

[ZPL Linux SDK]

[打印机 ZPL 指令集开发帮助文档 v2.0.7]

目录

[ZPL Linux SDK]	1
1. 手册信息	4
2. 运行平台	4
3. 备注	4
4. 方法	4
4.1. PrinterCreator	4
4.2. ReleasePrinter	5
4.3. PortOpen	5
4.4. ClosePort	6
4.5. WriteData	6
4.6. ReadData	7
4.7. ZPL_StartFormat	7
4.8. ZPL_EndFormat	8
4.9. ZPL_ScalableFontText	8
4.10. ZPL_Text	9
4.11. ZPL_BarCode39	11
4.12. ZPL_Pdf417	12
4.13. ZPL_CodeEan8	13
4.14. ZPL_UpceCode	14
4.15. ZPL_BarCode93	15
4.16. ZPL_BarCode128	16
4.17. ZPL_CodeEan13	18
4.18. ZPL_MicroPdf417	19
4.19. ZPL_QRCode	20
4.20. ZPL_UpcExtensions	22
4.21. ZPL_UpcaBarcode	23
4.22. ZPL_SetChangeFontEncoding	24
4.23. ZPL_SetChangeCaret	25
4.24. ZPL_SetChangeDelimiter	25
4.25. ZPL_SetChangeTilde	26
4.26. ZPL_GraphicBox	26
4.27. ZPL_GraphicCircle	27
4.28. ZPL_GraphicDiagonalLine	28
4.29. ZPL_GraphicEllipse	29
4.30. ZPL_PrintImage	30
4.31. ZPL_GraphicSymbol	30
4.32. ZPL_SetDiagnosticsMode	31
4.33. ZPL_SetLabelHome	32
4.34. ZPL_SetLabelLength	32
4.35. ZPL_SetLabelShift	33
4.36. ZPL_SetLabelTop	33
4.37. ZPL_SetPrintMode	34
4.38. ZPL_SetMediaType	35
4.39. ZPL_SetPrintingMirrorImage	35
4.40. ZPL_SetPrintOrientation	36
4.41. ZPL_SetPrintRate	36
4.42. ZPL_SetPrintWidth	37
4.43. ZPL_SetSerialCommunications	37
4.44. ZPL_SetPrintDarkness	38
4.45. ZPL_SetTearOffAdjustPosition	39
4.46. ZPL_PrintConfigurationLabel	39
4.47. ZPL_GetPrinterIpAddress	40
4.48. ZPL_GetPrinterStatus	41
4.49. ZPL_GetLabelLength	41

4.50. ZPL_GetLabelWidth	42
4.51. ZPL_GetPrinterSeriesNumber	42
4.52. ZPL_GetPrinterMacAddress	43
4.53. ZPL_GetPrinterName	43
4.54. ZPL_GetPrinterFirmwareVersion	44
4.55. ZPL_GetPrinterDpi	44
4.56. ZPL_GetPrinterModel	45
4.57. ZPL_LearnLabel	46
4.58. ZPL_SetReprintAfterError	46
4.59. ZPL_SetMediaTracking	47
4.60. ZPL_SetUserFontName	47
4.61. ZPL_Text_Block	48
4.62. ZPL_SetPrintQuantity	50
4.63. ZPL_DataMatrixBarcode	50
4.64. ZPL_GetPrinterOdometer	51
4.65. ZPL_SetPrintNetSetting	52
4.66. ZPL_WifiConfig	53
4.67. ZPL_SetPrinterBluetoothSSID	53
4.68. ZPL_SetPrinterBluetoothPIN	54
4.69. ZPL_SetPrinterSleepTime	54
4.70. ZPL_SetPrinterShutdownTime	55
4.71. ZPL_FirmwareUpgrade	55
4.72. ZPL_FontDownload	56
4.73. ZPL_RfidCalibration	57
4.74. ZPL_RfidWrite	57
4.75. ZPL_RfidDefineFont	58
4.76. ZPL_RfidRead	59
4.77. ZPL_RfidSetPower	60
4.78. ZPL_RfidDefineEPC	61
4.79. ZPL_RfidSetParam	61

1. 手册信息

本 SDK 手册提供了 Linux 应用程序开发所需的*.so 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. 运行平台

Linux debian 5.10.0 及以上版本

3. 备注

1. 错误代码返回值大于0时，属于 Linux 系统内部错误，请查阅相关帮助文档。
2. 打印机分辨率为200 dpi时，1 mm=8 dot(点);打印机分辨率为300 dpi时，1 mm=12 dot(点)。
3. SDK中引用了第三方库:libserialport、libusb-1.0。请提前在操作系统安装。
4. 串口连接需要root权限。

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 char* model  
);
```

参数：

*void** handle*

[in,out] 创建目标打印机对象。

const TCHAR model*

[in] 指定目标打印机型号。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_PARAMETER	-1	参数无效

4.2.ReleasePrinter

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 ReleasePrinter (  
    void* hPrinter  
);
```

参数：

void hPrinter*

[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
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败

4.3.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。

```
int PortOpen(  
    void* hPrinter,  
    const TCHAR* setting  
);
```

参数：

void hPrinter*

[in] 打印机对象句柄。

const TCHAR setting*

[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	USB,path	USB,USB路径	USB,/001/007
NET	NET, IP 地址 (IPV4)[, 端口]	指定网络打印机的IP地址和端口。如果不指定端，默认端口是9100。	NET,192.168.1.10 NET,192.168.1.10,9100
COM	COM,path,rate	指定连接的串口路径和波特率。	COM,/dev/ttyACM0,19200

返回值：

错误代码	值	描述
------	---	----

ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_USB_DEVICE_NOT_FOUND	-17	找不到USB设备
ERROR_IO_OPEN_FAILED	-8	打开IO失败
ERROR_CM_INVALID_PARAMETER	-1	参数无效

4.4. ClosePort

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 ClosePort (
    void* hPrinter
);
```

参数：

void* hPrinter
[in] 创建的目标打印机对象。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-3	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-2	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败

4.5. WriteData

此函数功能为向打印机发送数据。

```
int WriteData(
    void* handle,
    unsigned char* buffer,
    unsigned int size
);
```

参数：

void* handle
[in] 打印机对象句柄。
unsigned char* buffer
[in] 发给打印机的数据，数据是十六进制字符串。
unsigned int size
[in] 发送数据的长度。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效

ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时

4.6.ReadData

此函数功能为读取打印机的数据。

```
int ReadData(
    void* handle,
    unsigned char* buffer,
    unsigned int size
);
int ReadDataTimeout(void* handle, int timeout, unsigned char* buffer, unsigned int size);
```

参数：

void handle*

[in] 打印机对象句柄。

unsigned char buffer*

[in] 需要读取的打印机数据。

int timeout

[in] 读取超时时间

unsigned int size

[in] 所需读取的数据长度。

返回值：

错误代码	值	描述
	>0	成功，读取的数据长度
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_OPEN_FAILED	-8	打开IO失败

4.7. ZPL_StartFormat

此函数功能为表示一个新的标签格式的开始。

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

参数：

void handle*

[in] 创建的目标打印机对象。

返回值：

错误代码	值	描述
------	---	----

ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.8. ZPL_EndFormat

此函数功能为表示一个标签格式的结束。

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

参数：

void handle*
[in] 创建的目标打印机对象。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.9. ZPL_ScalableFontText

此函数功能为打印可缩放字体。

```
int ZPL_ScalableFontText(
    void* handle,
    int xPos,
    int yPos,
    char fontName,
    int orientation,
    int fontWidth,
    int fontHeight,
    const TCHAR* text
);
```

参数：

void handle*
[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围：0-32000，单位：dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

char fontName

[in] 字体 (取值：A-Z 和 0-9)。

int orientation

[in] 打印方向。

0：正常

90：顺时针旋转90度

180：顺时针旋转180度

270：顺时针旋转270度

int fontWidth

[in] 字体宽度。

int fontHeight

[in] 字体高度。

const TCHAR text*

[in] 文本数据。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.10. ZPL_Text

此函数功能为打印文本。

```
int ZPL_Text(  
    void* handle,  
    int xPos,  
    int yPos,  
    int fontNum,  
    int orientation,  
    int fontWidth,  
    int fontHeight,  
    const TCHAR* text  
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围：0-32000，单位：dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int fontNum

[in] 字体。

- 0 : FONT 0 - 可缩放字体
- 1 : FONT A - 位图字体
- 2 : FONT B - 位图字体
- 3 : FONT D - 位图字体
- 4 : FONT E - 位图字体
- 5 : FONT F - 位图字体
- 6 : FONT G - 位图字体
- 7 : FONT H - 位图字体
- 8 : FONT P - 位图字体
- 9 : FONT Q - 位图字体
- 10 : FONT R - 位图字体
- 11 : FONT S - 位图字体
- 12 : FONT T - 位图字体
- 13 : FONT U - 位图字体
- 14 : FONT V - 位图字体

int orientation

[in] 打印方向。

- 0 : 正常
- 90 : 顺时针旋转90度
- 180 : 顺时针旋转180度
- 270 : 顺时针旋转270度

int fontWidth

[in] 字体宽度。

int fontHeight

[in] 字体高度。

字体	高 x 宽 (以点为单位)
A	9 x 5
B	11 x 7
D	18 x 10
E	28 x 15
F	26 x 13
G	60 x 40
H	21 x 13
GS	24 x 24
P	20 x 18
Q	28 x 24
R	35 x 31
S	40 x 35
T	48 x 42
U	59 x 53
V	80 x 71
0	15 x 12

FONT A -- ABCDxyz 12345
 FONT B -- ABCDxyz 12345 UPPER CASE ONLY
 FONT D -- ABCDxyz 12345
 FONT E -- (OCR-B) ABCDxyz 12345
 FONT F -- ABCDxyz 12345
 FONT G -- ABYz 12
 FONT H -- (OCR-A) UPPER CASE ONLY
 FONT O -- (Scaleable) ABCDxyz 12345
 FONT GS -- ® ¢ ™ ®
 FONT P -- ABCDxyz 12345
 FONT Q -- ABCDxyz 12345
 FONT R -- ABCDxyz 12345
 FONT S -- ABCDxyz 12345
 FONT T -- ABCDxyz 12345
 FONT U -- ABCDxyz 12345
 FONT V -- ABCDxyz 12345

const TCHAR text*

[in] 文本数据。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效

ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.11. ZPL_BarCode39

此函数功能为打印 Barcode39条码。

```
int ZPL_BarCode39(
    void* handle,
    int xPos,
    int yPos,
    int orientation,
    int moduleWidth,
    int codeHeight,
    char line,
    char lineAboveCode,
    char digit,
    const TCHAR* text
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围：0-32000，单位：dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int orientation

[in] 打印方向。

0：正常

90：顺时针旋转90度

180：顺时针旋转180度

270：顺时针旋转270度

int moduleWidth

[in] 条码宽度 (范围：0-10，单位：dot)。

int codeHeight

[in] 条码高度 (范围：1-32000，单位：dot)。

char line

[in] 注释行。

‘N’：不打印

‘Y’：打印

char lineAboveCode

[in] 条码上方的注释行。

‘N’：不打印在条码上方

‘Y’：打印在条码上方

char digit

[in] 校验位。

‘N’：不打印校验位
‘Y’：打印校验位
const TCHAR* text
[in] 条码数据。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.12. ZPL_Pdf417

此函数功能为打印PDF417二维码。

```
int ZPL_Pdf417(
    void* handle,
    int xPos,
    int yPos,
    int orientation,
    int moduleWidth,
    int codeHeight,
    int securityLevel,
    int columns,
    int rows,
    char truncate,
    const TCHAR* text
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围：0-32000，单位：dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int orientation

[in] 打印方向。

0：正常

90：顺时针旋转90度

180：顺时针旋转180度

270：顺时针旋转270度

int moduleWidth

[in] 条码宽度 (范围：0-10，单位：dot)。

int codeHeight

[in] 条码高度 (范围：1-32000，单位：dot)。

int securityLevel

[in] 安全级别 (范围： 1-8) 。

int column
[in] 要编码的列数。

int rows
[in] 要编码的行数。

char truncate
[in] 截断层指示和停止模式。
‘N’:不截断
‘Y’:执行截断

const TCHAR text*
[in] 二维码数据。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.13. ZPL_CodeEan8

此函数功能为打印 CodeEan8条码。

```
int ZPL_CodeEan8(
    void* handle,
    int xPos,
    int yPos,
    int orientation,
    int moduleWidth,
    int codeHeight,
    char line,
    char lineAboveCode,
    const TCHAR* text
);
```

参数:

void handle*
[in] 创建的目标打印机对象。

int xPos
[in] 水平起始位置 (范围： 0-32000 ， 单位： dot) 。

int yPos
[in] 垂直起始位置 (范围： 0-32000 ， 单位： dot) 。

int orientation
[in] 打印方向。
0 : 正常
90 : 顺时针旋转90度
180 : 顺时针旋转180度

270 : 顺时针旋转270度

int moduleWidth

[in] 条码宽度 (范围: 0-10 , 单位: dot)。

int codeHeight

[in] 条码高度 (范围: 1-32000 , 单位: dot)。

char line

[in] 注释行。

‘N’: 不打印

‘Y’: 打印

char lineAboveCode

[in] 条码上方的注释行。

‘N’: 不打印在条码上方

‘Y’: 打印在条码上方

const TCHAR text*

[in] 条码数据。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.14. ZPL_UpceCode

此函数功能为打印 UPC-E 条码。

```
int ZPL_UpceCode(  
    void* handle,  
    int xPos,  
    int yPos,  
    int orientation,  
    int moduleWidth,  
    int codeHeight,  
    char line,  
    char lineAboveCode,  
    const TCHAR* text  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围: 0-32000 , 单位: dot)。

int yPos

[in] 垂直起始位置 (范围: 0-32000 , 单位: dot)。

int orientation

[in] 打印方向。

0 : 正常
 90 : 顺时针旋转90度
 180 : 顺时针旋转180度
 270 : 顺时针旋转270度

int moduleWidth

[in] 条码宽度 (范围: 0-10 , 单位: dot) 。

int codeHeight

[in] 条码高度 (范围: 1-32000 , 单位: dot) 。

char line

[in] 注释行。

‘N’: 不打印

‘Y’: 打印

char lineAboveCode

[in] 条码上方的注释行。

‘N’: 不打印在条码上方

‘Y’: 打印在条码上方

const TCHAR text*

[in] 条码数据。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.15. ZPL_BarCode93

此函数功能为打印 Barcode93条码。

```
int ZPL_BarCode93(
    void* handle,
    int xPos,
    int yPos,
    int orientation,
    int moduleWidth,
    int codeHeight,
    char line,
    char lineAboveCode,
    char digit,
    const TCHAR* text
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围: 0-32000 , 单位: dot) 。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int orientation

[in] 打印方向。

0：正常

90：顺时针旋转90度

180：顺时针旋转180度

270：顺时针旋转270度

int moduleWidth

[in] 条码宽度 (范围：0-10，单位：dot)。

int codeHeight

[in] 条码高度 (范围：1-32000，单位：dot)。

char line

[in] 注释行。

‘N’：不打印

‘Y’：打印

char lineAboveCode

[in] 条码上方的注释行。

‘N’：不打印在条码上方

‘Y’：打印在条码上方

char digit

[in] 校验位。

‘N’：不打印校验位

‘Y’：打印校验位

const TCHAR text*

[in] 条码数据。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.16. ZPL_BarCode128

此函数功能为打印 Barcode128条码。

int ZPL_BarCode128(

void handle,*

int xPos,

int yPos,

int orientation,

int moduleWidth,

int codeHeight,

char line,

char lineAboveCode,

char checkDigit,


```

        char mode,
        const TCHAR* text
    );

```

参数:

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围: 0-32000, 单位: dot)。

int yPos

[in] 垂直起始位置 (范围: 0-32000, 单位: dot)。

int orientation

[in] 打印方向。

0 : 正常

90 : 顺时针旋转90度

180 : 顺时针旋转180度

270 : 顺时针旋转270度

int moduleWidth

[in] 条码宽度 (范围: 0-10, 单位: dot)。

int codeHeight

[in] 条码高度 (范围: 1-32000, 单位: dot)。

char line

[in] 注释行。

'N': 不打印

'Y': 打印

char lineAboveCode

[in] 条码上方的注释行。

'N': 不打印在条码上方

'Y': 打印在条码上方

char checkDigit

[in] UCC 校验位。

'N': 不打印校验位

'Y': 打印校验位

char mode

[in] 模式。

'N': 不选择模式

'U': UCC 匹配模式

'A': 自动模式

'D': UCC/ EAN 模式

const TCHAR text*

[in] 条码数据。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.17. ZPL_CodeEan13

此函数功能为打印 CodeEan13条码。

```
int ZPL_CodeEan13(
    void* handle,
    int xPos,
    int yPos,
    int orientation,
    int moduleWidth,
    int codeHeight,
    char line,
    char lineAboveCode,
    const TCHAR* text
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围：0-32000，单位：dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int orientation

[in] 打印方向。

0：正常

90：顺时针旋转90度

180：顺时针旋转180度

270：顺时针旋转270度

int moduleWidth

[in] 条码宽度 (范围：0-10，单位：dot)。

int codeHeight

[in] 条码高度 (范围：1-32000，单位：dot)。

char line

[in] 注释行。

‘N’：不打印

‘Y’：打印

char lineAboveCode

[in] 条码上方的注释行。

‘N’：不打印在条码上方

‘Y’：打印在条码上方

const TCHAR text*

[in] 条码数据。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.18. ZPL_MicroPdf417

此函数功能为打印 MicroPdf417码。

```
int ZPL_MicroPdf417(  
    void* handle,  
    int xPos,  
    int yPos,  
    int orientation,  
    int moduleWidth,  
    int codeHeight,  
    int mode,  
    const TCHAR* text  
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围：0-32000，单位：dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int orientation

[in] 打印方向。

0：正常

90：顺时针旋转90度

180：顺时针旋转180度

270：顺时针旋转270度

int moduleWidth

[in] 条码宽度 (范围：0-10，单位：dot)。

int codeHeight

[in] 条码高度 (范围：1-32000，单位：dot)。

int mode

[in] 模式 (范围：0-33)。

Mode (M)	Number of Data Columns	Number of Data Rows	% of Cws for EC	Max Alpha Characters	Max Digits
0	1	11	64	6	8
1	1	14	50	12	17
2	1	17	41	18	26
3	1	20	40	22	32
4	1	24	33	30	44
5	1	28	29	38	55
6	2	8	50	14	20
7	2	11	41	24	35
8	2	14	32	36	52
9	2	17	29	46	67
10	2	20	28	56	82
11	2	23	28	64	93
12	2	26	29	72	105
13	3	6	67	10	14
14	3	8	58	18	26
15	3	10	53	26	38
16	3	12	50	34	49
17	3	15	47	46	67
18	3	20	43	66	96
19	3	26	41	90	132
20	3	32	40	114	167
21	3	38	39	138	202
22	3	44	38	162	237
23	4	6	50	22	32
24	4	8	44	34	49
25	4	10	40	46	67
26	4	12	38	58	85
27	4	15	35	76	111
28	4	20	33	106	155
29	4	26	31	142	208
30	4	32	30	178	261
31	4	38	29	214	313
32	4	44	28	250	366
33	4	4	50	14	20

`const TCHAR* text`
[in] 条码数据。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.19. ZPL_QRCode

此函数功能为打印二维码。

```
int ZPL_QRCode(
    void* handle,
    int xPos,
    int yPos,
    int orientation,
    int model,
    int dpi,
    char eccLevel,
    char input,
    char charMode,
    const TCHAR* text
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围: 0-32000, 单位: dot)。

int yPos

[in] 垂直起始位置 (范围: 0-32000, 单位: dot)。

int orientation

[in] 打印方向。

0 : 正常

90 : 顺时针旋转90度

180 : 顺时针旋转180度

270 : 顺时针旋转270度

int model

[in] 设置二维码版本 (1:原始版,2:强化版)。

int dpi

[in] 放大系数 (范围: 1-10)。

char eccLevel

[in] 纠错级别。

H:超高可靠性

Q:高可靠性

M:标准水平

L:高密度水平

char input

[in] 输入模式。

A:自动输入

M:手动输入

char charMode

[in] 数据类型。

N:数字

A:字母数字

B:8位字节模式

K: Kanji — 仅将 Kanji 字符按照基于 JIS X 0208的 ShiftJIS 系统处理。这意味着字符模式 K 之后的所有参数都应16位字符。如果出现8位字符 (如 ASCII 代码), 则会发生错误。

const TCHAR text*

[in] 二维码数据。仅当 charMode 是 B 时, 数据最前面四位应为数据大小, 例如数据为 qrcode 时传0006qrcode。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.20. ZPL_UpcExtensions

此函数功能为打印 UPC 扩展条码。

```
int ZPL_UpcExtensions(  
    void* handle,  
    int xPos,  
    int yPos,  
    int orientation,  
    int moduleWidth,  
    int codeHeight,  
    char line,  
    char lineAboveCode,  
    const TCHAR* text  
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围：0-32000，单位：dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int orientation

[in] 打印方向。

0：正常

90：顺时针旋转90度

180：顺时针旋转180度

270：顺时针旋转270度

int moduleWidth

[in] 条码宽度 (范围：0-10，单位：dot)。

int codeHeight

[in] 条码高度 (范围：1-32000，单位：dot)。

char line

[in] 注释行。

‘N’：不打印

‘Y’：打印

char lineAboveCode

[in] 条码上方的注释行。

‘N’：不打印在条码上方

‘Y’：打印在条码上方

const TCHAR text*

[in] 条码数据。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.21. ZPL_UpcaBarcode

此函数功能为打印 UPC-A 条码。

```
int ZPL_UpcaBarcode(  
    void* handle,  
    int xPos,  
    int yPos,  
    int orientation,  
    int moduleWidth,  
    int codeHeight,  
    char line,  
    char lineAboveCode,  
    char digit,  
    const TCHAR* text  
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围：0-32000，单位：dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int orientation

[in] 打印方向。

0：正常

90：顺时针旋转90度

180：顺时针旋转180度

270：顺时针旋转270度

int moduleWidth

[in] 条码宽度 (范围：0-10，单位：dot)。

int codeHeight

[in] 条码高度 (范围：1-32000，单位：dot)。

char line

[in] 注释行。

‘N’：不打印

‘Y’：打印

char lineAboveCode

[in] 条码上方的注释行。

‘N’：不打印在条码上方

‘Y’：打印在条码上方

char digit

[in] 校验位

‘N’：不打印校验位

‘Y’：打印校验位

const TCHAR text*

[in] 条码数据。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.22. ZPL_SetChangeFontEncoding

此函数功能为选择国际字符集。

```
int ZPL_SetChangeFontEncoding(
    void* handle,
    int encodeType
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int encodeType

[in] 字符集类型 (范围：0-31, 33-36)。

0 : 单字节编码 - 美国1字符集

1 : 单字节编码 - 美国2字符集

2 : 单字节编码 - 英国字符集

3 : 单字节编码 - 荷兰字符集

4 : 单字节编码 - 丹麦/挪威字符集

5 : 单字节编码 - 瑞典/芬兰字符集

6 : 单字节编码 - 德国字符集

7 : 单字节编码 - 法国1字符集

8 : 单字节编码 - 法国2字符集

9 : 单字节编码 - 意大利字符集

10 : 单字节编码 - 西班牙字符集

11 : 单字节编码 - 杂项字符集

12 : 单字节编码 - 日本字符组

13 : 代码页850

14 : 双字节亚洲编码

15 : Shift-JIS

16 : EUC-JP 和 EUC-CN

17 : 不推荐使用 - UCS-2 Big Endian

18-23 : 保留

24 : 单字节亚洲编码

25 : 保留

26 : 多字节亚洲编码

27 : 代码页1252

28 : Unicode (UTF-8编码) - Unicode 字符集

29 : Unicode (UTF- 16 Big- Endian 编码) - Unicode 字符集

30 : Unicode (UTF- 16 Little-Endian 编码) - Unicode 字符集

31 : 代码页1250

- 32 : 越南字符集
- 33 : 代码页 1251
- 34 : 代码页 1253
- 35 : 代码页 1254
- 36 : 代码页 1255

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.23. ZPL_SetChangeCaret

此函数功能为更改格式命令前缀。

```
int ZPL_SetChangeCaret(
    void* handle,
    char character
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char character

[in] 格式命令前缀。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.24. ZPL_SetChangeDelimiter

此函数功能为更改分隔符。

```
int ZPL_SetChangeDelimiter(
    void* handle,
```

```
char charactor
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char charactor

[in] 分隔符。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.25. ZPL_SetChangeTilde

此函数功能为更改控制命令前缀。

```
int ZPL_SetChangeTilde(
    void* handle,
    char charactor
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char charactor

[in] 控制命令前缀。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.26. ZPL_GraphicBox

此函数功能为绘制图形框。

```
int ZPL_GraphicBox(
    void* handle,
    int xPos,
    int yPos,
    int width,
    int height,
    int thickness,
    int rounding
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围: 0-32000, 单位: dot)。

int yPos

[in] 垂直起始位置 (范围: 0-32000, 单位: dot)。

int width

[in] 框的宽度 (范围: 1-32000, 单位: dot)。

int height

[in] 框的高度 (范围: 1-32000, 单位: dot)。

int thickness

[in] 边界厚度 (范围: 1-32000, 单位: dot)。

int rounding

[in] 转角程度 (范围: 0-8)。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.27. ZPL_GraphicCircle

此函数功能为绘制图形圆圈。

```
int ZPL_GraphicCircle(
    void* handle,
    int xPos,
    int yPos,
    int diameter,
    int thickness
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围：0-32000，单位：dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int diameter

[in] 圆的直径 (范围：3-4095，单位：dot)。

int thickness

[in] 边界厚度 (范围：1-4095，单位：dot)。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.28. ZPL_GraphicDiagonalLine

此函数功能为绘制对角线。

int ZPL_GraphicDiagonalLine(

void handle*,
int xPos,
int yPos,
int orientation,
int width,
int height,
int thickness

);

参数：

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围：0-32000，单位：dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int orientation

[in] 对角线的方向。

R (或/)：右倾斜的对角线

L (或\)：左倾斜的对角线

int width

[in] 框的宽度 (范围：1-32000，单位：dot)。

int height

[in] 框的高度 (范围：1-32000，单位：dot)。

int thickness

[in] 边界厚度 (范围：1-32000，单位：dot)。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.29. ZPL_GraphicEllipse

此函数功能为绘制图形椭圆。

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

参数：

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围：0-32000，单位：dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int width

[in] 椭圆宽度 (范围：3-4095，单位：dot)。

int height

[in] 椭圆高度 (范围：3-4095，单位：dot)。

int thickness

[in] 边界厚度 (范围：2-4095，单位：dot)。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.30. ZPL_PrintImage

此函数功能为打印图片 (仅支持单色bmp格式)。

```
int ZPL_PrintImage(  
    void* handle,  
    int xPos,  
    int yPos,  
    const TCHAR* imgName  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围: 0-32000, 单位: dot)。

int yPos

[in] 垂直起始位置 (范围: 0-32000, 单位: dot)。

const TCHAR imgName*

[in] 图片的路径。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.31. ZPL_GraphicSymbol

此函数功能为生成注册商标, 版权符号和其他符号。

```
int ZPL_GraphicSymbol(  
    void* handle,  
    int xPos,  
    int yPos,  
    int orientation,  
    int width,  
    int height,  
    const char symbol  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围: 0-32000, 单位: dot)。

int yPos

[in] 垂直起始位置 (范围：0-32000，单位：dot)。

int orientation

[in] 打印方向。

0：正常

90：顺时针旋转90度

180：顺时针旋转180度

270：顺时针旋转270度

int width

[in] 符号宽度。

int height

[in] 符号高度。

const char symbol

[in] 数据字符串。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.32. ZPL_SetDiagnosticsMode

此函数功能为启动诊断模式。

```
int ZPL_SetDiagnosticsMode(  
    void* handle,  
    int isEnabled  
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int isEnabled

[in] 是否开启诊断模式。

1：开启诊断模式

0：取消诊断模式

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败

ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.33. ZPL_SetLabelHome

此函数功能为设置标签首页位置。

```
int ZPL_SetLabelHome(
    void* handle
    int xPos,
    int yPos
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 横坐标起始位置 (范围: 0-32000, 单位: dot)。

int yPos

[in] 纵坐标起始位置 (范围: 0-32000, 单位: dot)。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.34. ZPL_SetLabelLength

此函数功能为设置标签长度。

```
int ZPL_SetLabelLength(
    void* handle,
    int length
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int length

[in] 标签长度 (范围: 1-32000, 单位: dot)。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.35. ZPL_SetLabelShift

此函数功能为将标签内容向左移动。

```
int ZPL_SetLabelShift(
    void* handle,
    int shift
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int shift

[in] 向左移动的值 (范围：-9999~9999，单位：dot)。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.36. ZPL_SetLabelTop

此函数功能为相对于标签的上边缘，将标签的位置向上或向下短距离移动。

```
int ZPL_SetLabelTop(
    void* handle,
    int top
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int top

[in] 最大化程度 (范围： -120 ~120 ， 单位： dot) 。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.37. ZPL_SetPrintMode

此函数功能为设置打印标签或标签组之后打印机执行的操作。

```
int ZPL_SetPrintMode(  
    void* handle,  
    char mode,  
    char prePeelSelect  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char mode

[in] 操作模式。

‘T’ : 撕开

‘P’ : 剥离 (取决于打印机型号)

‘R’ : 倒带 (取决于打印机型号)

‘A’ : 涂抹器 (取决于打印机型号)

‘C’ : 切刀 (取决于打印机型号)

‘D’ : 切刀延迟

‘F’ : RFID

‘L’ : 保留

‘U’ : 保留

‘K’ : Kiosk

char prePeelSelect

[in] 选择。

‘N’ : 不执行

‘Y’ : 执行

返回值:

ERROR_CM_SUCCESS, 成功

ERROR_CM_INVALID_HANDLE, 失败，句柄无效

ERROR_IO_OPEN_FAILED, 打开IO失败

4.38. ZPL_SetMediaType

此函数功能为选择在打印机中使用的媒介类型。

```
int ZPL_SetMediaType(  
    void* handle,  
    char type  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char type

[in] 媒介类型。

'T':碳带

'D':热敏

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.39. ZPL_SetPrintingMirrorImage

此函数功能为将标签的整个可打印区域打印为镜像图片。

```
int ZPL_SetPrintingMirrorImage(  
    void* handle,  
    char enable  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char enable

[in] 是否开启。

'N':不开启

'Y':开启

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效

ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.40. ZPL_SetPrintOrientation

此函数功能为将标签格式180度翻转打印。

```
int ZPL_SetPrintOrientation(
    void* handle,
    int orientation
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

Int orientation

[in] 是否翻转。

0:不翻转

180:执行翻转

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.41. ZPL_SetPrintRate

此函数功能为设置打印速度。

```
int ZPL_SetPrintRate(
    void* handle,
    int printSpeed,
    int slewSpeed,
    int backfeedSpeed
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int printSpeed

[in] 打印速度。单位为 inches/sec
int slewSpeed
 [in] 回转速度。单位为 inches/sec
int backfeedSpeed
 [in] 反馈速度。单位为 inches/sec

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.42. ZPL_SetPrintWidth

此函数功能为设置打印宽度。

```
int ZPL_SetPrintWidth(
    void* handle,
    int width
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int width

[in] 设置打印宽度 (范围: 2-944 , 单位: dot) 。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.43. ZPL_SetSerialCommunications

此函数功能为更改串行通信参数。

```
int ZPL_SetSerialCommunications(
    void* handle,
    int baudRate,
```

```

    int wordLength,
    char parity,
    int stopBits,
    char protocolModo
);

```

参数：

void handle*

[in] 创建的目标打印机对象。

int baudRate

[in] 带宽频率。范围如下：

110	300	600	1200	2400
4800	9600	14400	19200	28800
38400	57600	115200		

int wordLength

[in] 字长 (范围：7-8，单位：data bits)。

char parity

[in] 如下：

‘N’：表示：无。

‘E’：表示：偶校验。

‘O’：表示：奇校验。

int stopBits

[in] 范围：1-2。

char protocolModo

[in] 如下：

‘X’：表示：XON/XOFF。

‘D’：表示：DTR/DSR。

‘R’：表示：RTS。

‘M’：表示：DTR/DSR XON/XOFF r。

备注：1、XON/XOFF (继续传输/停止传输) 是一种流量控制协议

2、DTR (数据终端准备好)

3、DSR (数据准备好)

4、RTS (请求发送)

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.44. ZPL_SetPrintDarkness

此函数功能为设置打印的浓度。

```

int ZPL_SetPrintDarkness (
    void* handle,

```

```

        int darkness
    );

```

参数:

void handle*

[in] 创建的目标打印机对象。

int darkness

[in] 打印浓度(范围: 0-30 , 单位: dot)

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.45. ZPL_SetTearOffAdjustPosition

此函数功能为设置标签撕离的位置。

```

int ZPL_SetTearOffAdjustPosition (
    void* handle,
    int position
);

```

参数:

void handle*

[in] 创建的目标打印机对象。

int position

[in] 撕离位置(范围: - 120~+ 120)

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.46. ZPL_PrintConfigurationLabel

此函数功能为生成打印机配置标签。

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

参数:

void handle*

[in] 创建的目标打印机对象。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.47. ZPL_GetPrinterIpAddress

此函数功能为获取打印机 IP 地址。

```
int ZPL_GetPrinterIpAddress(
    void* handle
    char* ipAddress
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char ipAddress*

[in] 打印机的 IP 地址。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_READ_FAILED	-11	读取失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.48. ZPL_GetPrinterStatus

此函数功能为获取打印机当前状态。

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

参数:

void handle*

[in] 创建的目标打印机对象。

*int * status*

[in,out]打印机当前状态。

状态	十进制值	位
正常	0	-
打印头被打开	1	0
卡纸	2	1
缺纸	4	2
缺碳带	8	3
打印暂停	16	4
打印中	32	5
上盖打开	64	6
其它错误	128	7

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_READ_FAILED	-11	读取失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.49. ZPL_GetLabelLength

此函数功能为获取标签的长度。

```
int ZPL_GetLabelLength (  
    void* handle,  
    char* length  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char length*

[in] 标签的长度。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_READ_FAILED	-11	读取失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.50. ZPL_GetLabelWidth

此函数功能为获取标签的宽度。

```
int ZPL_GetLabelWidth(  
    void* handle,  
    char* width  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char width*

[in] 标签的宽度。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.51. ZPL_GetPrinterSeriesNumber

此函数功能为获取打印机序列号。

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

参数:

void handle*

[in] 创建的目标打印机对象。

char sn*

[in] 打印机序列号。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_READ_FAILED	-11	读取失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.52. ZPL_GetPrinterMacAddress

此函数功能为获取打印机的 MAC 地址。

```
int ZPL_GetPrinterMacAddress(  
    void* handle,  
    char* macAddress  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char macAddress*

[in] 打印机 MAC 地址。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_READ_FAILED	-11	读取失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.53. ZPL_GetPrinterName

此函数功能为获取打印机的名称。

```
int ZPL_GetPrinterName(  
    void* handle,  
    char* name  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char name*

[in] 打印机的名称。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_READ_FAILED	-11	读取失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.54. ZPL_GetPrinterFirmwareVersion

此函数功能为获取打印机的固件版本号。

```
int ZPL_GetPrinterFirmwareVersion(  
    void* handle,  
    char* version  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char version*

[in] 打印机的固件版本号。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.55. ZPL_GetPrinterDpi

此函数功能为获取打印机的分辨率。

```
int ZPL_GetPrinterDpi(  
    void* handle,
```

```
char* dpi
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char dpi*

[in] 打印机的分辨率。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_READ_FAILED	-11	读取失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.56. ZPL_GetPrinterModel

此函数功能为获取打印机的型号。

```
int ZPL_GetPrinterModel(
    void* handle,
    char* model
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char model*

[in] 打印机的型号。

示例:

```
char model[100] = { 0 };
ZPL_GetPrinterModel(printer, model);
printf("printer model is:%s\n", model);
```

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_READ_FAILED	-11	读取失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.57. ZPL_LearnLabel

此函数功能为标签自动学习。

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

参数:

void handle*

[in] 创建的目标打印机对象。

(此函数需要在 ZPL_StartFormat 之前调用)

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.58. ZPL_SetReprintAfterError

此函数功能为重新打印因发生错误而未能打印的标签 (错误情况包括Ribbon Out， Media Out， Head Open 三种)。

```
int ZPL_SetReprintAfterError(  
    void* handle,  
    char *pEnable  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char pEnable*

[in] 是否开启重印。

“on”: 开启重印

“of”: 不开启重印

(接口需在ZPL_StartFormat 之前调用)

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.59. ZPL_SetMediaTracking

此函数功能为指定使用的介质类型和黑标偏移量。

```
int ZPL_SetMediaTracking(  
    void* handle,  
    char mediaType,  
    int offset  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

char mediaType

[in] 介质类型。

'N': 连续介质 (连续纸)

'Y': 非连续介质网眼感应 (标签纸)

'W': 非连续介质网眼感应 (标签纸)

'M': 非连续介质标记感应 (黑标纸)

'A': 在校准过程中自动检测介质类型

'V': 连续介质, 可变长度 (与连续介质相同, 但如果打印标签的部分超出定义的 标签长度, 标签尺寸将自动扩展以包含它们)

int offset

[in] 黑标偏移量 (未使用到, 设为0)。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.60. ZPL_SetUserFontName

此函数功能为用户自己设置字体, 用于文本打印

```
int ZPL_SetUserFontName (  
    void* handle,  
    const TCHAR* text,  
    char alias  
);
```

参数:

void handle*

[in] 创建的目标打印机对象.

const TCHAR text*

[in] 字体名称

char alias

[in]别名

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.61. ZPL_Text_Block

此函数功能为打印文本块,可自动换行。

```
int ZPL_Text_Block(  
    void* handle,  
    int xPos,  
    int yPos,  
    int fontNum,  
    int orientation,  
    int fontWidth,  
    int fontHeight,  
    int textblockWidth,  
    int textblockHeight,  
    const TCHAR* text  
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围: 0-32000 , 单位: dot) 。

int yPos

[in] 垂直起始位置 (范围: 0-32000 , 单位: dot) 。

int fontNum

[in] 字体。

0 : FONT 0 - 可缩放字体

1 : FONT A - 位图字体

2 : FONT B - 位图字体

3 : FONT D - 位图字体

4 : FONT E - 位图字体

5 : FONT F - 位图字体

6 : FONT G - 位图字体

7 : FONT H - 位图字体

8 : FONT P - 位图字体

9 : FONT Q - 位图字体

10 : FONT R - 位图字体

- 11 : FONT S - 位图字体
- 12 : FONT T - 位图字体
- 13 : FONT U - 位图字体
- 14 : FONT V - 位图字体

int orientation

[in] 打印方向。

0 : 正常

90 : 顺时针旋转90度

180 : 顺时针旋转180度

270 : 顺时针旋转270度

int fontWidth

[in] 字体宽度。

int fontHeight

[in] 字体高度。

字体	高 x 宽 (以点为单位)
A	9 x 5
B	11 x 7
D	18 x 10
E	28 x 15
F	26 x 13
G	60 x 40
H	21 x 13
GS	24 x 24
P	20 x 18
Q	28 x 24
R	35 x 31
S	40 x 35
T	48 x 42
U	59 x 53
V	80 x 71
O	15 x 12

int textblockWidth

[in] 文本块宽度。

int textblockHeight

[in] 文本块高度。

const TCHAR text*

[in] 文本数据。

备注：数据暂不支持中文

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.62. ZPL_SetPrintQuantity

此函数功能为控制要打印的标签数量，打印机暂停前打印的标签数量以及每个序列号的复制次数。

```
int ZPL_SetPrintQuantity(  
    void* handle,  
    int totalQuantity,  
    int pauseAndCutValue,  
    int replicatesOfEachSerialNumber,  
    char overridePauseCount  
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

int totalQuantity

[in] 要打印的标签总量(范围：大于等于 1)

int pauseAndCut Value

[in] 暂停和切纸值(范围:大于等于 0,0 表示不暂停)

int replicatesOfEachSerialNumber

[in] 每个序列号的副本数(范围：大于等于 0)

char overridePauseCount

[in] 切纸或暂停(N = 暂停,Y = 切纸)

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.63. ZPL_DataMatrixBarcode

此函数功能为打印 Data Matrix 条码。

```
int ZPL_DataMatrixBarcode(  
    void* handle,  
    int xPos,  
    int yPos,  
    int orientation,  
    int codeHeight,  
    int level,  
    int columns,  
    int rows,  
    int formatId,  
    Int aspectRatio,  
    const TCHAR* text  
);
```

);

参数:

void handle*

[in] 创建的目标打印机对象。

int xPos

[in] 水平起始位置 (范围: 0-32000, 单位: dot)。

int yPos

[in] 垂直起始位置 (范围: 0-32000, 单位: dot)。

int orientation

[in] 打印方向。

0 : 正常

90 : 顺时针旋转90度

180 : 顺时针旋转180度

270 : 顺时针旋转270度

int codeHeight

[in] 条码高度 (范围: 1-32000, 单位: dot)。

int level

[in] 安全级别 (0、50、80、100、140、200)。

int column

[in] 要编码的列数。

int rows

[in] 要编码的行数。

Int formatId

[in] 格式 id (0-6)。

1 = 字段数据为数字 空格 (、) 无

2 = 字段数据为大写字母数字 空格 (、) 无

3 = 字段数据为大写字母数字 空格、句号、逗号、虚线和斜线(、 、)

4 = 字段数据为大写字母数字 空格 (、 、) 无

5 = 字段数据为完整 位字符集

6 = 字段数据为完整 位字符集

int aspectRatio

[in] 长宽比。

1 = 正方形

2 = 矩形

const TCHAR text*

[in] 条码数据。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.64. ZPL_GetPrinterOdometer

此函数功能为获取打印里程数。

```
int ZPL_GetPrinterOdometer(
    void* handle,
    char* meters
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

char meters*

[in] 打印里程数。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_READ_FAILED	-11	读取失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.65. ZPL_SetPrintNetSetting

此函数功能为设置网口信息。

```
int ZPL_SetPrintNetSetting(
    void* handle,
    const char* ipaddress,
    const char* mask,
    const char* gateway
);
```

参数：

void handle*

[in] 创建的目标打印机对象。

const char ipaddress*

[in] ip地址。格式为：xxx.xxx.xxx.xxx

const char mask*

[in] 子网掩码。格式为：xxx.xxx.xxx.xxx

const char gateway*

[in] 默认网关。格式为：xxx.xxx.xxx.xxx

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败

ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.66. ZPL_WifiConfig

此函数功能为设置wifi sta信息

```
int ZPL_WifiConfig(
    void* handle,
    int dhcp,
    const char* ipAddress,
    const char* mask,
    const char* gateway,
    const char* ssid,
    const char* password
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

int dhcp

[in] dhcp, (0:关闭, 1: 开启)

const char ipAddress*

[in] IP地址。格式为: xxx.xxx.xxx.xxx

const char mask*

[in] 子网掩码。格式为: xxx.xxx.xxx.xxx

const char gateway*

[in] 默认网关。格式为: xxx.xxx.xxx.xxx

const char ssid*

[in] WiFi ssid。

const char password*

[in] WiFi密码。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.67. ZPL_SetPrinterBluetoothSSID

此函数功能为设置蓝牙 SSID。

```
int ZPL_SetPrinterBluetoothSSID(
    void* handle,
```

```
const TCHAR* ssid
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

const TCHAR ssid*

[in] ssid 数据 (范围: 1-32)

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.68. ZPL_SetPrinterBluetoothPIN

此函数功能为设置蓝牙 pin 码。

```
int ZPL_SetPrinterBluetoothPIN(
    void* handle,
    const TCHAR* pin
);
```

参数:

void handle*

[in] 创建的目标打印机对象。

const TCHAR pin*

[in] pin 数据 (范围: 1-32)

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.69. ZPL_SetPrinterSleepTime

此函数功能为设置休眠时间。

```
int ZPL_SetPrinterSleepTime(
```

```

        void* handle,
        int time,
    );

```

参数：

void handle*

[in] 创建的目标打印机对象。

int time

[in] 休眠时间 (范围：0-10, 单位：分钟)

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.70. ZPL_SetPrinterShutdownTime

此函数功能为设置自动关机时间。

```

int ZPL_SetPrinterShutdownTime(
    void* handle,
    int time,
);

```

参数：

void handle*

[in] 创建的目标打印机对象。

int time

[in] 自动关机时间 (范围：0-63, 单位：分钟)

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.71. ZPL_FirmwareUpgrade

此函数功能为升级打印机固件

```

int ZPL_FirmwareUpgrade(

```

```

void* handle,
const TCHAR* cFileName,
void (*progressCallback)(float)
);

```

参数:

void handle*

[in] 创建的目标打印机对象。

const TCHAR cFileName*

[in] 固件文件路径

*void (*progressCallback)(float)*

更新进度回调

状态	值
更新进度	0~1
更新成功	0
内存不足	-4
读取文件失败	-11
发送数据失败	-9

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_READ_FAILED	-11	读取失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.72. ZPL_FontDownload

此函数功能为字库下载。

```

int ZPL_FontDownload(
void* handle,
const TCHAR* cFileName,
void (*progressCallback)(float)
);

```

参数:

void handle*

[in] 创建的目标打印机对象。

const TCHAR cFileName*

[in] 字库文件路径

*void (*progressCallback)(float)*

更新进度回调

状态	值
更新进度	0~1
更新成功	0

内存不足	-4
读取文件失败	-11
发送数据失败	-9

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Linux系统返回的错误码

4.73. ZPL_RfidCalibration

此函数功能为校正 RFID 应答器位置。

```
int ZPL_RfidCalibration(
    void* handle
)
```

参数:

void handle*

[in] 创建的目标打印机对象。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Windows系统返回的错误码

4.74. ZPL_RfidWrite

此函数功能为写入（编码）RFID 标签。

```
int ZPL_RfidWrite(
    void* handle,
    char format,
    int begin,
    int size,
    unsigned char memoryBlock,
    const TCHAR* text
)
```

参数:
void* handle
[in] 创建的目标打印机对象。
char format
[in] 格式: A = ASCII、H = 十六进制、E = EPC (需要先使用ZPL_RfidDefineEPC()定义EPC数据结构)
int begin
[in] 起始块编号
int size
[in] 写入的字节数
int memoryBlock
[in] 内存分段: E = EPC 96 位、0 = 保留、1 = EPC 2 = TID (标签 ID)、3 = 用户
const TCHAR* text
[in] 写入的文本数据

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Windows系统返回的错误码

4.75. ZPL_RfidDefineFont

此函数功能为定义打印rfid读取到的文本格式，结合ZPL_RfidRead一起使用。

```
int ZPL_RfidDefineFont(
    void* handle,
    int xPos,
    int yPos,
    int fontNum,
    int orientation,
    int fontWidth,
    int fontHeight
)
```

参数:
void* handle
[in] 创建的目标打印机对象。
int xPos
[in] 水平起始位置 (范围: 0-32000，单位: dot)。
int yPos
[in] 垂直起始位置 (范围: 0-32000，单位: dot)。
int fontNum
[in] 字体。
0: FONT 0 - 可缩放字体
1: FONT A - 位图字体
2: FONT B - 位图字体

3 : FONT D - 位图字体
 4 : FONT E - 位图字体
 5 : FONT F - 位图字体
 6 : FONT G - 位图字体
 7 : FONT H - 位图字体
 8 : FONT P - 位图字体
 9 : FONT Q - 位图字体
 10 : FONT R - 位图字体
 11 : FONT S - 位图字体
 12 : FONT T - 位图字体
 13 : FONT U - 位图字体
 14 : FONT V - 位图字体

int orientation

[in] 打印方向。

0 : 正常

90 : 顺时针旋转90度

180 : 顺时针旋转180度

270 : 顺时针旋转270度

int fontWidth

[in] 字体宽度。

int fontHeight

[in] 字体高度。

备注：当选择 FONT Z 时，宽高最小值为12*24，且只能倍增

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Windows系统返回的错误码

4.76. ZPL_RfidRead

此函数功能为读取（编码）RFID 标签。请结合ReadDataTimeout一起使用。

```
int ZPL_RfidRead(
    void* handle,
    char format,
    int begin,
    int size,
    unsigned char memoryBlock,
    int isPrint
)
```

参数：

void handle*

[in] 创建的目标打印机对象。

char format

[in] 格式：A = ASCII、H = 十六进制、E = EPC （需要先使用ZPL_RfidDefineEPC()定义EPC数据结构）

int begin

[in] 起始块编号

int size

[in] 写入的字节数

int memoryBlock

[in] 内存分段：E = EPC 96 位、0 = 保留、1 = EPC、2 = TID（标签 ID）、3 = 用户

int isPrint

[in] 读到的内容是否打印出来。1为打印 结合ZPL_RfidDefineFont一起使用，0为不打印。

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Windows系统返回的错误码

4.77. ZPL_RfidSetPower

此函数功能为设置 RFID 读取和写入功率级别

```
int ZPL_RfidSetPower(
    void* handle,
    unsigned char read,
    unsigned char write
)
```

参数：

void handle*

[in] 创建的目标打印机对象。

unsigned char read

[in] 读取功率，值：0 至 30

unsigned char write

[in] 写入功率，值：0 至 30

返回值：

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Windows系统返回的错误码

4.78. ZPL_RfidDefineEPC

此函数功能为定义 EPC 数据结构

```
int ZPL_RfidDefineEPC(  
    void* handle,  
    unsigned char* bits,  
    int count  
)
```

参数:

void handle*

[in] 创建的目标打印机对象。

unsigned char bits,*

[in] 分区大小集合。最大单个分区为64位,

int count

[in] 分区数量，最大16。

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败，句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Windows系统返回的错误码

4.79. ZPL_RfidSetParam

此函数功能为设置 RFID 参数

```
int ZPL_RfidSetParam(  
    void* handle,  
    unsigned char labelType,  
    int pos,  
    int len,  
    int number,  
    char err  
)
```

参数:

void handle*

[in] 创建的目标打印机对象。

unsigned char labelType

[in] 标签类型。接受值:8 = EPC Class 1, 第 2 代(Gen 2)

int pos

[in] 应答器的读取 / 写入位置（编程位置）

int len

[in] 无效打印输出的长度

int number

[in] 标签数量,读取 / 编码失败时尝试的标签数量。接受的值: 1 至 10
char err

[in] 错误处理,接受的值如下:

N = 不执行任何操作 (打印机放弃导致错误的标签格式, 并移至下一个队列标签)

P = 将打印机置于暂停模式 (标签格式将一直保留在队列中, 直到用户取消)

E = 将打印机置于出错模式 (标签格式将一直保留在队列中, 直到用户取消)

返回值:

错误代码	值	描述
ERROR_CM_SUCCESS	0	成功
ERROR_CM_INVALID_HANDLE	-2	失败, 句柄无效
ERROR_CM_INVALID_PARAMETER	-1	参数无效
ERROR_CM_INSUFFICIENT_MEMORY	-4	申请内存失败
ERROR_IO_WRITE_FAILED	-9	写入数据失败
ERROR_IO_WRITE_TIMEOUT	-10	写入数据超时
其他值	其他值	Windows系统返回的错误码