文本数据。

*int alignment*

[in] 文本对齐方式。

| 对齐方式             | 值 | 描述  |
|------------------|---|-----|
| ALIGNMENT_LEFT   | 0 | 左对齐 |
| ALIGNMENT_CENTER | 1 | 居中  |
| ALIGNMENT_RIGHT  | 2 | 右对齐 |

*int attribute*

[in] 设置文本字体属性，属性设置可以叠加。

| 文本字体属性                   | 值  | 描述       |
|--------------------------|----|----------|
| TEXT_NORMAL_MODE         | 0  | 默认设置字体 A |
| TEXT_FONT_BOLD           | 2  | 设置字体加黑   |
| TEXT_FONT_UNDERLINE_MODE | 4  | 设置字体下划线  |
| TEXT_FONT_REVERSE        | 8  | 设置字体反白   |
| TEXT_FONT_DW_DMODE       | 48 | 设置字体倍高倍宽 |

*int textSize*

[in] 设置字体大小（文本长度超过打印纸范围将不被打印）。

设置字体宽度:

| 字体宽度             | 值   | 描述       |
|------------------|-----|----------|
| TEXT_SIZE_0WIDTH | 0   | 字体宽度 × 1 |
| TEXT_SIZE_1WIDTH | 16  | 字体宽度 × 2 |
| TEXT_SIZE_2WIDTH | 32  | 字体宽度 × 3 |
| TEXT_SIZE_3WIDTH | 48  | 字体宽度 × 4 |
| TEXT_SIZE_4WIDTH | 64  | 字体宽度 × 5 |
| TEXT_SIZE_5WIDTH | 80  | 字体宽度 × 6 |
| TEXT_SIZE_6WIDTH | 96  | 字体宽度 × 7 |
| TEXT_SIZE_7WIDTH | 112 | 字体宽度 × 8 |

设置字体高度:

| 字体高度              | 值 | 描述       |
|-------------------|---|----------|
| TEXT_SIZE_0HEIGHT | 0 | 字体高度 × 1 |
| TEXT_SIZE_1HEIGHT | 1 | 字体高度 × 2 |
| TEXT_SIZE_2HEIGHT | 2 | 字体高度 × 3 |
| TEXT_SIZE_3HEIGHT | 3 | 字体高度 × 4 |
| TEXT_SIZE_4HEIGHT | 4 | 字体高度 × 5 |
| TEXT_SIZE_5HEIGHT | 5 | 字体高度 × 6 |
| TEXT_SIZE_6HEIGHT | 6 | 字体高度 × 7 |
| TEXT_SIZE_7HEIGHT | 7 | 字体高度 × 8 |

返回值:

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.16 PrintTextS

此函数功能为打印文本。

```
int PrintTextS(  
    void* handle,  
    const TCHAR* data  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*const TCHAR\* data*,

[in] 需要打印的文件数据。

返回值：

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.17 PrintBarCode

此函数功能为打印条码。在标准模式下，条码打印位置在新行或者没有数据存在于缓存时才能正常打印。在页模式下，未接受打印条码命令时，条码数据保存于缓存中，不打印条码。

```
int PrintBarCode(  
    void* handle,  
    int bcType,  
    const TCHAR* bcData,  
    int width,  
    int height,  
    int alignment,  
    int hriPosition  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int bcType*

[in] 设置条码类型。

*const TCHAR\* bcData,*  
[in] 条码数据。

| 条码类型                           | 值   | 条码数据长度                               | 数据有效值范围                                                                                           |
|--------------------------------|-----|--------------------------------------|---------------------------------------------------------------------------------------------------|
| BARCODE_UPC_A                  | 65  | $11 \leq n \leq 12$                  | $48 \leq \text{data} \leq 57$                                                                     |
| BARCODE_UPC_E                  | 66  | $11 \leq n \leq 12$                  | $48 \leq \text{data} \leq 57$                                                                     |
| BARCODE_EAN13<br>BARCODE_JAN13 | 67  | $12 \leq n \leq 13$                  | $48 \leq \text{data} \leq 57$                                                                     |
| BARCODE_EAN8<br>BARCODE_JAN8   | 68  | $7 \leq n \leq 8$                    | $48 \leq \text{data} \leq 57$                                                                     |
| BARCODE_CODE39                 | 69  | $1 \leq n \leq 255$                  | $48 \leq \text{data} \leq 57, 65 \leq \text{data} \leq 90,$<br>$\text{data}=32,36,37,43,45,46,47$ |
| BARCODE_ITF                    | 70  | $1 \leq n \leq 255$<br>(even number) | $48 \leq \text{data} \leq 57$                                                                     |
| BARCODE_CODABAR                | 71  | $1 \leq n \leq 255$                  | $48 \leq \text{data} \leq 57, 65 \leq \text{data} \leq 68,$<br>$\text{data}=36,43,45,46,47,58$    |
| BARCODE_CODE93                 | 72  | $1 \leq n \leq 255$                  | $0 \leq \text{data} \leq 127$                                                                     |
| BARCODE_CODE128                | 73  | $2 \leq n \leq 255$                  | $0 \leq \text{data} \leq 127$                                                                     |
| BARCODE_STANDARD<br>_PDF417    | 101 | $2 \leq n \leq 928$                  | $0 \leq \text{data} \leq 255$                                                                     |
| BARCODE_TRUNCATE<br>D_PDF417   | 102 | $2 \leq n \leq 928$                  | $0 \leq \text{data} \leq 255$                                                                     |
| BARCODE_QRCODE1                | 103 | $2 \leq n \leq 928$                  | $0 \leq \text{data} \leq 255$                                                                     |
| BARCODE_QRCODE2                | 104 | $2 \leq n \leq 928$                  | $0 \leq \text{data} \leq 255$                                                                     |

*int width*

[in] 条码宽度有效值范围：2-7，当条码打印宽度超过打印纸可打印范围，条码不打印。此参数对二维码无效。

*int height*

[in] 设置条码打印高度。有效范围：1-255，此参数对二维码无效。

*int alignment*

[in] 设置条码对齐方式。

| 对齐方式             | 值 | 描述  |
|------------------|---|-----|
| ALIGNMENT_LEFT   | 0 | 左对齐 |
| ALIGNMENT_CENTER | 1 | 居中  |
| ALIGNMENT_RIGHT  | 2 | 右对齐 |

*int hriPosition*

[in] 设置条码可见字符位置。

| 位置                | 值 | 描述            |
|-------------------|---|---------------|
| BRACODE_HRI_NONE  | 0 | 不打印可见字符       |
| BRACODE_HRI_ABOVE | 1 | 在条码上方打印可见字符   |
| BRACODE_HRI_BELOW | 2 | 在条码下方打印可见字符   |
| BRACODE_HRI_BOTH  | 3 | 在条码上、下方打印可见字符 |

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.18 PrintSymbol

此函数功能为打印二维码。

```
int PrintSymbol(  
    void* handle,  
    int type,  
    const TCHAR* data,  
    int errLevel,  
    int width,  
    int height,  
    int alignment  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int type*

[in] 二维码类型

| 类型                       | 值   | 描述           |
|--------------------------|-----|--------------|
| BARCODE_STANDARD_PDF417  | 101 | 标准样式 PDF417码 |
| BARCODE_TRUNCATED_PDF417 | 102 | 缩简样式 PDF417码 |
| BARCODE_QRCODE1          | 103 | QR Code 样式1  |
| BARCODE_QRCODE2          | 104 | QR Code 样式2  |

**const TCHAR\* data,**

[in] 2D code data.

| 数据长度                 | 数据值大小                         |
|----------------------|-------------------------------|
| $1 \leq n \leq 7089$ | $0 \leq \text{data} \leq 255$ |

*int errLevel*

[in] 二维码设置错误校正等级。

| 错误校正等级                          | 值  | 代码或容错率 |
|---------------------------------|----|--------|
| PDF417_ERROR_CORRECTION_LEVEL_0 | 48 | 2      |
| PDF417_ERROR_CORRECTION_LEVEL_1 | 49 | 4      |
| PDF417_ERROR_CORRECTION_LEVEL_2 | 50 | 8      |
| PDF417_ERROR_CORRECTION_LEVEL_3 | 51 | 16     |
| PDF417_ERROR_CORRECTION_LEVEL_4 | 52 | 32     |
| PDF417_ERROR_CORRECTION_LEVEL_5 | 53 | 64     |
| PDF417_ERROR_CORRECTION_LEVEL_6 | 54 | 128    |
| PDF417_ERROR_CORRECTION_LEVEL_7 | 55 | 256    |
| PDF417_ERROR_CORRECTION_LEVEL_8 | 56 | 512    |
| QRCODE_ERROR_CORRECTION_LEVEL_L | 48 | 7%     |
| QRCODE_ERROR_CORRECTION_LEVEL_M | 49 | 15%    |
| QRCODE_ERROR_CORRECTION_LEVEL_Q | 50 | 25%    |
| QRCODE_ERROR_CORRECTION_LEVEL_H | 51 | 30%    |

*int width*

[in] 二维码宽度  $0 \leq n \leq 255$ 。

*int height*

[in] 二维码高度  $0 \leq n \leq 255$  (此参数对 QRCode 无效)。

*int alignment*

[in] 二维码对齐方式。

| 对齐方式             | 值 | 描述  |
|------------------|---|-----|
| ALIGNMENT_LEFT   | 0 | 左对齐 |
| ALIGNMENT_CENTER | 1 | 居中  |
| ALIGNMENT_RIGHT  | 2 | 右对齐 |

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

# 4.19 PrintTwoQRCode

此函数功能为在同一区域打印二维码。打印机必须支持页模式。

```
int PrintTwoQRCode(  
    void* handle,  
    TCHAR* data1,  
    int data1Len,  
    int width1,  
    int hAlign1,  
    int vAlign1,  
    TCHAR* data2,  
    int data2Len,  
    int width2,  
    int hAlign2,  
    int vAlign2  
);
```

参数：

*void\* handle*  
[in,out] 创建的目标打印机对象。

*TCHAR\* data1*  
[in] 二维码1数据。

*int data1Len*  
[in] 二维码 1 数据长度。

| 数据长度                 | 数据值大小                  |
|----------------------|------------------------|
| $1 \leq n \leq 7089$ | $0 \leq data \leq 255$ |

*int width1*  
[in] 二维码 1 模块宽度。  $0 \leq n \leq 255$

*int hAlign1*

[in] 二维码 1 水平对齐方式。

[in] 当 hAlign1 值大于 2 时,可以自定义二维码 1 水平位置（不能超过当前页面宽度）。

| 对齐方式             | 值 | 描述   |
|------------------|---|------|
| ALIGNMENT_LEFT   | 0 | 左对齐  |
| ALIGNMENT_CENTER | 1 | 水平居中 |
| ALIGNMENT_RIGHT  | 2 | 右对齐  |

*int vAlign1*

[in] 二维码 1 垂直对齐方式。

[in] 当 vAlign1 值大于 2 时,可以自定义二维码 1 垂直位置（不能超过当前设置的二维码高度,否则无法打印二维码）。

| 对齐方式             | 值 | 描述   |
|------------------|---|------|
| ALIGNMENT_TOP    | 0 | 上对齐  |
| ALIGNMENT_CENTER | 1 | 垂直居中 |
| ALIGNMENT_BOTTOM | 2 | 下对齐  |

*TCHAR\* data2*

[in] 二维码 2 数据。

*int data2Len*

[in] 二维码 2 数据长度。

| 数据长度                 | 数据值大小                         |
|----------------------|-------------------------------|
| $1 \leq n \leq 7089$ | $0 \leq \text{data} \leq 255$ |

*int width2*

[in] 二维码 2 模块宽度。  $0 \leq n \leq 255$

*int hAlign2*

[in] 二维码 2 水平对齐方式。

[in] 当 hAlign2 值大于 2 时,可以自定义二维码 2 水平位置（不能超过当前页面宽度）。

| 对齐方式             | 值 | 描述   |
|------------------|---|------|
| ALIGNMENT_LEFT   | 0 | 左对齐  |
| ALIGNMENT_CENTER | 1 | 水平居中 |
| ALIGNMENT_RIGHT  | 2 | 右对齐  |

*int vAlign2*

[in] 二维码 2 垂直对齐方式。

[in] 当 vAlign2 值大于 2 时,可以自定义二维码 2 垂直位置（不能超过当前设置的二维码高度,否则无法打印二维码）。

| 对齐方式             | 值 | 描述   |
|------------------|---|------|
| ALIGNMENT_TOP    | 0 | 上对齐  |
| ALIGNMENT_CENTER | 1 | 垂直居中 |
| ALIGNMENT_BOTTOM | 2 | 下对齐  |

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.20 PrintImage

此函数功能为打印指定的图片(支持 bmp、jpg、gif 等格式)。在页模式下，位图只储存在打印缓冲区且不打印。

```
int PrintImage(  
    void* handle,  
    const TCHAR* filePath,  
    int scaleMode  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*const TCHAR\* filePath*

[in] 图片的完整路径。

*int scaleMode*

[in] 打印图片的缩放模式。

| 模式                        | 值 | 描述   |
|---------------------------|---|------|
| PRINT_IMAGE_NORMAL        | 0 | 正常模式 |
| PRINT_IMAGE_DOUBLE_WIDTH  | 1 | 倍宽模式 |
| PRINT_IMAGE_DOUBLE_HEIGHT | 2 | 倍高模式 |
| PRINT_IMAGE_QUADRUPLE     | 3 | 四倍模式 |

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_NOT_ENOUGHMEMORY   | -9   | 计算机内存不足   |
| E_IMAGE_BAD_SIZE     | -25  | 图片大小错误    |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.21 PrintBitMapData

此函数功能为用指定图数据打印光栅位图（光栅格式）。

```
int PrintBitMapData(  
    void* handle,  
    int scaleMode,  
    int width,  
    int height,  
    unsigned char* data  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int scaleMode*

[in] 打印图片的缩放模式。

| 模式                        | 值 | 描述   |
|---------------------------|---|------|
| PRINT_IMAGE_NORMAL        | 0 | 正常模式 |
| PRINT_IMAGE_DOUBLE_WIDTH  | 1 | 倍宽模式 |
| PRINT_IMAGE_DOUBLE_HEIGHT | 2 | 倍高模式 |
| PRINT_IMAGE_QUADRUPLE     | 3 | 四倍模式 |

*int width*

[in] 宽度指定图水平方向上的 n 个字节。

0≤宽度≤72

*int height*

[in] 高度指定图垂直方向上的 n 个点。

*unsigned char\* data*

[in] 数据指定图数据（光栅格式）。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_NOT_ENOUGHMEMORY   | -9   | 计算机内存不足   |
| E_IMAGE_BAD_SIZE     | -25  | 图片大小错误    |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.22 DefineNVImageCompatible

此函数功能为在 NV 图形缓存区域定义指定的 NV 位图。可以同时下载多张图片，下载到打印机的图片编号从 1 开始累加。只有部分机型支持此功能，未来机型可能不支持此功能。建议使用 NV 图形函数<DefineNVImage>。

```
int DefineNVImageCompatible(  
    void* handle,  
    const TCHAR** filePathList,  
    int imageQty  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*const TCHAR\*\* filePathList*

[in] 指定的图片路径列表。

*int imageQty*

[in] 指定图片的数量。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_NOT_ENOUGHMEMORY   | -9   | 计算机内存不足   |
| E_IMAGE_BAD_SIZE     | -25  | 图片大小错误    |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.23 PrintNVImageCompatible

此函数功能为打印由<DefineNVImageCompatible>下载的 NV 位图。只有部分机型支持此功能，未来机型可能不支持此功能。建议使用 NV 图形函<PrintNVImage>。

```
int PrintNVImageCompatible(  
    void* handle,  
    int imgNo,  
    int scaleMode  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int imgNo*

[in] 打印指定的第 n 张图片（不打印在 NV 缓冲区未定义的图片系列号）。  
 $1 \leq n \leq 255$

*int scaleMode*

[in] 打印图片的缩放模式。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.24 DefineDownloadedImageCompatible

此函数功能为在下载的区域定义下载的位图。只有部分机型支持此功能，未来机型可能不支持此功能。建议使用 NV 图形函数 <DefineDownloadedImage>。

```
int DefineDownloadedImageCompatible(  
    void* handle,  
    const TCHAR* filePath  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*const TCHAR\* filePath*

[in] 图片的完整路径。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_NOT_ENOUGHMEMORY   | -9   | 计算机内存不足   |
| E_IMAGE_BAD_SIZE     | -25  | 图片大小错误    |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.25 PrintDownloadedImageCompatible

此函数功能为打印下载的位图。只有部分机型支持此功能，未来机型可能不支持此功能。建议使用 NV 图形函数<PrintDownloadedImage>。

```
int PrintDownloadedImageCompatible(  
    void* handle,  
    int scaleMode  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int scalemode*

[in] 打印图片的缩放模式。

| 模式                        | 值 | 描述   |
|---------------------------|---|------|
| PRINT_IMAGE_NORMAL        | 0 | 正常模式 |
| PRINT_IMAGE_DOUBLE_WIDTH  | 1 | 倍宽模式 |
| PRINT_IMAGE_DOUBLE_HEIGHT | 2 | 倍高模式 |
| PRINT_IMAGE_QUADRUPLE     | 3 | 四倍模式 |

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.26 GetFirmwareVersion

此函数功能为获取当前打印机的固件版本号。

```
int GetFirmwareVersion(  
    void* handle,  
    int* version,  
    int versionLen  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int\* version*

[in,out] 打印机固件版本号（例如：1.3.12）。

*int versionLen*

[in] 固件版本数据长度。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |
| E_IO_READ_FAILED     | -331 | 读取数据失败    |
| E_IO_READ_TIMEOUT    | -332 | 读取数据超时    |

## 4.27 SelectPageMode

此函数功能为切换标准模式到页模式（仅打印机支持页模式并且在标准模式下时有效）。

```
int SelectPageMode(  
    void* handle  
);
```

参数：

*void\* handle*  
[in,out] 创建的目标打印机对象。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.28 SelectStandardMode

此函数功能为切换页模式到标准模式（仅在页模式下有效）。

```
int SelectStandardMode(  
    void* handle  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.29 SelectPrintDirectionInPageMode

此函数功能为在页模式下，选择打印机的打印方向。此函数只有在页模式下有效。

```
int SelectPrintDirectionInPageMode(  
    void* handle,  
    int direction  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int direction*

[in] 选择打印方向。

| 打印方向                          | 值 | 描述   | 起始位置 |
|-------------------------------|---|------|------|
| PRINT_DIRECTION_LEFT_TO_RIGHT | 0 | 左->右 | 左上角  |
| PRINT_DIRECTION_BOTTOM_TO_TOP | 1 | 下->上 | 左下角  |
| PRINT_DIRECTION_RIGHT_TO_LEFT | 2 | 右->左 | 右下角  |
| PRINT_DIRECTION_TOP_TO_BOTTOM | 3 | 上->下 | 右上角  |

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.30 SetAbsoluteVerticalPrintPositionInPageMode

此函数功能为在页模式下，设置垂直打印位置（当起始打印位置为左上角或右下角时，为设置垂直位置；当起始打印位置为左下角或右上角时，为设置水平位置）。

```
int SetAbsoluteVerticalPrintPositionInPageMode(  
    void* handle,  
    int position  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int position*

[in] 设置垂直位置。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.31 PrintAndReturnStandardMode

此函数功能为打印并返回到标准模式（仅在页模式下有效）。

```
int PrintAndReturnStandardMode(  
    void* handle,  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.32 SetPrintAreaInPageMode

此函数功能为在页模式下，设置打印区域的大小和逻辑起点。打印区域的宽度和高度都不能设为零。

```
int SetPrintAreaInPageMode(  
    void* handle,  
    int horizontal,  
    int vertical,  
    int width,  
    int height  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int horizontal*

[in] 设置起始打印的水平位置。

*int vertical*

[in] 设置起始打印的垂直位置。

*int width*

[in] 设置可打印区域的水平宽度。

*int height*

[in] 设置可打印区域的垂直高度。

当打印纸为 80mm 宽度时：水平起点 = 0，垂直起点 = 0，宽度 = 576，高度 = 840。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_PARAMETER  | -1   | 无效的参数     |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.33 PrintDataInPageMode

此函数功能为在页模式下打印数据，打印后不返回标准模式（仅在页模式下有效）。

```
int PrintDataInPageMode(  
    void* handle  
);
```

参数：

*void\* handle*  
[in,out] 创建的目标打印机对象。

返回值：

| 错误代码                 | 值    | 描述        |
|----------------------|------|-----------|
| E_SUCCESS            | 0    | 正常        |
| E_INVALID_MODEL_TYPE | -3   | 该机型不支持此功能 |
| E_BAD_HANDLE         | -6   | 句柄无效      |
| E_IO_PORT_NOT_OPEN   | -309 | 端口未打开     |
| E_IO_WRITE_FAILED    | -321 | 写入数据失败    |
| E_IO_WRITE_TIMEOUT   | -322 | 写入数据超时    |

## 4.34 DirectIO

此函数功能为用户自定义发送和读取打印机的数据。如果某些功能未提供函数接口时，用户可通过此接口向打印机发送指令数据。

```
int DirectIO(  
    void* handle,  
    unsigned char* writeData,  
    unsigned int writeNum,  
    unsigned char* readData,  
    unsigned int readNum,  
    unsigned int* preadNum  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*unsigned char\* writeData*

[in] 写入打印机的数据。

*unsigned int writeNum*,

[in] 写入数据的长度。当 writNum=0时，不执行写入数据操作。

*unsigned char\* readData*,

[in,out] 获取打印机返回的数据。

*unsigned int readNum*,

[in] 预设需要读取的数据长度。当 readNum=0时，不执行读取数据操作。

*unsigned int\* preadedNum*

[in,out] 实际读取到的数据长度。

返回值：

| 错误代码               | 值    | 描述     |
|--------------------|------|--------|
| E_SUCCESS          | 0    | 正常     |
| E_BAD_HANDLE       | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN | -309 | 端口未打开  |
| E_IO_WRITE_FAILED  | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT | -322 | 写入数据超时 |
| E_IO_READ_FAILED   | -331 | 读取数据失败 |
| E_IO_READ_TIMEOUT  | -332 | 读取数据超时 |

## 4.35 SetAbsolutePrintPosition

此函数功能为从打印区域左边沿移动打印位置至  $n \times$  (水平或垂直移动单元)。  
打印机忽略超过打印区域的任何设置。  
选择标准模式时，使用水平移动单元。  
选择页模式时，使用水平或垂直移动单元作为打印方向。

```
int SetAbsolutePrintPosition(  
    void* handle,  
    int position  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int position*

[in] 水平起始打印位置。

返回值：

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.36 PositionNextLabel

此函数功能为打印标签内容并定位下一个标签起始位置。

```
int PositionNextLabel(  
    void* handle  
);
```

参数：

*void\* handle*  
[in,out] 创建的目标打印机对象。

返回值：

| 错误代码               | 值    | 描述     |
|--------------------|------|--------|
| E_SUCCESS          | 0    | 正常     |
| E_BAD_HANDLE       | -6   | 无效的参数  |
| E_IO_PORT_NOT_OPEN | -309 | 端口未打开  |
| E_IO_WRITE_FAILED  | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT | -322 | 写入数据超时 |

### 4.37 DefineNVImage

此函数功能为定义 NV 图形数据（光栅格式）作为在 NV 图形区域的键码（kc1及kc2）所指定的一个记录。

```
int DefineNVImage(  
    void* handle,  
    const char* imagePath,  
    unsigned char kc1,  
    unsigned char kc2  
);
```

参数：

*void\* handle*  
[in,out] 创建的目标打印机对象。

*const char\* imagePath*  
[in] 指定图片完整路径。

*unsigned char kc1*  
[in] 键码1 32≤kc1≤126。

*unsigned char kc2*  
[in] 键码2 32≤kc2≤126。

返回值：

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.38 PrintNVImage

此函数功能为打印由键码（kc1及 kc2）定义的 NV 图形数据。

```
int PrintNVImage(  
    void* handle,  
    unsigned char kc1,  
    unsigned char kc2,  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*unsigned char kc1*

[in] 键码1 32≤kc1≤126。

*unsigned char kc2*

[in] 键码2 32≤kc2≤126。

返回值：

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

### 4.39 DefineDownloadedImage

此函数功能为定义下载的图形数据（光栅格式）作为在下载图形区域的键码（kc1及kc2）所指定的一个记录。

```
int DefineDownloadedImage(  
    void* handle,  
    const char* imagePath,  
    unsigned char kc1,  
    unsigned char kc2  
);
```

参数：

*void\* handle*  
[in,out] 创建的目标打印机对象。

*const char\* imagePath*  
[in] 图片的路径。

*unsigned char kc1*  
[in] 键码1 32≤kc1≤126。

*unsigned char kc2*  
[in] 键码2 32≤kc2≤126。

返回值：

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.40 PrintDownloadedImage

此函数功能为打印下载的由键码(kc1 and kc2)定义的图形数据。

```
int PrintDownloadedImage(  
    void* handle,  
    unsigned char kc1,  
    unsigned char kc2  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*unsigned char kc1*

[in] 键码1  $32 \leq kc1 \leq 126$ 。

*unsigned char kc2*

[in] 键码2  $32 \leq kc2 \leq 126$ 。

返回值：

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.41 DefineBufferedImage

此函数功能为在缓冲区储存图形数据（光栅格式）。

```
int DefineBufferedImage(  
    void* handle,  
    const char* imagePath  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*const char\* imagePath*

[in] 指定图片的完整路径。

返回值：

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.42 PrintBufferedImage

此函数功能为打印由<DefineBufferedImage>存储的缓冲图形数据，当打印缓冲区没有存储图形数据时，打印机将不会打印。

```
int PrintBufferedImage(  
    void* handle  
);
```

参数：

*void\* handle*  
[in,out] 创建的目标打印机对象。

返回值：

| 错误代码               | 值    | 描述     |
|--------------------|------|--------|
| E_SUCCESS          | 0    | 正常     |
| E_BAD_HANDLE       | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN | -309 | 端口未打开  |
| E_IO_WRITE_FAILED  | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT | -322 | 写入数据超时 |

## 4.43 DeleteAllNVImages

此函数功能为删除所有的 NV 图形数据，删除的区域由“未使用区域”指定，所有的键码指定为未定义。

```
int PrintBufferedImage(  
    void* handle  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

返回值：

| 错误代码               | 值    | 描述     |
|--------------------|------|--------|
| E_SUCCESS          | 0    | 正常     |
| E_BAD_HANDLE       | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN | -309 | 端口未打开  |
| E_IO_WRITE_FAILED  | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT | -322 | 写入数据超时 |

## 4.44 GetCashDrawerState

此函数功能为获取当前钱箱的状态。

```
int GetCashDrawerState(  
    void* handle,  
    int* drawerState  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int\* drawerState*

[in,out] 获取钱箱状态值。

0: 表示钱箱打开

1: 表示钱箱关闭或未连接钱箱

返回值：

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.45 ClearBuffer

此函数功能为清除打印机缓存数据。

```
int ClearBuffer(  
    void* handle  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

返回值：

| 错误代码               | 值    | 描述     |
|--------------------|------|--------|
| E_SUCCESS          | 0    | 正常     |
| E_BAD_HANDLE       | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN | -309 | 端口未打开  |
| E_IO_WRITE_FAILED  | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT | -322 | 写入数据超时 |

## 4.46 FormatError

此函数功能为返回未能成功召回函数的信息。

```
int FormatError(  
    int errorNo,  
    int langid,  
    unsigned char* buf,  
    int pos,  
    int bufSize  
);
```

参数：

*int errorNo*

[in] 函数返回的错误码。

*int langid*

[in] 语言 ID 当前只支持简体中文和英文，默认为0（英文）。

*unsigned char\* buf*

[in,out] 保存错误信息。

*int pos*

[in] 数据内存始位置。

*int bufSize*

[in] 数据内存的大小。

返回值：

| 错误代码        | 值              | 描述   |
|-------------|----------------|------|
| E_SUCCESS   | 0              | 正常   |
| OTHER_ERROR | GetLastError() | 系统错误 |

## 4.47 SetAlign

此函数功能为设置打印对齐方式。在页模式下对齐方式无效。

```
int SetAlign(  
    void* handle,  
    int align  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int align*

[in] 设置对齐方式。

| 值    | 对齐方式 |
|------|------|
| 0,48 | 左对齐  |
| 1,49 | 居中   |
| 2,50 | 右对齐  |

返回值:

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.48 SetTextBold

此函数功能为打开或关闭突出模式。

```
int SetTextBold(  
    void* handle,  
    int bold  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int bold*

[in] 设置文本突出模式。

0: 突出模式关闭。

1: 突出模式打开。

返回值:

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.49 SetTextFont

此函数功能为设置文本字体。

```
int SetTextFont(  
    void* handle,  
    int font  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int font*

[in] 设置字体类型

| 值    | 字体     |
|------|--------|
| 0,48 | Font A |
| 1,49 | Font B |

返回值:

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.50 SetBuzzer

此函数功能为打开或关闭蜂鸣器。

```
int SetBuzzer(  
    void* handle,  
    int enable  
);
```

参数：

*void\* handle*

[in,out] 创建的目标打印机对象。

*int enable*

[in] 设置蜂鸣器为打开或关闭。

0: 蜂鸣器关闭。

1: 蜂鸣器打开。

返回值:

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.51 SetLog

此函数功能为开启或关闭日志功能。

```
int SetLog(  
    int enable,  
    const TCHAR* path  
);
```

参数：

*int enable*

[in] 开启或关闭日志功能。

0：日志功能关闭。

1：日志功能打开。

*const TCHAR\* path*

[in] 日志文件完整路径（例如: D:\\log.txt）。

返回值:

| 错误代码                | 值   | 描述       |
|---------------------|-----|----------|
| E_SUCCESS           | 0   | 正常       |
| E_INVALID_PARAMETER | -1  | 无效的参数    |
| E_OPEN_LOG_ERROR    | -31 | 日志功能开启失败 |

## 4.52 SetHorizontalAndVerticalMotionUnits

此函数功能为设置水平方向、垂直方向的移动单元。

```
int SetHorizontalAndVerticalMotionUnits(  
    void* handle,  
    int horizontal,  
    int vertical  
);
```

参数:

*void\* handle*

[in,out] 创建的目标打印机对象。

*int horizontal*

[in] 水平运动单位  $0 \leq \text{horizontal} \leq 255$ 。

*int vertical*

[in] 垂直运动单位  $0 \leq \text{vertical} \leq 255$ 。

返回值:

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.53 DrawLine

此函数功能为打印直线（仅在页模式下适用）。

```
int DrawLine(  
    void* handle,  
    int    x_start,  
    int    y_start,  
    int    x_end,  
    int    y_end,  
    int    lineWidth  
);
```

参数:

*void\* handle*  
[in,out] 创建的目标打印机对象。

*int x\_start*  
[in] 水平起始位置(单位: dot)。

*int y\_start*  
[in] 垂直起始位置(单位: dot)。

*int x\_end*  
[in] 水平结束位置(单位: dot)。

*int y\_end*  
[in] 垂直结束位置(单位: dot)。

*int lineWidth*  
[in] 线条宽度  $1 \leq \text{lineWidth} \leq 255$ 。

返回值:

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |

## 4.54 DrawRectangle

此函数功能为打印矩形（仅在页模式下适用）。

```
int DrawRectangle(  
    void* handle,  
    int    x_start,  
    int    y_start,  
    int    x_area,  
    int    y_area,  
    int    lineWidth  
);
```

参数:

*void\* handle*

[in,out] 创建的目标打印机对象。

*int x\_start*

[in] 水平起始位置(单位: dot)。

*int y\_start*

[in] 垂直起始位置(单位: dot)。

*int x\_area*

[in] 水平打印区域(单位: dot)。

*int y\_area*

[in] 垂直打印区域(单位: dot)。

*int lineWidth*

[in] 线条宽度  $1 \leq \text{lineWidth} \leq 255$ 。

返回值:

| 错误代码                | 值    | 描述     |
|---------------------|------|--------|
| E_SUCCESS           | 0    | 正常     |
| E_INVALID_PARAMETER | -1   | 无效的参数  |
| E_BAD_HANDLE        | -6   | 句柄无效   |
| E_IO_PORT_NOT_OPEN  | -309 | 端口未打开  |
| E_IO_WRITE_FAILED   | -321 | 写入数据失败 |
| E_IO_WRITE_TIMEOUT  | -322 | 写入数据超时 |