

[TSPL Windows SDK]

[打印机 TSPL 指令集开发帮助文档 v1.5]

1.手册信息	3
2.操作系统	3
3.备注	3
4.方法	4
4.1 PrinterCreator	4
4.2 PrinterCreatorS	5
4.3 PrinterDestroy.....	6
4.4 PortOpen.....	7
4.5 PortClose	9
4.6 WriteData	10
4.7 ReadData	11
4.8 DirectIO	12
4.9 TSPL_SelfTest.....	14
4.10 TSPL_BitMap.....	15
4.11 TSPL_Setup.....	17
4.12 TSPL_ClearBuffer.....	19
4.13 TSPL_Box	20
4.14 TSPL_BarCode.....	22
4.15 TSPL_QrCode	25
4.16 TSPL_Text	27
4.17 TSPL_Print	29
4.18 TSPL_FormFeed	30
4.19 TSPL_SetTear	31
4.20 TSPL_SetRibbon	32
4.21 TSPL_Offset.....	33
4.22 TSPL_Direction.....	34
4.23 TSPL_Feed	35
4.24 TSPL_Home	36
4.25 TSPL_Learn.....	37
4.26 TSPL_GetSN	38
4.27 TSPL_GetPrinterStatus.....	39
4.28 TSPL_SetCodePage.....	40
4.29 TSPL_PDF417.....	42
4.30 TSPL_Block	45
4.31 TSPL_Reverse	47
4.32 TSPL_GapDetect.....	48

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。

4.方法

4.1 PrinterCreator

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 char* model  
  
);
```

参数：

```
void* handle  
    [in, out] 创建目标打印机对象。  
const char* 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_ENOUGMEMORY	-2	计算机内存不足
E_INVALID_MODEL	-8	机型名称无效

4.2 PrinterCreatorS

创建指定机型的目标打印机，其功能与 PrinterCreator 相同（使用任何打印机操作之前必须先创建打印机对象）。

```
Void* PrinterCreatorS(  
  
    const char* model  
  
);
```

参数:

const char model*
[in] 指定目标打印机型号。

返回值:

成功: 返回打印机对象的句柄。

失败: 返回无效。

4.3 PrinterDestroy

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。
```

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

打开通讯端口，以进行打印操作。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 char* ioSettings  
  
);
```

参数:

```
void* handle  
    [in, out] 创建的目标打印机对象。  
const char* ioSettings  
    [in]设置连接目标打印机的通讯端口参数。
```

配置列表:

类别	配置	描述	示例
USB	USB[, Position/M odle/PortNum]	USB: 连接任一本公司 USB 打印机。 USB[, Position]: 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 USB, Port_#0004. Hub_#0003 USB, LPG4 USB, USB001
NET	NET, IP 地址 (IPV4)[, Port]	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 NET, 192. 168. 0. 36, 9100
COM	COMn, BAUDRATE_ rate	指定连接的串口端口号和波特率。	COM5, BAUDRATE_19200
LPT	LPTn	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

此函数功能为关闭打印连接的通讯端口。

```
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 WriteData

此函数功能为向打印机发送数据。

```
int WriteData(  
  
    void* handle,  
  
    unsigned char* writeData,  
  
    unsigned int writeNum  
  
);
```

参数：

- void* handle*
[in, out] 创建的目标打印机对象。
- unsigned char* writeData*
[in] 发给打印机的数据，数据是十六进制字符串。
- unsigned int writeNum*
[in] 发送数据的长度。

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	通信端口未打开。
E_IO_WRITE_FAILED	-321	写入失败。
E_IO_WRITE_TIMEOUT	-322	写入超时。
E_IO_READ_FAILED	-331	读取失败。
E_IO_READ_TIMEOUT	-332	读取超时。

4.7 ReadData

此函数功能为读取打印机的数据。

```
int ReadData(  
  
    void* handle,  
  
    unsigned char* readData,  
  
    unsigned int readNum,  
  
    unsigned int* preadedNum  
  
);
```

参数:

- void* handle*
[in, out] 创建的目标打印机对象。
- unsigned char* readData*
[in] 需要读取的打印机数据。
- unsigned int readNum*
[in] 所需读取的数据长度。
- unsigned int* preadedNum*
[in] 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	通信端口未打开。
E_IO_READ_FAILED	-331	读取失败。
E_IO_READ_TIMEOUT	-332	读取超时。

4.8 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，用户可以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] 写入打印机的数据长度。当 writeNum=0时，不实行写入数据操作。

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，不实行读取数据操作。

unsigned int preadNum*

[in, out] 实际读取的数据长度。

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。

4.9 TSPL_SelfTest

此函数功能为打印机打印自检页，自检页中包含打印机的基本配置信息。

```
int TSPL_SelfTest(  
  
    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。

4.10 TSPL_BitMap

此函数功能为打印指定的图片(支持 bmp、jpg、gif 等格式),此函数适用于打印 LOGO。

将指定路径的图片转化为位图数据发送给打印机并且打印。

```
int TSPL_BitMap(  
  
    void* handle,  
  
    int xPos,  
  
    int yPos,  
  
    int mode,  
  
    const char* filePath  
  
);
```

参数:

```
void* handle  
    [in, out] 创建的目标打印机对象。  
int xPos  
    [in] 水平起始位置(单位 dot)。  
int yPos  
    [in] 垂直起始位置(单位 dot)。  
int mode  
    [in] 打印图片的模式。
```

模式	值
OVERWRITE	0
OR	1
XOR	2

```
const char* filePath  
    [in] 图片的正确路径。
```

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_ENOUGH_MEMORY	-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	通信端口未打开。
E_IO_WRITE_FAILED	-321	写入失败。
E_IO_WRITE_TIMEOUT	-322	写入超时。

4.11 TSPL_Setup

此函数功能为打印机基本参数设置。

```
int TSPL_Setup(  
  
    void* handle,  
  
    int labelWidth,  
  
    int labelHeight,  
  
    int speed,  
  
    int density,  
  
    int type,  
  
    int gap,  
  
    int offset  
  
);
```

参数:

void handle*

[in,out] 创建的目标打印机对象。

int labelWidth

[in] 设置预打印标签宽度(单位:mm)。

int labelHeight

[in] 设置预打印标签长度(单位:mm)。

int speed

[in] 设置打印机打印速度(范围: 普通机型可设置速度 2-4, 高速机型可设置 2-7, 具体规格请参考对应的产品规格书)。

int density

[in] 设置打印机打印浓度(范围:0-15)。

int type

[in] 设置标签纸类型(0:黑标/连续纸 1:缝标/连续纸)。

int gap

[in] 两张标签之间的间距，如果设置为0，表示当前为连续纸。

int offset

[in] 设置剥离模式时每张标签停止的位置。

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。

4.12 TSPL_ClearBuffer

此函数功能为清除打印机内存缓存。执行打印数据前, 请先清除打印机缓存。

```
int TSPL_SelfTest(  
  
    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。

4.13 TSPL_Box

此函数功能为打印直线或矩形。

```
int TSPL_Box(  
  
    void* handle,  
  
    int x_start,  
  
    int y_start,  
  
    int x_end,  
  
    int y_end,  
  
    int thickness,  
  
    );
```

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 thickness

[in] 线条宽度

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 TSPL_BarCode

此函数功能为打印一维条码。

```
int TSPL_BarCode(  
  
    void* handle,  
  
    int xPos,  
  
    int yPos,  
  
    int type,  
  
    int height,  
  
    int readable,  
  
    int rotation,  
  
    int narrow,  
  
    int wide,  
  
    const char* data  
  
);
```

参数:

void handle*
[in, out] 创建的目标打印机对象。

const int xPos
[in] 水平起始位置。

const int yPos
[in] 垂直起始位置。

const int type

[in] 设置条码类型。

条码类型	值
Code 128	0
Code 128M	1
EAN 128	2
Interleaved 2 of 5	3
Interleaved 2 of 5 with check digits	4
Code 39	5
Code 39C	6
Code 39S	7
Code 93	8
EAN 13	9
EAN 13 with 2 digits add-on	10
EAN 13 with 5 digits add-on	11
EAN 8	12
EAN 8 with 2 digits add-on	13
EAN 8 with 5 digits add-on	14
Codabar	15
Postnet	16
UPC-A	17
UPC-A with 2 digits add-on	18
UPC-A with 5 digits add-on	19
UPC-E	20
UPC-E with 2 digits add-on	21
UPC-E with 5 digits add-on	22
China post code	23
MSI code	24
MSI with check digit	25
PLESSEY code	26
ITF 14 code	27
EAN 14 code	28
Code 11	29

int height

[in] 设置条码打印高度, 以点(dot)表示。

int readable

[in] 设置可读(0:不可读, 1:可读)。

int rotation

[in] 设置条码方向。

对齐方式	值
不旋转	0
旋转90度	1
旋转180度	2
旋转270度	3

int narrow

[in] 设置窄 Bar 宽度, 以点(dot)表示。

int wide

[in] 设置宽 Bar 宽度, 以点(dot)表示。

const char data*

[in] 条码数据。

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。

4.15 TSPL_QrCode

此函数功能为打印二维码。

```
int TSPL_QrCode(  
    void* handle,  
    int xPos,  
    int yPos,  
    int eccLv,  
    int width,  
    int mode,  
    int rotation,  
    int model,  
    int mask,  
    const char* data  
);
```

参数:

void handle*

[in, out] 创建的目标打印机对象。

int xPos

[in] 水平起始位置。

int yPos

[in] 垂直起始位置。

int eccLv
[in] 纠错级别。

容错级别	值
7%	0
15%	1
25%	2
30%	3

int width
[in] 设置二维码打印宽度, (范围:1-10)。

int mode
[in] 自动/手动编码(A:自动, 1:手动)。

int rotation
[in] 设置条码方向。

对齐方式	值
不旋转	0
旋转90度	1
旋转180度	2
旋转270度	3

int model
[in] 设置二维码版本(0:基本版, 1:强化版)。

int mask
[in] 范围: 0 - 8 。

const char data*
[in] 二维码数据。

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。

4.16 TSPL_Text

此函数功能为打印机打印文本。

```
int TSPL_Text(  
    void* handle,  
    int xPos,  
    int yPos,  
    int font,  
    int rotation,  
    int xMultiplication,  
    int yMultiplication ,  
    const char* data  
);
```

参数:

void handle*

[in, out] 创建的目标打印机对象。

int xPos

[in] 水平起始位置。

int yPos

[in] 垂直起始位置。

int font
[in] 字体。

字体点阵	值
normal	0
8x12	1
12x20	2
16x24	3
24x32	4
32x48	5
14x19	6
21x27	7
14x25	8
汉字	9

int rotation
[in] 设置打印方向。

对齐方式	值
不旋转	0
旋转90度	1
旋转180度	2
旋转270度	3

int xMultiplication
[in] 字体宽度放大倍数。

int yMultiplication
[in] 字体高度放大倍数。

const char data*
[in] 文本数据。

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 TSPL_Print

此函数功能为执行打印操作。

```
int TSPL_Text(  
  
    void* handle,  
  
    int num,  
  
    int copies  
  
);
```

参数:

- void* handle*
[in, out] 创建的目标打印机对象。
- int num*
[in] 不同标签个数。
- int copies*
[in] 打印每个标签的份数。

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。

4.18 TSPL_FormFeed

此函数功能为打印机执行走纸操作, 定位到下一张标签。

```
int TSPL_FormFeed(  
  
    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。

4.19 TSPL_SetTear

此函数功能为设置打印机启用/禁用撕纸功能。

```
int TSPL_SetTear(  
  
    void* handle,  
  
    int on_off  
  
);
```

参数:

```
void* handle  
    [in, out] 创建的目标打印机对象。  
int on_off  
    [in] 启用/禁用撕纸功能 (0: OFF, 1:ON)。
```

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。

4.20 TSPL_SetRibbon

此函数功能为设置用于启用/禁用带传感器检测。

```
int TSPL_SetRibbon(  
  
    void* handle,  
  
    int on_off  
  
);
```

参数:

- void* handle*
[in, out] 创建的目标打印机对象。
- int on_off*
[in] 启用/禁用带传感器检测:(0: OFF, 1:ON)。

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。

4.21 TSPL_Offset

此函数功能为定义打印机每个标签走纸额外指定长度。

```
int TSPL_Offset(  
  
    void* handle,  
  
    int distance  
  
);
```

参数:

```
void* handle  
    [in, out] 创建的目标打印机对象。  
int distance  
    [in] 指定走纸长度(单位:mm)。
```

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。

4.22 TSPL_Direction

此函数功能为设置打印机打印方向。

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

参数:

```
void* handle  
    [in, out] 创建的目标打印机对象。  
int direction  
    [in] 打印方向(0: 正常方向, 非0: 旋转180度)。
```

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。

4.23 TSPL_Feed

此函数功能为打印机走纸指定长度。

```
int TSPL_Feed(  
  
    void* handle,  
  
    int len  
  
);
```

参数:

```
void* handle  
    [in, out] 创建的目标打印机对象。  
int len  
    [in] 指定走纸长度(单位:dot)。
```

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。

4.24 TSPL_Home

此函数功能为打印机定位标签。

```
int TSPL_Home(  
  
    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。

4.25 TSPL_Learn

此函数功能为打印机学习标签。

```
int TSPL_Learn(  
  
    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。

4.26 TSPL_GetSN

此函数功能为获取打印机 SN 序列号。

```
int TSPL_GetSN(  
  
    void* handle,  
  
    char* sn  
  
);
```

参数:

```
void* handle  
    [in, out] 创建的目标打印机对象。  
char* sn  
    [in] 获取 SN 序列号。
```

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。

4.27 TSPL_GetPrinterStatus

此函数功能为获取打印机状态。

```
int TSPL_GetPrinterStatus(  
  
    void* handle,  
  
    int* status  
  
);
```

参数:

void handle*
[in, out]创建的目标打印机对象。
int status*
[in, out]打印机当前状态。

状态	值
正常	0
打印头被打开	1
卡纸	2
缺纸	4
缺碳带	8
打印暂停	16
打印中	32
上盖打开	64
其它错误	128

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。

4.28 TSPL_SetCodePage

此函数功能为打印机设置代码页。

```
int SPL_CodePage(  
  
    void* handle,  
  
    int codepage  
  
);
```

参数:

```
void* handle  
    [in,out] 创建的目标打印机对象。  
int codepage  
    [in]国际字符集。
```

7-bit code page		8-bit code page		Windows code page		ISO code page	
USA	USA	437	USA	1250	Central Europe	8859-1	Latin 1
BRI	British	737	Greek	1251	Cyrillic	8859-2	Latin 2
GER	German	850	Multilingual	1252	Latin I	8859-3	Latin 3
FRE	French	851	Greek1	1253	Greek	8859-4	Batltic
DAN	Danish	852	Slavic	1254	Turkish	8859-5	Cyrillic
ITA	Italian	855	Cyrillic	1255	Hebrew	8859-6	Arabic
SPA	Spansh	857	Turkish	1256	Arabic	8859-7	Greek
SWE	Sweidsh	860	Portuguese	1257	Batltic	8859-8	Hebrew
SWI	Swiss	861	Icelandic	1258	Vietnam	8859-9	Turkish
		862	Hebrew	932	Japanese shift-JIS	8859-10	Latin 6
		863	Canadian/Fre nch	936	simplified	8859-15	Latin 9
		864	Arabic	949	Korean		
		865	Nordic	950	Traditional Chinese BIG5		
		866	Russian	UTF-8	UTF 8		
		869	Greek2				

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。

4.29 TSPL_PDF417

此函数功能为打印 PDF417 二维码。

```
int TSPL_PDF417(  
  
    void* handle,  
  
    int xPos,  
  
    int yPos,  
  
    int width,  
  
    int height,  
  
    int rotate,  
  
    char* option,  
  
    const char* data  
  
);
```

参数:

void handle*

[in, out] 创建的目标打印机对象。

int xPos

[in] 水平起始位置。

int yPos

[in] 垂直起始位置。

int width

[in] 条码宽度(单位:dot)。

int height

[in] 条码高度(单位:dot)。

int rotate

[in] 设置条码方向。

对齐方式	值
不旋转	0
旋转90度	1
旋转180度	2
旋转270度	3

char option*

[in]可选项。

P	数据压缩方式： 0:自动 1:二进制模式
E	错误校验等级(0~8)
M	条码中心打印模式 0:该模式将打印在左上角对齐区域 1:将打印在中间区域
Ux, y, z	代码可读 x:可读字符指定的 x 坐标 y:可读字符指定的 y 坐标 c:每一行最大可读字符数
W	模块宽度 (2~9:dot)
H	小条码的高(4~99:dot)
R	最大行数
C	最大列数
T	是否截断 0:否 1:是
Lm	表示长度(1~2048)

const char data*

[in] 二维码数据。

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。

4.30 TSPL_Block

此函数功能为打印段内容到标签中。

```
int TSPL_Block(  
  
    void* handle,  
  
    int xPos,  
  
    int yPos,  
  
    int width,  
  
    int height,  
  
    int font,  
  
    int rotate,  
  
    int x_multiplication,  
  
    int y_multiplication,  
  
    int space,  
  
    int alginment,  
  
    const char* data  
  
);
```

参数:

void handle*

[in, out] 创建的目标打印机对象。

int xPos

[in] 水平起始位置。

int yPos

[in] 垂直起始位置。

int width
[in] 条码宽度(单位:dot)。

int height
[in] 条码高度(单位:dot)。

int rotate
[in] 设置条码方向。

对齐方式	值
不旋转	0
旋转90度	1
旋转180度	2
旋转270度	3

int x_multiplication
[in]横向字体放大倍数(1~10)。

int y_multiplication
[in]垂直字体放大倍数(1~10)。

int space
[in]设置行之间的距离(dot)。

int alginment
[in] 设置文本对齐方式。

对齐方式	值
0	默认
1	左对齐
2	中对齐
3	右对齐

const char data*
[in]内容数据。最大为4092个字节数。

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。

4.31 TSPL_Reverse

将指定区域内打印内容反白, 即空白部分为黑色, 预打印部分为白色。

```
int TSPL_Reverse (  
  
    void* handle,  
  
    int xPos,  
  
    int yPos,  
  
    int width,  
  
    int height  
  
);
```

参数:

- void* handle*
[in, out] 创建的目标打印机对象。
- int xPos*
[in] 水平起始位置。
- int yPos*
[in] 垂直起始位置。
- int width*
[in] 水平宽度。
- int height*
[in] 垂直长度。

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。

4.32 TSPL_GapDetect

此函数功能确认标签的尺寸和两标签间间距的大小。

```
int TSPL_GapDetect(  
  
    void* handle,  
  
    int paper_length,  
  
    int gap_length  
);
```

参数:

```
Void* handle  
    [in, out] 创建的目标打印机对象  
int paper_length  
    [in] 纸张的长度（单位：dot）。  
int gap_length  
    [in] 间隙的长度（单位：dot）。
```

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。