

# [ZPL Windows SDK]

[Printer ZPL Command Development Manual v2.0]

1. Information of the Manual .....	4
2. Operation System .....	4
3. Remark .....	4
4. Method .....	5
4.1 PrinterCreator .....	5
4.2 PrinterCreatorS .....	6
4.3 PrinterDestroy .....	7
4.4 PortOpen .....	8
4.5 PortClose .....	10
4.6 WriteData .....	11
4.7 ReadData .....	12
4.8 DirectIO .....	13
4.9 ZPL_StartFormat .....	15
4.10 ZPL_EndFormat .....	16
4.11 ZPL_ScalableFontText .....	17
4.12 ZPL_Text .....	19
4.13 ZPL_BarCode39 .....	22
4.14 ZPL_Pdf417 .....	24
4.15 ZPL_CodeEan8 .....	26
4.16 ZPL_UpceCode .....	28
4.17 ZPL_BarCode93 .....	30
4.18 ZPL_BarCode128 .....	32
4.19 ZPL_CodeEan13 .....	34
4.20 ZPL_MicroPdf417 .....	36
4.21 ZPL_QRCode .....	38
4.22 ZPL_UpcExtensions .....	40
4.23 ZPL_UpcaBarcode .....	42
4.24 ZPL_SetChangeFontEncoding .....	44
4.25 ZPL_SetChangeCaret .....	46
4.26 ZPL_SetChangeDelimiter .....	47
4.27 ZPL_SetChangeTilde .....	48
4.28 ZPL_GraphicBox .....	49
4.29 ZPL_GraphicCircle .....	51
4.30 ZPL_GraphicDiagonalLine .....	52
4.31 ZPL_GraphicEllipse .....	54
4.32 ZPL_PrintImage .....	56
4.33 ZPL_GraphicSymbol .....	57
4.34 ZPL_SetDiagnosticsMode .....	59
4.35 ZPL_SetLabelHome .....	60
4.36 ZPL_SetLabelLength .....	61
4.37 ZPL_SetLabelShift .....	62
4.38 ZPL_SetLabelTop .....	63
4.39 ZPL_SetPrintMode .....	64
4.40 ZPL_SetMediaType .....	65
4.41 ZPL_SetPrintingMirrorImage .....	66
4.42 ZPL_SetPrintOrientation .....	67
4.43 ZPL_SetPrintRate .....	68
4.44 ZPL_SetPrintWidth .....	69
4.45 ZPL_SetSerialCommunications .....	70
4.46 ZPL_SetPrintDarkness .....	72
4.47 ZPL_SetTearOffAdjustPosition .....	73
4.48 ZPL_PrintConfigurationLabel .....	74
4.49 ZPL_GetPrinterIpAddress .....	75
4.50 ZPL_GetPrinterStatus .....	76
4.51 ZPL_GetLabelLength .....	78
4.52 ZPL_GetLabelWidth .....	79

4.53 ZPL_GetPrinterSeriesNumber .....	80
4.54 ZPL_GetPrinterMacAddress .....	81
4.55 ZPL_GetPrinterName .....	82
4.56 ZPL_GetPrinterFirmwareVersion .....	83
4.57 ZPL_GetPrinterDpi .....	84
4.58 ZPL_LearnLabel .....	85
4.59 ZPL_SetReprintAfterError .....	86
4.60 ZPL_SetNetworkSetting .....	87
4.61 ZPL_SetMediaTracking .....	89
4.62 ZPL_SetUserFontName .....	90
4.63 ZPL_SetVietMode .....	91
4.64 ZPL_SetVietFontEncoding .....	92
4.65 ZPL_Text_Block .....	93
4.66 ZPL_RfidWrite .....	96
4.67 ZPL_RfidRead .....	98
4.68 ZPL_RfidCalibration .....	100
4.69 ZPL_SetPrintQuantity .....	101
4.70 ZPL_DataMatrixBarcode .....	103
4.71 ZPL_GetPrinterName .....	105
4.72 ZPL_GetPrinterSeriesNumber .....	106
4.73 ZPL_GetPrinterOdometer .....	107
4.74 ZPL_GetPrinterFonts .....	108
4.75 ZPL_SetPrinterInstruction .....	109
4.76 ZPL_SetPrinterNetMode .....	110
4.76 ZPL_SetPrinterNetSSID .....	111
4.77 ZPL_SetPrinterNetPwdSwitch .....	112
4.78 ZPL_SetPrinterNetPwd .....	113
4.79 ZPL_SetPrinterNetDHCP .....	114
4.80 ZPL_SetPrintIpAddress .....	115
4.81 ZPL_SetPrintSubnetMask .....	116
4.82 ZPL_SetPrintDefaultGateway .....	117
4.83 ZPL_SetPrinterBluetoothSSID .....	118
4.84 ZPL_SetPrinterBluetoothPIN .....	119
4.85 ZPL_SetPrinterSleepTime .....	120
4.86 ZPL_SetPrinterShutdownTime .....	121
4.87 ZPL_FirmwareUpgrade .....	122
4.88 ZPL_FontDownload .....	124
4.89 ZPL_VectorFontDownload .....	126
4.90 ZPL_RfidReturnHostDatalog .....	128
4.91 ZPL_RfidCorrectXpdrPosition .....	129
4.92 ZPL_RfidDefineDataStruct .....	131
4.93 ZPL_RfidRetryCount .....	133
4.94 ZPL_RfidSetParameters .....	134
4.95 ZPL_RfidSetPowerLevel .....	135
4.96 ZPL_RfidSetLockTagAndPassword .....	137
4.97 ZPL_RfidReadChipSerialization .....	139
4.98 DownloadFontFile .....	140
4.99 DownloadFMWImg .....	141
4.100 ZPL_RfidReadEmpty .....	142
4.101 ZPL_Cutter .....	143

## **1. Information of the Manual**

This SDK manual provides the dll file information for Windows application development.

We continuously promote and update the function and quality of all our products. Any change to the product specification and the manual will be without any further notice.

## **2. Operation System**

- Windows 2003/XP/7/8/10

## **3. Remark**

- When error code Return Value is greater than 0, it is the internal error of Windows system, please refer to related help file.

## 4. Method

### 4.1 PrinterCreator

Set up the target printer of specified model (should create target printer before using any function).

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

#### Parameter:

*void\* handle*

[in,out] The created target printer object.

*const TCHAR\* model*

[in] Specify the model of target printer.

#### Return Value:

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_INVALID_MODEL	-8	Invalid model name

## 4.2 PrinterCreatorS

Set up the target printer of specified model, the function is same to PrinterCreator (should create target printer before using any function).

**void\* PrinterCreatorS(**

**const TCHAR\* *model***

**);**

### **Parameter:**

*const TCHAR\* model*

[in] Specify the model of target printer

### **Return:**

Success : return the handle of printer object.

Fail: return NULL, invalid handle.

## 4.3 PrinterDestroy

Release the resource of specified model printer that has set up (after operation completed and no more operation for printer, it should release the printer that has set up).

**int PrinterDestroy(**

**void\* *handle***

**);**

### Parameter:

*void\* handle*

[in] The handle of target printer object which needs to release.

### Return Value:

Error Code	Value	Description
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle

## 4.4 PortOpen

Open the communication port and connect with the printer. After successfully connected, other functions can be used. If failed connecting, please check the error information.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*const TCHAR\* ioSettings*

[in] Set up the parameter of communication port that connected to the target printer. Please see as below:

### Configuration List:

Type	Configuration	Description	Sample
USB	<b>USB</b> [,Position/Model/PortNum]	USB: connect any USB printer of our company USB[,Position]: When connecting to multi printers of our company, can specify connecting to one particular USB printer through USB position information (Position parameter)	USB USB,Port_#0004.Hub_#0003 USB,LPG4 USB,USB001
NET	<b>NET</b> , IP Add (IPV4)[,Port]	Specify the IP add and port of internet printer. If not specifying port, the default port is 9100.	NET,192.168.0.36 NET,192.168.0.36,9100
COM	<b>COMn</b> ,BAUDRATE_rate	Specify the number and baud rate of connected serial port .	COM5,BAUDRATE_19200
LPT	<b>LPTn</b>	Specify the number of connected parallel port.	LPT1

Note: [ ] indicates selective parameter

\*If you connect to many different printers of our company at the same time, it is recommended to connect them by "USB,model".



**Return Value:**

Error Code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_OPEN_FAILED	-311	Port open failed

## 4.5 PortClose

This function is to close the communication port and disconnect with the printer.

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

### Parameter:

*void\* handle*  
[in,out] The created target printer object.

### Return Value:

Error Code	Value	Description
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle

## 4.6 WriteData

This function is to send data to the printer.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*unsigned char\* writeData*

[in] The data sent to the printer (hex string).

*unsigned int writeNum*

[in] The length of the data sent.

### Return Value:

Error Code	Value	Description
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout
E_IO_READ_FAILED	-331	Read failed
E_IO_READ_TIMEOUT	-332	Read timeout

## 4.7 ReadData

This function is to read the printer data.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*unsigned char\* readData*

[in] Printer data that needs to be read.

*unsigned int readNum*

[in] The length of data that needs to be read.

*unsigned int\* preadedNum*

[in] The length of the data actually read.

### Return Value:

Error Code	Value	Description
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_READ_FAILED	-331	Read failed
E_IO_READ_TIMEOUT	-332	Read timeout

## 4.8 DirectIO

This function is for the user to customize the data sent and read by the printer. If some functions do not provide a function interface, the user can send command data to the printer through this interface.

```
int DirectIO(  
  
    void* handle,  
  
    unsigned char* writedata,  
  
    unsigned int writeNum,  
  
    unsigned char* readdata,  
  
    unsigned int readNum,  
  
    unsigned int* preadedNum  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*unsigned char\* writedata*

[in] The data written to the printer.

*unsigned int writeNum*

[in] The length of the data written to the printer.

When writeNum=0, the write data operation is not performed.

*unsigned char\* readdata*

[in,out] Get the data returned by the printer.

*unsigned int readNum*

[in] Preset the length of data that needs to be read.

When readNum=0, the read data operation is not performed.

*unsigned int\* preadedNum*

[in,out] The length of the data actually read.

**Return Value:**

Error Code	Value	Description
E_SUCCESS	0	Normal
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout
E_IO_READ_FAILED	-331	Read failed
E_IO_READ_TIMEOUT	-332	Read timeout

## 4.9 ZPL\_StartFormat

This function is to indicate the beginning of a new label format.

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

### Parameter:

*void\* handle*  
[in,out] The created target printer object.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.10 ZPL\_EndFormat

This function is to indicate the end of a label format.

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

### Parameter:

*void\* handle*  
[in,out] The created target printer object.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.11 ZPL\_ScalableFontText

This function is to print scalable fonts.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*char fontName*

[in] Font(range: A-Z and 0-9).

*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int* *fontWidth*

[in] Font width.

*int* *fontHeight*

[in] Font height.

*char\** *text*

[in]Text data.

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.12 ZPL\_Text

This function is to print text.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int fontNum*

[in] Font.

- 0 : FONT 0 - Scalable font
- 1 : FONT A - Bitmap font
- 2 : FONT B - Bitmap font
- 3 : FONT D - Bitmap font
- 4 : FONT E - Bitmap font
- 5 : FONT F - Bitmap font
- 6 : FONT G - Bitmap font
- 7 : FONT H - Bitmap font
- 8 : FONT GS - Bitmap font
- 9 : FONT P - Bitmap font
- 10 : FONT Q - Bitmap font
- 11 : FONT R - Bitmap font
- 12 : FONT S - Bitmap font
- 13 : FONT T - Bitmap font
- 14 : FONT U - Bitmap font
- 15 : FONT V - Bitmap font
- 16 : SIMSUN.TTF – Song font
- 17 : FONT Z - Vietnam font

*int orientation*

[in] Print direction.

- 0 : normal
- 90 : Rotate 90 degrees clockwise
- 180 : Rotate 180 degrees clockwise
- 270 : Rotate 270 degrees clockwise

*int fontWidth*

[in] Font width.

*int fontHeight*

[in] Font height.

**Note:** When FONT Z is selected, the minimum width and height are 12\*24, and can only be multiplied.

*char\* text*

[in]Text data.

**FONT A** -- ABCDxyz 12345

**FONT B** -- ABCDWXYZ 12345 UPPER CASE ONLY

**FONT D** -- ABCDwxyz 12345

**FONT E** -- **(OCR-B)ABCDwxyz 12345**

**FONT F** -- ABCDwxyz 12345

**FONT G** -- **AByz 12**

**FONT H** -- **(OCR-A) UPPER CASE ONLY**

**FONT O** -- **(Scaleable) ABCDwxyz 12345**

**FONT GS** -- © ® ™ ®

**FONT P** -- ABCDwxyz 12345

**FONT Q** -- ABCDwxyz 12345

**FONT R** -- ABCDwxyz 12345

**FONT S** -- ABCDwxyz 12345

**FONT T** -- ABCDwxyz 12345

**FONT U** -- **ABCDwxyz 12345**

**FONT V** -- **ABCDwxyz 12345**

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.13 ZPL\_BarCode39

This function is to print Barcode39 barcodes.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int moduleWidth*

[in] Bar code width (range: 0-10,unit:dot).

*int codeHeight*

[in] Bar code height(range: 1-32000,unit:dot).

*char line*

[in] Comment line.

'N': not print

'Y': print

*char lineAboveCode*

[in] The comment line above the barcode.

'N': not print above the barcode

'Y': print above the barcode

*char digit*

[in] Check Digit.

'N': do not print check digit

'Y': print check digit

*char\* text*

[in] Text data.

#### **Return Value:**

<b>Error code</b>	<b>Value</b>	<b>Description</b>
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.14 ZPL\_Pdf417

This function is to print the Pdf417 code.

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

### Parameter:

*void\* handle*

[in,out]The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).



*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int moduleWidth*

[in] Bar code width (range: 0-10,unit:dot).

*int codeHeight*

[in] Bar code height(range: 1-32000,unit:dot).

*int securityLevel*

[in] Security level (range:1-8).

*int column*

[in] The number of columns to encode.

*int rows*

[in] The number of rows to encode.

*char truncate*

[in] Truncated layer indication and stop mode.

'N': not truncated

'Y': execution truncation

*char\* text*

[in] QR code data.

#### **Return value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.15 ZPL\_CodeEan8

This function is to print CodeEan8 barcodes.

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

### Parameter:

*void\* handle*

[in,out]The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int moduleWidth*

[in] Bar code width (range: 0-10,unit:dot).

*int codeHeight*

[in] Bar code height(range: 1-32000,unit:dot).

*char line*

[in] Comment line.

'N': not print

'Y': print

*char lineAboveCode*

[in] The comment line above the barcode.

'N': not print above the barcode

'Y': print above the barcode

*char\* text*

[in] Text data.

#### **Return Value:**

<b>Error code</b>	<b>Value</b>	<b>Description</b>
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.16 ZPL\_UpceCode

This function is to print UPC-E barcodes.

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

### Parameter:

*void\* handle*

[in,out]The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int moduleWidth*

[in] Bar code width (range: 0-10,unit:dot).

*int codeHeight*

[in] Bar code height(range: 1-32000,unit:dot).

*char line*

[in] Comment line.

'N': not print

'Y': print

*char lineAboveCode*

[in] The comment line above the barcode.

'N': not print above the barcode

'Y': print above the barcode

*char\* text*

[in] Text data.

#### **Return Value:**

<b>Error code</b>	<b>Value</b>	<b>Description</b>
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.17 ZPL\_BarCode93

This function is to print Barcode93 barcodes.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int moduleWidth*

[in] Bar code width (range: 0-10,unit:dot).

*int codeHeight*

[in] Bar code height(range: 1-32000,unit:dot).

*char line*

[in] Comment line.

'N': not print

'Y': print

*char lineAboveCode*

[in] The comment line above the barcode.

'N': not print above the barcode

'Y': print above the barcode

*char digit*

[in] Check Digit.

'N': do not print check digit

'Y': print check digit

*char\* text*

[in] Text data.

#### **Return Value:**

<b>Error code</b>	<b>Value</b>	<b>Description</b>
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.18 ZPL\_BarCode128

This function is to print Barcode128 barcodes.

```
int ZPL_BarCode128(  
    void* handle,  
    int xPos,  
    int yPos,  
    int orientation,  
    int moduleWidth,  
    int codeHeight,  
    char line,  
    char lineAboveCode,  
    char checkDigit,  
    char mode,  
    char* text  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).



*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int moduleWidth*

[in] Bar code width (range: 0-10,unit:dot).

*int codeHeight*

[in] Bar code height(range: 1-32000,unit:dot).

*char line*

[in] Comment line.

'N': not print

'Y': print

*char lineAboveCode*

[in] The comment line above the barcode.

'N': not print above the barcode

'Y': print above the barcode

*char checkDigit*

[in] UCC check Digit.

'N': do not print check digit

'Y': print check digit

*char mode*

[in] Mode.

'N': no choice mode

'U': UCC matching mode

'A': automatic mode

'D': UCC/EAN mode

*char\* text*

[in] Text data.

#### **Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.19 ZPL\_CodeEan13

This function is to print CodeEan13 barcodes.

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

### Parameter:

*void\* handle*

[in,out]The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int moduleWidth*

[in] Bar code width (range: 0-10,unit:dot).

*int codeHeight*

[in] Bar code height(range: 1-32000,unit:dot).

*char line*

[in] Comment line.

'N': not print

'Y': print

*char lineAboveCode*

[in] The comment line above the barcode.

'N': not print above the barcode

'Y': print above the barcode

*char\* text*

[in] Text data.

#### **Return Value:**

<b>Error code</b>	<b>Value</b>	<b>Description</b>
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.20 ZPL\_MicroPdf417

This function is to print MicroPdf417 codes.

**int ZPL\_MicroPdf417(**

**void\*** *handle*,

**int** *xPos*,

**int** *yPos*,

**int** *orientation*,

**int** *moduleWidth*,

**int** *codeHeight*,

**int** *mode*,

**char\*** *text*

**);**

### **Parameter:**

*void\* handle*

[in,out]The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int moduleWidth*

[in] Bar code width (range: 0-10,unit:dot).

*int codeHeight*

[in] Bar code height(range: 1-32000,unit:dot).

*int mode*

[in] Mode(range: 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

*char\* text*

[in] Text data.

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.21 ZPL\_QRCode

This function is to print a QR code.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int model*

[in] Set the QR code version (1: original version, 2: enhanced version).

*int dpi*

[in] Magnification factor (range: 1-20).

*char eccLevel*

[in] Error correction level.

H: Ultra high reliability

Q: High reliability

M: standard level

L: high density level

*char input*

[in] Input mode。

A:Automatic Input

M:Manual Input

*char charMode*

[in] character Mode。

N:Numeric

A:Alphanumeric

B:8-bit byte mode

K:Kanji ,handles only Kanji characters in accordance with the Shift JIS system based on JIS X 0208. This means that all parameters after the character mode K should be 16-bit characters. If there are any 8-bit characters (such as ASCII code), an error occurs.

*char\* text*

[in] data.Only when charMode is B, the first four digits of the data should be the data size, for example, if the data is qrcode, pass 0006qrcode.

#### **Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.22 ZPL\_UpcExtensions

This function is to print UPC extended barcodes.

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

### Parameter:

*void\* handle*

[in,out]The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise



*int moduleWidth*

[in] Bar code width (range: 0-10,unit:dot).

*int codeHeight*

[in] Bar code height(range: 1-32000,unit:dot).

*char line*

[in] Comment line.

'N': not print

'Y': print

*char lineAboveCode*

[in] The comment line above the barcode.

'N': not print above the barcode

'Y': print above the barcode

*char\* text*

[in] Text data.

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.23 ZPL\_UpcaBarcode

This function is to print UPC-A barcodes.

**int ZPL\_UpcaBarcode(**

**void\*** *handle*,

**int** *xPos*,

**int** *yPos*,

**int** *orientation*,

**int** *moduleWidth*,

**int** *codeHeight*,

**char** *line*,

**char** *lineAboveCode*,

**char** *digit*,

**char\*** *text*

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int moduleWidth*

[in] Bar code width (range: 0-10,unit:dot).

*int codeHeight*

[in] Bar code height(range: 1-32000,unit:dot).

*char line*

[in] Comment line.

'N': not print

'Y': print

*char lineAboveCode*

[in] The comment line above the barcode.

'N': not print above the barcode

'Y': print above the barcode

*char digit*

[in] Check Digit.

'N': do not print check digit

'Y': print check digit

*char\* text*

[in] Text data.

#### **Return Value:**

<b>Error code</b>	<b>Value</b>	<b>Description</b>
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.24 ZPL\_SetChangeFontEncoding

This function is to select an international character set.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int encodeType*

[in] Character set type (range: 0-31, 33-36).

- 0 : single byte encoding - US 1 character set
- 1 : Single-byte encoding - US 2 character set
- 2 : Single Byte Encoding - British Character Set
- 3 : Single Byte Encoding - Dutch Character Set
- 4 : Single-byte encoding - Danish/Norwegian character set
- 5 : Single-byte encoding - Swedish/Finnish character set
- 6 : Single byte encoding - German character set
- 7 : Single-byte encoding - French 1 character set
- 8 : Single-byte encoding - French 2 character set
- 9 : Single-byte encoding - Italian character set
- 10 : Single Byte Encoding - Spanish Character Set
- 11 : Single Byte Encoding - Miscellaneous Character Set
- 12 : Single-byte encoding - Japanese character set
- 13 : Code Page 850
- 14 : Double Byte Asian Code
- 15 : Shift-JIS
- 16 : EUC-JP and EUC-CN
- 17 : Not recommended - UCS-2 Big Endian
- 18-23 : Reserved
- 24 : Single Byte Asian Code
- 25 : Reserved
- 26 : Multibyte Asian Code

27 : Code Page 1252  
 28 : Unicode (UTF-8 encoding) - Unicode character set  
 29 : Unicode (UTF-16 Big-Endian encoding) - Unicode character set  
 30 : Unicode (UTF-16 Little-Endian encoding) - Unicode character set  
 31 : Code Page 1250  
 33 : Code page 1251  
 34 : Code page 1253  
 35 : Code page 1254  
 36 : Code page 1255  
 39 : Vietnam Character Set

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.25 ZPL\_SetChangeCaret

This function is to change the format command prefix.

```
int ZPL_SetChangeCaret(
```

```
    void* handle,
```

```
    char charactor
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char charactor*

[in] Format command prefix.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.26 ZPL\_SetChangeDelimiter

This function is to change the separator.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char charactor*

[in] Separator.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.27 ZPL\_SetChangeTilde

This function is to change the control command prefix.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char charactor*

[in] Control command prefix.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.28 ZPL\_GraphicBox

This function is to draw a graphic box.

**int ZPL\_GraphicBox(**

**void\*** *handle*,

**int** *xPos*,

**int** *yPos*,

**int** *width*,

**int** *height*,

**int** *thickness*,

**int** *rounding*,

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int width*

[in] The width of the box (range: 1-32000, unit: dot).

*int height*

[in] The height of the box (range: 1-32000, unit: dot).

*int thickness*

[in] Boundary thickness (range: 1-32000, unit: dot).

*int rounding*

[in] Degree of rotation (range: 0-8).

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.29 ZPL\_GraphicCircle

This function is to draw a graphic circle.

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

### Parameter:

*void\* handle*  
[in,out] The created target printer object.

*int xPos*  
[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*  
[in] Vertical starting position (range: 0-32000,unit:dot).

*int diameter*  
[in] Round diameter(range:3-4095,unit:dot).

*int thickness*  
[in] Boundary thickness(range:1-4095,unit:dot).

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.30 ZPL\_GraphicDiagonalLine

This function is to draw diagonals.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int orientation*

[in] The direction of the diagonal.

0x52(R or /) : right slanted diagonal

0x4c (L or \) : left slanted diagonal

*int width*

[in] The width of the box (range: 1-32000, unit: dot).

*int height*

[in] The height of the box (range: 1-32000, unit: dot).

*int thickness*

[in] Boundary thickness (range: 1-32000, unit: dot).

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.31 ZPL\_GraphicEllipse

This function is to draw a graphical ellipse.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int width*

[in] Ellipse width (range: 3-4095, unit: dot).

*int height*

[in] Ellipse height (range: 3-4095, unit: dot).

*int thickness*

[in] Boundary thickness (range: 2-4095, unit: dot).

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.32 ZPL\_PrintImage

This function is to print image.

```
int ZPL_PrintImage(
```

```
    void* handle,
```

```
    int xPos,
```

```
    int yPos,
```

```
    char* imgName
```

```
);
```

### parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*char\* imgName*

[in] The path to the image.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.33 ZPL\_GraphicSymbol

This function is to generate registered trademarks, copyright symbols and other symbols.

**int ZPL\_GraphicSymbol(**

**void\* *handle*,**

**int *xPos*,**

**int *yPos*,**

**int *orientation*,**

**int *width*,**

**int *height*,**

**char\* *type***

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int orientation*

[in] Print direction.

0 : normal

90 : Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int width*

[in] Symbol width.

*int height*

[in] Symbol height.

*char\* type*

[in] Data string.

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.34 ZPL\_SetDiagnosticsMode

This function is to start the diagnostic mode.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int isEnabled*

[in] Whether to enable the diagnostic mode.

- 1 : Turn on diagnostic mode
- 0 : Cancel diagnostic mode

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.35 ZPL\_SetLabelHome

This function is to set the label home position.

```
int ZPL_SetLabelHome(
```

```
    void* handle
```

```
    int xPos,
```

```
    int yPos
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.36 ZPL\_SetLabelLength

This function is to set the label length.

```
int ZPL_SetLabelLength(
```

```
    void* handle,
```

```
    int length
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int length*

[in] Label length (range: 1-32000, unit: dot).

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.37 ZPL\_SetLabelShift

This function is to move the contents of the label to the left.

**int ZPL\_SetLabelShift(**

**void\* *handle*,**

**int *shift***

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int shift*

[in] The value to move to the left (range: -9999–9999, unit: dot).

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.38 ZPL\_SetLabelTop

This function is to move the position of the label up or down a short distance relative to the top edge of the label.

**int ZPL\_SetLabelTop(**

**void\* *handle*,**

**int *top***

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int top*

[in] Maximum degree (range: -120–120, unit: dot).

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.39 ZPL\_SetPrintMode

This function is to set the action the printer performs after printing a label or label group.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char mode*

[in] Operating mode.

'T': tear open

'P': stripping (depending on the printer model)

'R': rewind (depending on the printer model)

'A': applicator (depending on printer model)

'C': cutter (depending on printer model)

'D': cutter delay

'F': RFID

'L': reserved

'U': reserved

'K': Kiosk

*char prePeelSelect*

[in] select.

'N': not execute

'Y': execute

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.40 ZPL\_SetMediaType

This function is to select the type of media used in the printer.

```
int ZPL_SetMediaType(
```

```
    void* handle,
```

```
    char type
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char type*

[in] Media type.

‘T’ : thermal transfer media

‘D’ : direct thermal media

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.41 ZPL\_SetPrintingMirrorImage

This function is to print the entire printable area of the label as a mirror image.

```
int ZPL_SetPrintingMirrorImage(
```

```
    void* handle,
```

```
    char enable
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char enable*

[in] Whether to open.

‘N’: not open

‘Y’: open

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.42 ZPL\_SetPrintOrientation

This function is to flip the label format 180 degrees.

**int ZPL\_SetPrintOrientation(**

**void\* *handle*,**

**int *orientation***

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int orientation*

[in] Whether to flip.

0: don't flip

180: perform a flip

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.43 ZPL\_SetPrintRate

This function is to set the print speed.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int printSpeed*

[in] Print speed. (unit: inches/sec)

*int slewSpeed*

[in] Swing speed. (unit: inches/sec)

*int backfeedSpeed*

[in] Feedback speed. (unit: inches/sec)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.44 ZPL\_SetPrintWidth

This function is to set print width.

**int ZPL\_SetPrintWidth(**

**void\* *handle*,**

**int *width***

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int width*

[in] Set the print width (range: 2-944, unit: dot).

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.45 ZPL\_SetSerialCommunications

This function is to change the serial communication parameters.

**int ZPL\_SetSerialCommunications(**

**void\*** *handle*,

**int** *baudRate*,

**int** *wordLength*,

**char** *parity*,

**int** *stopBits*,

**char** *protocolModo*,

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int baudRate*

[in] Bandwidth frequency. The scope is as follows:

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

*int wordLength*

[in] Word length: 7-8, unit: data bits.

*char parity*

[in] as follows:

'N': means: none.

'E': means: even.

'O': means: odd.

*int stopBits*

[in] Range: 1-2.

*char protocolModo*

[in] as follows:

'X': indicates: XON/XOFF.

'D': indicates: DTR/DSR.

'R': indicates: RTS.

'M': indicates: DTR/DSR XON/XOFF r.

remark: 1、XON/XOFF (transmitter on/transmitter off)

2、DTR (Data Terminal Ready)

3、DSR (Data Set Ready)

4、RTS (Request To Send)

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.46 ZPL\_SetPrintDarkness

This function is to set print darkness.

```
int ZPL_SetPrintDarkness (  
  
    void* handle,  
  
    int darkness  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int darkness*

[in] Print darkness(Range: 0-30, unit: dot).

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.47 ZPL\_SetTearOffAdjustPosition

This function is to set the position where the label is torn away.

**int ZPL\_SetTearOffAdjustPosition (**

**void\*** *handle*,

**int** *position*

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int position*

[in] Peel off position (range: -120~+120).

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.48 ZPL\_PrintConfigurationLabel

This function is to generate a printer configuration label.

**int ZPL\_PrintConfigurationLabel(**

**void\* *handle***

**);**

**Parameter:**

*void\* handle*

[in,out] The created target printer object.

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.49 ZPL\_GetPrinterIpAddress

This function is to get the printer IP address.

**int ZPL\_GetPrinterIpAddress(**

**void\*** *handle*

**char\*** *ipAddress*

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* ipAddress*

[in] Printer's IP address.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.50 ZPL\_GetPrinterStatus

This function is to get the status of the printer.

```
int ZPL_GetPrinterStatus (
```

```
    void* handle,
```

```
    int* status
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int\* status*

[in] The status of the printer.

HT/HD/XT/XD series models:

Status	Value
High Temperature	1
Standby	2
Printing	4
TOF Error	8
Label End	16
Ribbon End	32
Label Seizing	64
Label Jumpping	128
Label Calibrating	256
Cuter Error	512
Form Error	1024
Memory Write Error	2048
illegal Command	4096
Lid Not In Place	8192
Ribbon Almost End	16384

Other models:

Status	Value
Standby	0
Out Of Paper	1
Open The Lid	2
Time Out	4
High Temperature	8
Ribbon End	16

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.51 ZPL\_GetLabelLength

This function is to get the length of the label.( not available on HT130/300)

**int ZPL\_GetLabelLength (**

**void\* *handle*,**

**char\* *length***

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* length*

[in] The length of the label.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.52 ZPL\_GetLabelWidth

This function is to get the width of the label. ( not available on HT130/300)

```
int ZPL_GetLabelWidth(
```

```
    void* handle,
```

```
    char* width
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* width*

[in] The width of the label.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.53 ZPL\_GetPrinterSeriesNumber

This function is to get the printer serial number.

```
int ZPL_GetPrinterSeriesNumber(
```

```
    void* handle,
```

```
    char* sn
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* sn*

[in] Printer serial number.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.54 ZPL\_GetPrinterMacAddress

This function is to get the printer's MAC address.

```
int ZPL_GetPrinterMacAddress(
```

```
    void* handle,
```

```
    char* macAddress
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* macAddress*

[in] The MAC address of the printer.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.55 ZPL\_GetPrinterName

This function is to get the printer's name.

```
int ZPL_GetPrinterName(
```

```
    void* handle,
```

```
    char* name
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* name*

[in] The name of the printer.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.56 ZPL\_GetPrinterFirmwareVersion

This function is to get the firmware version number of the printer.

```
int ZPL_GetPrinterFirmwareVersion(
```

```
    void* handle,
```

```
    char* version
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* version*

[in] The firmware version number of the printer.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.57 ZPL\_GetPrinterDpi

This function is to get the resolution of the printer.

```
int ZPL_GetPrinterDpi(
```

```
    void* handle,
```

```
    char* dpi
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* dpi*

[in] The resolution of the printer.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.58 ZPL\_LearnLabel

This feature is used for automatic label learning.

```
int ZPL_LearnLabel(
```

```
    void* handle,
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

(The interface needs to be called before ZPL\_StartFormat)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.59 ZPL\_SetReprintAfterError

This function is to reprint the labels that failed to print due to an error (error conditions include Ribbon Out, Media Out, Head Open).

**int ZPL\_SetReprintAfterError(**

**void\* *handle*,**

**char\* *pEnable***

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*Char\* pEnable*

[in] Whether to enable reprint.

“on”: turn on reprint switch

“off”: close reprint switch

(The interface needs to be called before ZPL\_StartFormat)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.60 ZPL\_SetNetworkSetting

This function is to change the network settings on the printer.

**int ZPL\_SetNetworkSetting(**

**void\* handle,**

**char\* network**

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* network*

[in] **Format** "a,b,c,d,e,f,g,h,i,j"

a position (the device that is being modified):

1 express: external wired;

2 express: internal wired;

3 express: wireless;

b position (IP resolution):

A express: All;

B express: BOOTP;

C express: DHCP and BOOTP;

D express: DHCP;

G express: Gleaning only (Not recommended when the Wireless Print Server

or Wireless Plus Print Server is installed.);

R express: RARP;

P express: Permanent;

c position (IP address) : format: xxx.xxx.xxx.xxx

d position (subnet mask) : format: xxx.xxx.xxx.xxx

e position (default gateway) : format: xxx.xxx.xxx.xxx

f position (WINS server address) : format: xxx.xxx.xxx.xxx

g position (connection timeout checking) : Whether timeout detection: Y=yes, N=no

h position (timeout value) : range: 0-9999。

i position (ARP broadcast interval) :range: 0-30。

j position (ARP broadcast interval) :range: 1-65535。

Parameter setting example: "1, A, 192.168.1.1, 255.255.255.0, 192.168.1.1, 192.168.1.1, Y, 300, 0, 9100"

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.61 ZPL\_SetMediaTracking

This function is to specify the media type being used and the black mark offset.

**int ZPL\_SetMediaTracking(**

**void\*** *handle*,

**char** *mediaType*,

**int** *offset*

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char mediaType*

[in] Media Type.

‘N’: continuous media(continuous paper)

‘Y’: non-continuous media web sensing(label paper)

‘W’: non-continuous media web sensing(label paper)

‘M’: non-continuous media mark sensing(black mark paper)

‘A’: auto-detects the type of media during calibration

‘V’: continuous media, variable length(Same as continuum, but if the portion of the printed label exceeds the defined label length, the label size will automatically expand to include them)

*int offset*

[in] Black mark offset (unused, set to 0) .

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.62 ZPL\_SetUserFontName

This function is to Set user-defined fonts,use for print text

```
int ZPL_SetPrintDefaultGateway (  
  
    void* handle  
  
    const TCHAR* text  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*const TCHAR\* text*

[in] Font name

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.63 ZPL\_SetVietMode

This function is to Set Vietnamese mode

```
int ZPL_SetVietMode(  
  
    void* handle  
  
    int vietmode  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int vietmode*

[in] mode

1: ASCII

2: UTF-8

(The interface needs to be called before ZPL\_StartFormat)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.64 ZPL\_SetVietFontEncoding

This function is to set Vietnamese character

```
int ZPL_SetVietFontEncoding(
```

```
    void* handle
```

```
);
```

**Parameter:**

*void\* handle*

[in,out] The created target printer object.

(The interface needs to be called before ZPL\_StartFormat)

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.65 ZPL\_Text\_Block

This function is to print text block.

```
int ZPL_Text_Block(  
  
    void* handle,  
  
    int xPos,  
  
    int yPos,  
  
    int fontNum,  
  
    int orientation,  
  
    int fontWidth,  
  
    int fontHeight,  
  
    int textBlockWidth,  
  
    int textBlockHeight,  
  
    char* text  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000,unit:dot).

*int yPos*

[in] Vertical starting position (range: 0-32000,unit:dot).

*int fontNum*

[in] Font.

- 0 : FONT 0 - Scalable font
- 1 : FONT A - Bitmap font
- 2 : FONT B - Bitmap font
- 3 : FONT D - Bitmap font
- 4 : FONT E - Bitmap font
- 5 : FONT F - Bitmap font
- 6 : FONT G - Bitmap font
- 7 : FONT H - Bitmap font
- 8 : FONT GS - Bitmap font
- 9 : FONT P - Bitmap font
- 10 : FONT Q - Bitmap font
- 11 : FONT R - Bitmap font
- 12 : FONT S - Bitmap font
- 13 : FONT T - Bitmap font
- 14 : FONT U - Bitmap font
- 15 : FONT V - Bitmap font

FONT A -- ABCDwxyz 12345

FONT B -- ABCDHXYZ 12345 UPPER CASE ONLY

FONT D -- ABCDwxyz 12345

FONT E -- (OCR-B) ABCDwxyz 12345

FONT F -- ABCDwxyz 12345

FONT G -- **AByz 12**

FONT H -- (OCR-A) UPPER CASE ONLY

FONT O -- (Scaleable) ABCDwxyz 12345

FONT GS -- © ® ™ ®

FONT P -- ABCDwxyz 12345

FONT Q -- ABCDwxyz 12345

FONT R -- ABCDwxyz 12345

FONT S -- ABCDwxyz 12345

FONT T -- ABCDwxyz 12345

FONT U -- **ABCDwxyz 12345**

FONT V -- **ABCDwxyz 12345**

*int orientation*

[in] Print direction.

- 0 : normal
- 90 : Rotate 90 degrees clockwise
- 180 : Rotate 180 degrees clockwise
- 270 : Rotate 270 degrees clockwise

*int fontWidth*

[in] Font width.

*int fontHeight*

[in] Font height.

*int textBlockWidth*

[in] Text block width

*int textBlockHeight*

[in] Text block height

*char\* text*

[in]Text data.

Note: The data does not support Chinese at this time

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.66 ZPL\_RfidWrite

This function is used to write RFID data

**int ZPL\_RfidWrite(**

**void\* handle,**

**char format,**

**int begin,**

**int size,**

**char memoryBlock,**

**const TCHAR\* text**

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char format*

[in] format.

A = ASCII

H = Hexadecimal

E = EPC

*int begin*

[in] starting block number

*int size*

[in] Number of bytes to write

*char memoryBlock*

[in] memoryBlock

1:EPC

2:TID

3:user

A:EPC and Auto adjust PC bits (When writing data, this parameter performs the operation on Gen 2 bit address 20h of the EPC memory bank and accesses the number of bytes specified in the ^FD. The PC bits will be updated to match the amount of data written to the tag. When reading data, this parameter reads the amount of data specified in the PC bits on the tag.)



*const TCHAR\* text*  
*[in] Data to write*

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.67 ZPL\_RfidRead

This function is used to make RFID data readable (reading RFID data requires calling this interface and calling ReadData after ZPL\_EndFormat)

**int ZPL\_RfidRead(**

**void\*** *handle*,

*char format*,

*int begin*,

*int size*,

*char memoryBlock*,

*const TCHAR\** *headtext*,

*const TCHAR\** *tailText*

**);**

### Parameter:

*void\** *handle*

[in,out] The created target printer object.

*char format*

[in] *format*.

*A = ASCII*

*H = Hexadecimal*

*E = EPC*

*int begin*

[in] *starting block number*

*int size*

[in] *Number of bytes to read*

*char memoryBlock*

[in] *memoryBlock*

*1:EPC*

*2:TID*

*3:user*

*A:EPC and Auto adjust PC bits (When writing data, this parameter performs the operation on Gen 2 bit address 20h of the EPC memory bank and accesses the number of bytes specified in the ^FD. The PC*

*bits will be updated to match the amount of data written to the tag.  
When reading data, this parameter reads the amount of data specified  
in the PC bits on the tag.)*

*const TCHAR\* headtext*  
    [in] *headtext*  
*const TCHAR\* tailtext*  
    [in] *tailtext*

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.68 ZPL\_RfidCalibration

This function is used to calibrate RFID tags

```
int ZPL_RfidCalibration(
```

```
    void* handle
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.69 ZPL\_SetPrintQuantity

This function is to give control over several printing operations. It controls the number of labels to print, the number of labels printed before printer pauses, and the number of replications of each serial number.

### **int ZPL\_SetPrintQuantity(**

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

### **Parameter:**

*void\* handle*

[in,out] The created target printer object.

*int totalQuantity*

[in] total quantity of labels to print (range: greater or equal to 1).

*int pauseAndCutValue*

[in] pause and cut value (range: greater or equal to 0, 0 Means no pause).

*int replicatesOfEachSerialNumber*

[in] replicates of each. (range: greater or equal to 0).

*char overridePauseCount*

[in] Cut paper or pause.

'N': pause

'Y': Cut paper

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.70 ZPL\_DataMatrixBarcode

This function is to print Data Matrix.

```
int ZPL_DataMatrixBarcode(
```

```
    void* handle,
```

```
    int xPos,
```

```
    int yPos,
```

```
    int orientation,
```

```
    int codeHeight,
```

```
    int level,
```

```
    int columns,
```

```
    int rows,
```

```
    int formatId,
```

```
    int aspectRatio,
```

```
    char* text
```

```
);
```

**Parameter:**

*void\* handle*

[in,out] The created target printer object.

*int xPos*

[in] Horizontal starting position (range: 0-32000, unit: dot).

*int yPos*

[in] Vertical starting position (range: 0-32000, unit: dot).

*int orientation*

[in] Printing direction.

0: normal

90: Rotate 90 degrees clockwise

180: Rotate 180 degrees clockwise

270: Rotate 270 degrees clockwise

*int codeHeight*

[in] code height (range: 1-32000, unit: dot).

*int level*

[in] Security Level (0、50、80、100、140、200)。

*int column*

[in] The number of columns to be encoded.

*int rows*

[in] The number of lines to be encoded.

*Int formatId*

[in] Format id (0-6).

1 = Field data is number + space (0..9, "-")-no \& ' '

2 = Field data is uppercase alphanumeric + space (A..Z, ' ' ) - no \& ' '

3 = Field data is uppercase alphanumeric + space, period, comma,  
dotted line and slash(0..9, A..Z, "-./")

4 = The field data is uppercase alphanumeric + space (0..9, A..Z, ' ' ) -  
no \& ' '

5 = The field data is a complete 128 ASCII 7-bit character set

6 = The field data is a complete 256 ASCII 8-bit character set

*int aspectRatio*

[in] Aspect ratio.

1 = square

2 = rectangle

*char\* text*

[in] code data.

#### **Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.71 ZPL\_GetPrinterName

This function is to get the printer model.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* name*

[in] printer model。

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.72 ZPL\_GetPrinterSeriesNumber

This function is to get the printer serial number。

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* sn*

[in] printer serial number。

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.73 ZPL\_GetPrinterOdometer

This function is to get the number of printed mileage。

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* meters*

[in] printed mileage。

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.74 ZPL\_GetPrinterFonts

This function is to get the printer's built-in font.

```
int ZPL_GetPrinterFonts(  
  
    void* handle,  
  
    char* fonts  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* fonts*

[in] printer's built-in font, The format is E\_xxx。

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.75 ZPL\_SetPrinterInstruction

This function is to set the print instruction set.

```
int ZPL_SetPrinterInstruction(  
  
    void* handle,  
  
    int type  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int type*

[in] Instruction set type 0: ZPL, 1: cpcl

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.76 ZPL\_SetPrinterNetMode

The function of this function is to set the wifi mode.

```
int ZPL_SetPrinterNetMode(  
  
    void* handle,  
  
    int mode  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int mode*

[in] wifi mode。 (0:close, 1:sta, 2: ap)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.76 ZPL\_SetPrinterNetSSID

The function of this function is to set wifi SSID.

```
int ZPL_SetPrinterNetSSID(  
  
    void* handle,  
  
    int mode,  
  
    const TCHAR* ssid  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int mode*

[in] wifi mode。 (1:sta, 2: ap)

*const TCHAR\* ssid*

[in] ssid data (range: 1-32)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.77 ZPL\_SetPrinterNetPwdSwitch

The function of this function is to set wifi Password switch。

```
int ZPL_SetPrinterNetPwdSwitch(  
  
    void* handle,  
  
    int mode,  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int mode*

[in] wifi password switch。 (0: off, 1: on)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.78 ZPL\_SetPrinterNetPwd

This function is to set the wifi password.

```
int ZPL_SetPrinterNetPwd(  
  
    void* handle,  
  
    int mode,  
  
    const TCHAR* pwd  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int mode*

[in] wifi mode。 (1:sta, 2: ap)

*const TCHAR\* pwd*

[in] password (range: 1-64)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.79 ZPL\_SetPrinterNetDHCP

This function is to set wifi DHCP。

```
int ZPL_SetPrinterNetDHCP(  
  
    void* handle,  
  
    int mode,  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int mode*

[in] Whether to open (0: close, 1: open)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.80 ZPL\_SetPrintIpAddress

This function is to set the wifi IP address.

```
int ZPL_SetPrintIpAddress(  
  
    void* handle,  
  
    int mode,  
  
    const TCHAR* ipaddress  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int mode*

[in] wifi mode。 (0:off, 1: on)

const TCHAR\* ipaddress

[in] ip address。 The format is: xxx.xxx.xxx.xxx

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.81 ZPL\_SetPrintSubnetMask

This function is to set the wifi subnet mask.

```
int ZPL_SetPrintSubnetMask(  
  
    void* handle,  
  
    const TCHAR* mask  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*const TCHAR\* mask*

[in] subnet mask。The format is: xxx.xxx.xxx.xxx

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.82 ZPL\_SetPrintDefaultGateway

This function is to set the wifi default gateway.

```
int ZPL_SetPrintDefaultGateway(  
  
    void* handle,  
  
    const TCHAR* gateway  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*const TCHAR\* gateway*

[in] default gateway The format is: xxx.xxx.xxx.xxx

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.83 ZPL\_SetPrinterBluetoothSSID

This function is to set the Bluetooth SSID.

```
int ZPL_SetPrinterBluetoothSSID(  
  
    void* handle,  
  
    const TCHAR* ssid  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*const TCHAR\* ssid*

[in] ssid data (range: 1-32)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.84 ZPL\_SetPrinterBluetoothPIN

This function is to set the Bluetooth pin code

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*const TCHAR\* pin*

[in] pin data (range: 1-32)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.85 ZPL\_SetPrinterSleepTime

This function is to set the sleep time

```
int ZPL_SetPrinterSleepTime(  
  
    void* handle,  
  
    int time,  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int time*

[in] sleep time(range: 0-999, unit: minute)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.86 ZPL\_SetPrinterShutdownTime

This function is to set the automatic shutdown time.

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

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int time*

[in] Automatic shutdown time (range: 0-999, unit: minute)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.87 ZPL\_FirmwareUpgrade

This function is to upgrade the printer firmware and is only applicable to HM-T300 PRO. This interface needs to be called before PrinterCreator or after PrinterDestroy

```
int ZPL_FirmwareUpgrade(  
  
    void* handle,  
  
    const TCHAR* cFileName,  
  
    const TCHAR* model,  
  
    const TCHAR* ioSettings  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*const TCHAR\* cFileName*

[in] Firmware file

*const TCHAR\* model*

[in] model

*const TCHAR\* ioSettings*

[in] Set up the parameter of communication port that connected to the target printer. Please see as below:

### Configuration List:

Type	Configuration	Description	Sample
USB	<b>USB</b> [,Position/Model/PortNum]	USB: connect any USB printer of our company USB[,Position]: When connecting to multi printers of our company, can specify connecting to one particular USB printer through USB position information (Position parameter)	USB USB,Port_#0004.Hub_#0003 USB,LPG4 USB,USB001
NET	<b>NET</b> , IP Add (IPV4)[,Port]	Specify the IP add and port of internet printer. If not specifying	NET,192.168.0.36 NET,192.168.0.36,9100

		port, the default port is 9100.	
COM	<b>COM<math>n</math></b> ,BAUDRATE_ <i>rate</i>	Specify the number and baud rate of connected serial port .	COM5,BAUDRATE_19200
LPT	<b>LPT<math>n</math></b>	Specify the number of connected parallel port.	LPT1

Note: [ ] indicates selective parameter

#### Return Value:

Error code	Value	Description
E_SUCCESS	1	success
E_FAILED	0	failed
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.88 ZPL\_FontDownload

This function is a font download, only applicable to HM-T300 PRO. This interface needs to be called before PrinterCreator or after PrinterDestroy

```
int ZPL_FontDownload(  
  
    void* handle,  
  
    const TCHAR* cFileName,  
  
    const TCHAR* model,  
  
    const TCHAR* ioSettings  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*const TCHAR\* cFileName*

[in] Font file

*const TCHAR\* model*

[in] model

*const TCHAR\* ioSettings*

[in] Set up the parameter of communication port that connected to the target printer. Please see as below:

### Configuration List:

Type	Configuration	Description	Sample
USB	<b>USB</b> [,Position/Model/PortNum]	USB: connect any USB printer of our company USB[,Position]: When connecting to multi printers of our company, can specify connecting to one particular USB printer through USB position information (Position parameter)	USB USB,Port_#0004.Hub_#0003 USB,LPG4 USB,USB001
NET	<b>NET</b> , IP Add (IPV4)[,Port]	Specify the IP add and port of internet printer. If not specifying	NET,192.168.0.36 NET,192.168.0.36,9100

		port, the default port is 9100.	
COM	<b>COM<math>n</math></b> ,BAUDRATE_ <i>rate</i>	Specify the number and baud rate of connected serial port .	COM5,BAUDRATE_19200
LPT	<b>LPT<math>n</math></b>	Specify the number of connected parallel port.	LPT1

Note: [ ] indicates selective parameter

#### Return Value:

Error code	Value	Description
E_SUCCESS	1	success
E_FAILED	0	failed
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.89 ZPL\_VectorFontDownload

This function is a vector font download, only applicable to HM-T300 PRO.  
This interface needs to be called before PrinterCreator or after PrinterDestroy

```
int ZPL_VectorFontDownload(  
  
    void* handle,  
  
    const TCHAR* cFileName,  
  
    const TCHAR* model,  
  
    const TCHAR* ioSettings  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*const TCHAR\* cFileName*

[in] Vector font file

*const TCHAR\* model*

[in] model

*const TCHAR\* ioSettings*

[in] Set up the parameter of communication port that connected to the target printer. Please see as below:

### Configuration List:

Type	Configuration	Description	Sample
USB	<b>USB</b> [,Position/Model/PortNum]	USB: connect any USB printer of our company USB[,Position]: When connecting to multi printers of our company, can specify connecting to one particular USB printer through USB position information (Position parameter)	USB USB,Port_#0004.Hub_#0003 USB,LPG4 USB,USB001
NET	<b>NET</b> , IP Add (IPV4)[,Port]	Specify the IP add and port of internet printer. If not specifying	NET,192.168.0.36 NET,192.168.0.36,9100

		port, the default port is 9100.	
COM	<b>COM<math>n</math></b> ,BAUDRATE_ <i>rate</i>	Specify the number and baud rate of connected serial port .	COM5,BAUDRATE_19200
LPT	<b>LPT<math>n</math></b>	Specify the number of connected parallel port.	LPT1

Note: [ ] indicates selective parameter

#### Return Value:

Error code	Value	Description
E_SUCCESS	1	success
E_FAILED	0	failed
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.90 ZPL\_RfidReturnHostDatalog

This function returns the RFID data log to the host.

**int ZPL\_RfidReturnHostDatalog(**

**void\* *handle***

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.91 ZPL\_RfidCorrectXpdrPosition

This function is used to correct the RFID transponder position.

```
int ZPL_RfidCorrectXpdrPosition(  
  
    void* handle,  
  
    char* pStartStr,  
  
    char* pEndStr,  
  
    char* pStartPosition,  
  
    char* pEndPosition,  
  
    char model  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* pStartStr*

[in] Start string (range: less than 65 characters).

*char\* pEndStr*

[in] End string (range: less than 65 characters).

*char\* pStartPosition*

[in] Start position (forward range: F0 to Fxxx, backward range: B0 to B30).

*char\* pEndPosition*

[in] End position (forward range: F0 to Fxxx, backward range: B0 to B30).

*char model*

[in] Select the antenna and read/write power level.

A=Automatic. The printer will automatically scan the antenna and read/write power during the calibration process.

M=Manual. The printer uses the current antenna and read/write power level settings.

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.92 ZPL\_RfidDefineDataStruct

This function defines the EPC data structure.(When used with a read/write RFID data interface, the interface needs to be placed between ZPL\_StartFormat and ZPL\_EndFormat.)

```
int ZPL_RfidDefineDataStruct(  
  
    void* handle,  
  
    int nTotalNum,  
  
    int* pPartitionSize,  
  
    int nPartitionLenth,  
  
    );
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int nTotalNum*

[in] The total number of digits for the partition (range: 1 to n, where n is the number of digits for the label).

*int\* pPartitionSize*

[in] Array/pointer that stores the size of the partition (range: not NULL).

*int nPartitionLenth*

[in] The length of the partition size (range: length greater than or equal to 1, less than or equal to 64).

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open

E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.93 ZPL\_RfidRetryCount

This function provides the number of RFID retries for the specified block.(When used with a read/write RFID data interface, the interface needs to be placed between ZPL\_StartFormat and ZPL\_EndFormat.)

**int ZPL\_RfidRetryCount(**

**void\* handle,**

**int nRetryCount,**

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int nRetryCount*

[in] Number of retries (range: greater than or equal to 1, less than or equal to 10).

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_NOT_ENOUGH_BUFFER	-2	内存不足
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

## 4.94 ZPL\_RfidSetParameters

This function is used to set RFID parameters.(You need to place the interface between ZPL\_StartFormat and ZPL\_EndFormat.)

**int ZPL\_RfidSetParameters(**

**void\* *handle*,**

**int *nTagNum*,**

**int *nErrorAct*,**

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int nTagNum*

[in] Number of labels (range: greater than or equal to 1, less than or equal to 10).

*int nErrorAct*

[in] *Error handling*

N = Do nothing (the printer discards the label format that caused the error and moves to the next queue label)

P = Put the printer in pause mode (the label format will remain in the queue until the user cancels)

E = Place the printer in error mode (the label format will remain in the queue until cancelled by the user).

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.95 ZPL\_RfidSetPowerLevel

This function functions to set the RFID read and write power levels.

```
int ZPL_RfidSetPowerLevel(  
  
    void* handle,  
  
    int nReadPower,  
  
    int nWritePower,  
  
    int nAntennaType,  
  
    );
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int nReadPower*

[in] Read power

Range:

R53.16.3, V53.17.7Z, and later: greater than or equal to 0, less than or equal to 30.

R60.16.4, R62.16.4, R63.16.4, SP994Q, SP999G, SP1027G, SP1056F, SP1082G and later versions: H=High, M=Medium, L=Low.

R65. X and other earlier firmware versions: H=High, M=Medium, and L=Low.

*int nWritePower*

[in] Write power

Range:

R53.16.3, V53.17.7Z, and later: greater than or equal to 0, less than or equal to 30.

R60.16.4, R62.16.4, R63.16.4, SP994Q, SP999G, SP1027G, SP1056F, SP1082G and later versions: H=High, M=Medium, L=Low.

Earlier firmware: H=High, M=Medium, L=Low.

*int nAntennaType*

[in] RFID antenna element selection.

1 = Antenna Port 1

2 = Antenna Port 2

**Return Value:**

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout



## 4.96 ZPL\_RfidSetLockTagAndPassword

This function is used to set the RFID tag password and lock the tag.(When used with a read/write RFID data interface, the interface needs to be placed between ZPL\_StartFormat and ZPL\_EndFormat.)

**int ZPL\_RfidSetLockTagAndPassword(**

**void\*** *handle*,

**char\*** *password*,

**int** *nMemoryBlock*,

**char** *locktype*,

**);**

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*char\* password*

[in] Password. The password must be a 2-digit hexadecimal character (0x00-0xFF)。

*int nMemoryBlock*

[in] Memory blocking

K = Password cracking

A = Access password

E = EPC

T = Label identification (TID)

U = User

*char locktype*

[in] Lock the style. This parameter is used to specify the status of the RFID tag password.

U = Unlocked

L = Locked

O = Permanently unlock (open state)

P = Permanently locked (protected state)

W = Write value (only used to crack password memory segments)

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.97 ZPL\_RfidReadChipSerialization

This function reads the unique RFID chip family.(You need to place the interface between ZPL\_StartFormat and ZPL\_EndFormat.)

```
int ZPL_RfidReadChipSerialization(
```

```
    void* handle,
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.98 DownloadFontFile

This function is for the font library download function

```
int DownloadFontFile(  
  
    void* handle,  
  
    int iPacketSize,  
  
    char* filename,  
  
    char* desc,  
  
    void* pfnDownProgress  
  
    );
```

### Parameter:

*void\** handle  
[in,out] The created target printer object.

*int* iPacketSize  
[in] Send block size

*char\** filename  
[in] File name/path to download.

*char\** desc  
[in] Device descriptor.

*void\** pfnDownProgress  
[in] Callback progress function.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.99 DownloadFMWImg

The function function is to download and update firmware of Img type.

```
int DownloadFMWImg(  
  
    void* handle,  
  
    char* filename,  
  
    char* desc,  
  
    void* pfnDownProgress  
  
);
```

### Parameter:

*void\* handle*  
[in,out] The created target printer object.

*char\* filename*  
[in] File name/path to download.

*char\* desc*  
[in] Device descriptor.

*void\* pfnDownProgress*  
[in] Callback progress function.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.100 ZPL\_RfidReadEmpty

This function is used to read blank content from RFID.

```
int ZPL_RfidReadEmpty(
```

```
    void* handle,
```

```
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout

## 4.101 ZPL\_Cutter

This function is the full and half cutting function of the cutting tool. This function is associated with ZPL\_SetPrintMode, ZPL\_SetPrintQuantity combination use.

```
int ZPL_Cutter(  
  
    void* handle,  
  
    int partialCutReserveDistance  
  
);
```

### Parameter:

*void\* handle*

[in,out] The created target printer object.

*int partialCutReserveDistance*

[in] Indicates the number of millimeters left uncut in the medium.

partialCutReserveDistance = 0: Full cut function.

10<=partialCutReserveDistance<=60: Half cut function, with a value of retained uncut millimeters.

### Return Value:

Error code	Value	Description
E_SUCCESS	0	Normal
E_INVALID_PARAMETER	-1	Invalid parameter
E_NOT_ENOUGH_BUFFER	-2	No enough memory
E_INVALID_MODEL_TYPE	-3	This model does not support this feature.
E_BAD_HANDLE	-6	Invalid handle
E_IO_PORT_NOT_OPEN	-309	Communication port not open
E_IO_WRITE_FAILED	-321	Write failed
E_IO_WRITE_TIMEOUT	-322	Write timeout