

[ZPL Windows SDK]

[Printer ZPL Command Development Manual v2.1.5]

| | |
|--|----|
| [ZPL Windows SDK] | 1 |
| 1. Information of the Manual | 4 |
| 2. Operation System | 4 |
| 3. Remark | 4 |
| 4. NuGet Package Manager | 4 |
| 5. Method | 4 |
| 5.1. PrinterCreator | 4 |
| 5.2. ReleasePrinter | 5 |
| 5.3. OpenPort | 5 |
| 5.4. ClosePort | 6 |
| 5.5. WriteData | 6 |
| 5.6. ReadData | 7 |
| 5.7. ZPL_StartFormat | 7 |
| 5.8. ZPL_EndFormat | 8 |
| 5.9. ZPL_ScalableFontText | 8 |
| 5.10. ZPL_Text | 9 |
| 5.11. ZPL_BarCode39 | 10 |
| 5.12. ZPL_Pdf417 | 11 |
| 5.13. ZPL_CodeEan8 | 12 |
| 5.14. ZPL_UpceCode | 13 |
| 5.15. ZPL_BarCode93 | 14 |
| 5.16. ZPL_BarCode128 | 15 |
| 5.17. ZPL_CodeEan13 | 16 |
| 5.18. ZPL_MicroPdf417 | 17 |
| 5.19. ZPL_QRCode | 18 |
| 5.20. ZPL_UpcExtensions | 20 |
| 5.21. ZPL_UpcaBarcode | 21 |
| 5.22. ZPL_SetChangeFontEncoding | 22 |
| 5.23. ZPL_SetChangeCaret | 23 |
| 5.24. ZPL_SetChangeDelimiter | 23 |
| 5.25. ZPL_SetChangeTilde | 24 |
| 5.26. ZPL_GraphicBox | 24 |
| 5.27. ZPL_GraphicCircle | 25 |
| 5.28. ZPL_GraphicDiagonalLine | 25 |
| 5.29. ZPL_GraphicEllipse | 26 |
| 5.30. ZPL_PrintImage | 27 |
| 5.31. ZPL_GraphicSymbol | 27 |
| 5.32. ZPL_SetDiagnosticsMode | 28 |
| 5.33. ZPL_SetLabelHome | 29 |
| 5.34. ZPL_SetLabelLength | 29 |
| 5.35. ZPL_SetLabelShift | 30 |
| 5.36. ZPL_SetLabelTop | 30 |
| 5.37. ZPL_SetPrintMode | 31 |
| 5.38. ZPL_SetMediaType | 32 |
| 5.39. ZPL_SetPrintingMirrorImage | 32 |
| 5.40. ZPL_SetPrintOrientation | 33 |
| 5.41. ZPL_SetPrintRate | 33 |
| 5.42. ZPL_SetPrintWidth | 34 |
| 5.43. ZPL_SetSerialCommunications | 34 |
| 5.44. ZPL_SetPrintDarkness | 35 |
| 5.45. ZPL_SetTearOffAdjustPosition | 36 |
| 5.46. ZPL_PrintConfigurationLabel | 36 |
| 5.47. ZPL_GetPrinterIpAddress | 37 |
| 5.48. ZPL_GetPrinterStatus | 37 |
| 5.49. ZPL_GetLabelLength | 38 |

| | |
|---|----|
| 5.50. ZPL_GetLabelWidth | 38 |
| 5.51. ZPL_GetPrinterSeriesNumber | 39 |
| 5.52. ZPL_GetPrinterMacAddress | 39 |
| 5.53. ZPL_GetPrinterName | 40 |
| 5.54. ZPL_GetPrinterFirmwareVersion | 40 |
| 5.55. ZPL_GetPrinterDpi | 41 |
| 5.56. ZPL_GetPrinterModel | 41 |
| 5.57. ZPL_LearnLabel | 42 |
| 5.58. ZPL_SetReprintAfterError | 42 |
| 5.59. ZPL_SetMediaTracking | 43 |
| 5.60. ZPL_SetUserFontName | 44 |
| 5.61. ZPL_Text_Block | 44 |
| 5.62. ZPL_SetPrintQuantity | 45 |
| 5.63. ZPL_DataMatrixBarcode | 46 |
| 5.64. ZPL_GetPrinterOdometer | 47 |
| 5.65. ZPL_SetPrintNetSetting | 48 |
| 5.66. ZPL_WifiConfig | 48 |
| 5.67. ZPL_SetPrinterBluetoothSSID | 49 |
| 5.68. ZPL_SetPrinterBluetoothPIN | 50 |
| 5.69. ZPL_SetPrinterSleepTime | 50 |
| 5.70. ZPL_SetPrinterShutdownTime | 51 |
| 5.71. ZPL_FirmwareUpgrade | 51 |
| 5.72. ZPL_FontDownload | 52 |
| 5.73. ZPL_RfidCalibration | 52 |
| 5.74. ZPL_RfidWrite | 53 |
| 5.75. ZPL_RfidDefineFont | 54 |
| 5.76. ZPL_RfidRead | 55 |
| 5.77. ZPL_RfidSetPower | 55 |
| 5.78. ZPL_RfidDefineEPC | 56 |
| 5.79. ZPL_RfidSetParam | 57 |

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 10 or above

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.

The printer resolution is 200 dpi,1 mm=8 dot;The printer resolution is 300 dpi,1 mm=12 dot.

4. NuGet Package Manager

If you use Visual Studio tools, you can use Nuget.

NuGet URL:<http://47.245.124.98:8081/repository/nuget-hosted/>

User and password: You can obtain your account and password for free by consulting customer service.

5. Method

5.1.PrinterCreator

Set up the target printer of specified model (the printer object must be created before any printer operation).

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

Parameter:

*void** handle*

[in,out] The created target printer object.

const TCHAR model*

[in] Specify the model of target printer.

Return Value:

| Code | Value | Description |
|----------------------------|-------|------------------|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |

5.2.ReleasePrinter

The method is to release the resources of the printer object (the created printer object must be released after the operation is completed).

```
int ReleasePrinter (  
    void* hPrinter  
);
```

Parameter:

void hPrinter*

[in] Handle to the target printer object that needs to be released

Return Value:

| Code | Value | Description |
|------------------------------|-------|----------------------------|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |

5.3.OpenPort

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. Currently it supports USB, internet, serial interface.

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

Parameter:

void hPrinter*

[in] The created target printer object.

setting

[in] Set the communication port parameters to connect to the target printer. See the table below for details:

Configuration List:

| Type | Configuration | Description | Sample |
|------|-------------------------------|--|---|
| USB | USB,Model/PortNum | USB,printer model USB,The port number If you connect multiple printers of different models of our company at the same time, it is recommended to use "USB, model" to connect | USB,4B-2054A USB,USB031 |
| NET | NET, IP address (IPV4)[,port] | Specify the IPAddress and port.If no port is specified,The default port is 9100. | NET,192.168.1.10 NET,192.168.1.10,9100 |
| COM | COMn,rate | Specify the number and baud rate of connected serial port | COM10,19200 |

| | | | |
|-----|------|--|------|
| LPT | LPTn | Specify the number of connected parallel port. | LPT3 |
|-----|------|--|------|

Return Value:

| Code | Value | Description |
|-------------------------------|-------|----------------------------|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_USB_DEVICE_NOT_FOUND | -17 | Failed, device not found |
| ERROR_IO_OPEN_FAILED | -8 | Failed to open port |

5.4.ClosePort

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

```
int ClosePort (
    void* hPrinter
);
```

Parameter:

void hPrinter*
[in] The created target printer object.

Return Value:

| Code | Value | Description |
|------------------------------|-------|----------------------------|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |

5.5.WriteData

This function is to send data to the printer.

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

Parameter:

void handle*
[in] The created target printer object.
unsigned char buffer*
[in] The data sent to the printer (hex string).
unsigned int size
[in] The length of the sent data.

Return Value:

| Code | Value | Description |
|------------------------------|-------|----------------------------|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |

| | | |
|------------------------|-----|----------------------|
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |

5.6.ReadData

This function is to read the printer data.

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

Parameter:

void handle*

[in] The created target printer object.

int timeout

[in] Read timeout

unsigned char buffer*

[in] Printer data to be read.

unsigned int size

[in] The length of the data to be read.

Return Value:

| Code | Value | Description |
|------------------------------|-------|----------------------------|
| >0 | >0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_OPEN_FAILED | -8 | Failed to open port |

5.7. ZPL_StartFormat

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

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

Parameter:

void handle*

[in] The created target printer object.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.8. ZPL_EndFormat

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

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

Parameter:

void handle*

[in] The created target printer object.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.9. 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,  
    const TCHAR* text  
);
```

Parameter:

void handle*

[in] 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.
*const TCHAR** *text*
[in]Text data.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.10. 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,
    const TCHAR* text
);
```

Parameter:

*void** *handle*

[in] 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 P - Bitmap font
- 9 : FONT Q - Bitmap font
- 10 : FONT R - Bitmap font
- 11 : FONT S - Bitmap font
- 12 : FONT T - Bitmap font
- 13 : FONT U - Bitmap font
- 14 : FONT V - Bitmap font

int *orientation*

```
FONT A -- ABCDxyz 12345
FONT B -- ABCDxyz 12345 UPPER CASE ONLY
FONT D -- ABCDxyz 12345
FONT E -- (OCR-B)ABCDwxyz 12345
FONT F -- ABCDwxyz 12345
FONT G -- ABByz 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
```

[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.

const TCHAR text*

[in]Text data.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.11. 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,
    const TCHAR* text
);
```

Parameter:

void handle*

[in] 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

const TCHAR text*
[in] Text data.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.12. 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,
    const TCHAR* text
);
```

Parameter:

void handle*
[in]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
const TCHAR text*
 [in] QR code data.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.13. 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,
    const TCHAR* text
);
```

Parameter:

void handle*
 [in]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
const TCHAR text*
 [in] Text data.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.14. 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,
    const TCHAR* text
);
```

Parameter:

void handle*
 [in]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
const TCHAR text*
[in] Text data.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.15. 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,
    const TCHAR* text
);
```

Parameter :

void handle*
[in] 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

const TCHAR text*
[in] Text data.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.16. 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,
    const TCHAR* text
);
```

Parameter:

void handle*
[in] 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

const TCHAR text*

[in] Text data.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.17. 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,
    const TCHAR* text
);
```

Parameter:

void handle*

[in]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

const TCHAR text*
[in] Text data.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.18. 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,
    const TCHAR* text
);
```

Parameter :

void handle*
[in]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 |

const TCHAR text*

[in] Text data.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.19. 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,
    const TCHAR* text
);

```

Parameter :

void handle*

[in] 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- 10).

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.

const TCHAR 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 :

| Code | Value | Description |
|------------------------------|--------------|--------------------------------|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the |

5.20. 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,
    const TCHAR* text
);
```

Parameter:

void handle*

[in] 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

const TCHAR text*

[in] Text data.

Return Value:

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.21. 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,  
    const TCHAR* text  
);
```

Parameter:

void handle*

[in] 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

const TCHAR text*

[in] Text data.

Return Value:

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.22. ZPL_SetChangeFontEncoding

This function is to select an international character set.

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

Parameter:

void handle*

[in] 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:

| Code | Value | Description |
|------------------------------|--------------|--------------------------------|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the |

| | | |
|--|--|----------------|
| | | Windows system |
|--|--|----------------|

5.23. ZPL_SetChangeCaret

This function is to change the format command prefix.

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

Parameter:

void handle*

[in] The created target printer object.

char character

[in] Format command prefix.

Return Value:

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.24. ZPL_SetChangeDelimiter

This function is to change the separator.

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

Parameter:

void handle*

[in] The created target printer object.

char character

[in] Separator.

Return Value:

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.25. ZPL_SetChangeTilde

This function is to change the control command prefix.

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

Parameter:

void handle*

[in] The created target printer object.

char character

[in] Control command prefix.

Return Value:

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.26. 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] 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 :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.27. 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] 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 :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.28. 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] 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 :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.29. 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] 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 :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.30. ZPL_PrintImage

This function is to print image.

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

parameter :

void handle*
[in] 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).
const TCHAR imgName*
[in] The path to the image.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.31. 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,
const char symbol
);

```

Parameter:

void handle*

[in] 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.

const char symbol

[in] Data string.

Return Value:

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.32. ZPL_SetDiagnosticsMode

This function is to start the diagnostic mode.

```

int ZPL_SetDiagnosticsMode(
    void* handle,
    int isEnabled
);

```

Parameter:

void handle*

[in] 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 :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.33. ZPL_SetLabelHome

This function is to set the label home position.

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

Parameter:

void handle*

[in] 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 :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.34. ZPL_SetLabelLength

This function is to set the label length.

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

Parameter:

void handle*

[in] The created target printer object.

int length

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

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.35. 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] The created target printer object.

int shift

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

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.36. 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] The created target printer object.

int top

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

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.37. 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] 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 :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.38. 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] The created target printer object.

char type

[in] Media type.

'T' : thermal transfer media

'D' : direct thermal media

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.39. 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] The created target printer object.

char enable

[in] Whether to open.

'N': not open

'Y': open

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.40. ZPL_SetPrintOrientation

This function is to flip the label format 180 degrees.

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

Parameter:

void handle*

[in] The created target printer object.

int orientation

[in] Whether to flip.

0 : don't flip

180: perform a flip

Return Value:

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.41. 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] 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:

| Code | Value | Description |
|-------------------------|-------|----------------------------|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |

| | | |
|------------------------------|--------------|---|
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.42. ZPL_SetPrintWidth

This function is to set print width.

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

Parameter:

void handle*

[in] The created target printer object.

int width

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

Return Value:

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.43. 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] 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 protocol/Modo

[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 :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.44. ZPL_SetPrintDarkness

This function is to set print darkness.

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

Parameter:

void handle*

[in] The created target printer object.

int darkness

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

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.45. 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] The created target printer object.

int position

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

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.46. ZPL_PrintConfigurationLabel

This function is to generate a printer configuration label.

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

Parameter:

void handle*

[in] The created target printer object.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.47. ZPL_GetPrinterIpAddress

This function is to get the printer IP address.

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

Parameter:

void handle*

[in] The created target printer object.

char ipAddress*

[in] Printer's IP address.

Return Value:

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.48. ZPL_GetPrinterStatus

This function is to get the status of the printer.

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

Parameter:

void handle*

[in] The created target printer object.

*int * status*

[in,out] The status of the printer.

| Status | Value | Bit |
|--------------------------|-------|-----|
| Normal | 0 | - |
| The print head is opened | 1 | 0 |
| Paper jam | 2 | 1 |
| Out of paper | 4 | 2 |
| Out of ribbon | 8 | 3 |
| Print pause | 16 | 4 |
| Printing | 32 | 5 |
| Cover opened | 64 | 6 |
| Other error | 128 | 7 |

Return Value:

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.49. ZPL_GetLabelLength

This function is to get the length of the label.

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

Parameter:

void handle*

[in] The created target printer object.

char length*

[in] The length of the label.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.50. ZPL_GetLabelWidth

This function is to get the width of the label.

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

Parameter:

void handle*

[in] The created target printer object.

char width*

[in] The width of the label.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.51. ZPL_GetPrinterSeriesNumber

This function is to get the printer serial number.

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

Parameter:

void handle*

[in] The created target printer object.

char sn*

[in] Printer serial number.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.52. ZPL_GetPrinterMacAddress

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

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

Parameter:

void handle*

[in] The created target printer object.

char macAddress*

[in] The MAC address of the printer.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.53. ZPL_GetPrinterName

This function is to get the printer's name.

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

Parameter:

void handle*

[in] The created target printer object.

char name*

[in] The name of the printer.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.54. 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] The created target printer object.

char version*

[in] The firmware version number of the printer.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.55. ZPL_GetPrinterDpi

This function is to get the resolution of the printer.

```
int ZPL_GetPrinterDpi(
    void* handle,
    char* dpi
);
```

Parameter:

void handle*

[in] The created target printer object.

char dpi*

[in] The resolution of the printer.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.56. ZPL_GetPrinterModel

This function is to get the model of the printer.

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

Parameter:

void handle*

[in] The created target printer object.

char dpi*

[in] The model of the printer.

Eg:

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

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.57. ZPL_LearnLabel

This feature is used for automatic label learning.

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

Parameter :

void handle*

[in] The created target printer object.

(The interface needs to be called before ZPL_StartFormat)

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.58. 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] 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 :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.59. 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] 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 :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.60. ZPL_SetUserFontName

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

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

Parameter:

void handle*

[in] The created target printer object.

const TCHAR text*

[in] Font name

char alias

[in] *alias*

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.61. 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,  
    const TCHAR* text  
);
```

Parameter:

void handle*

[in] 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 P - Bitmap font
 9 : FONT Q - Bitmap font
 10 : FONT R - Bitmap font
 11 : FONT S - Bitmap font
 12 : FONT T - Bitmap font
 13 : FONT U - Bitmap font
 14 : FONT V - Bitmap 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.

int textBlockWidth

[in] Text block width

int textBlockHeight

[in] Text block height

const TCHAR text*

[in]Text data.

Note: The data does not support Chinese at this time

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.62. ZPL_SetPrintQuantity

This function is to gives 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] 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:

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.63. 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,

const TCHAR text*

);

Parameter:

void handle*

[in] 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

const TCHAR text*

[in] code data.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.64. ZPL_GetPrinterOdometer

This function is to get the number of printed mileage.

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

Parameter:

void handle*

[in] The created target printer object.

char meters*

[in] printed mileage。

Return Value :

| Code | Value | Description |
|-------------------------|-------|----------------------------|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |

| | | |
|------------------------------|--------------|---|
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.65. ZPL_SetPrintNetSetting

This function is to set the network port information.

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

Parameter:

void handle*

[in] The created target printer object.

const char ipaddress*

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

const char mask*

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

const char gateway*

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

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.66. ZPL_WifiConfig

This function is to set wifi sta information.

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


);

Parameter:

void handle*

[in] The created target printer object.

int dhcp

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

const char ipAddress*

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

const char mask*

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

const char gateway*

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

const char ssid*

[in] WiFi ssid.

const char password*

[in] WiFi password.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.67. ZPL_SetPrinterBluetoothSSID

This function is to set the Bluetooth SSID.

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

Parameter:

void handle*

[in] The created target printer object.

const TCHAR ssid*

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

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.68. ZPL_SetPrinterBluetoothPIN

This function is to set the Bluetooth pin code.

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

Parameter:

void handle*

[in] The created target printer object.

const TCHAR pin*

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

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.69. ZPL_SetPrinterSleepTime

This function is to set the sleep time.

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

Parameter:

void handle*

[in] The created target printer object.

int time

[in] sleep time(range: 0-10, unit: minute)

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.70. ZPL_SetPrinterShutdownTime

This function is to set the automatic shutdown time.

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

Parameter:

void handle*

[in] The created target printer object.

int time

[in] Automatic shutdown time (range: 0-63, unit: minute)

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.71. ZPL_FirmwareUpgrade

This function is to upgrade the printer firmware.

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

Parameter:

void handle*

[in] The created target printer object.

const TCHAR cFileName*

[in] Firmware file path

*void (*progressCallback)(float)*

update progress callback

| describe | Value |
|---------------------|------------------------------|
| update progress | 0~1 |
| Update success | ERROR_CM_SUCCESS |
| Not enough memory | ERROR_CM_INSUFFICIENT_MEMORY |
| Failed to read file | ERROR_IO_READ_FAILED |
| Failed to send data | ERROR_IO_WRITE_FAILED |

Return Value :

| Code | Value | Description |
|------------------|-------|-------------|
| ERROR_CM_SUCCESS | 0 | success |

| | | |
|------------------------------|--------------|---|
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.72. ZPL_FontDownload

This function is a font download.

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

Parameter:

void handle*

[in] The created target printer object.

const TCHAR cFileName*

[in] Font file path

*void (*progressCallback)(float)*

update progress callback

| describe | Value |
|---------------------|-------|
| update progress | 0~1 |
| Update success | 0 |
| Not enough memory | -4 |
| Failed to read file | -11 |
| Failed to send data | -9 |

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.73. ZPL_RfidCalibration

This function is to initiate tag calibration for RFID media.

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

Parameter:*void* handle*

[in] The created target printer object.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.74. ZPL_RfidWrite

This function is to Write RFID Format.

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

Parameter:*void* handle*

[in] The created target printer object.

char format

[in] format: A = ASCII, H = Hexadecimal, E = EPC (We need to first use ZPL_RfidDefineEPC() to define the EPC data structure)

int begin

[in] starting block number

int size

[in] number of bytes to read or write

int memoryBlock

[in] Specifies the Gen 2 memory bank.: 0 = Reserved, 1 = EPC 2 = TID (Tag ID), 3 = User

const TCHAR text*

[in] Text data

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.75. ZPL_RfidDefineFont

This function is to define the text format read by RFID for printing, and to be used in conjunction with ZPL_RfidRead.

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

Parameter:

void handle*

[in] 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 P - Bitmap font

9 : FONT Q - Bitmap font

10 : FONT R - Bitmap font

11 : FONT S - Bitmap font

12 : FONT T - Bitmap font

13 : FONT U - Bitmap font

14 : FONT V - Bitmap 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.

Return Value :

| Code | Value | Description |
|------------------|-------|-------------|
| ERROR_CM_SUCCESS | 0 | success |

| | | |
|------------------------------|--------------|---|
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.76. ZPL_RfidRead

This function is to Read RFID Format, Please use it in conjunction with the ReadDataTimeout function.

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

Parameter:

void handle*

[in] The created target printer object.

char format

[in] format: A = ASCII, H = Hexadecimal, E = EPC (We need to first use ZPL_RfidDefineEPC() to define the EPC data structure)

int begin

[in] starting block number

int size

[in] number of bytes to read or write

int memoryBlock

[in] Specifies the Gen 2 memory bank.: 0 = Reserved, 1 = EPC 2 = TID (Tag ID), 3 = User

int isPrint

[in] Is the content read printed out. 1 is used in conjunction with ZPL_RfidDefineFont for printing, 0 is not printed.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.77. ZPL_RfidSetPower

This function is to Set RF Power Levels for Read and Write

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

Parameter:

void handle*

[in] The created target printer object.

unsigned char read

[in] read power, Values: 0 to 30

unsigned char write

[in] write power, Values: 0 to 30

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.78. ZPL_RfidDefineEPC

This function is to Define EPC Data Structure

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

Parameter:

void handle*

[in] The created target printer object.

unsigned char* bits,

[in] Partition size set. The maximum single partition is 64 bits.

int count

[in] The maximum number of partitions is 16.

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |

5.79. ZPL_RfidSetParam

This function is to set up RFID parameters including tag type.

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

Parameter:

void handle*

[in] The created target printer object.

unsigned char labelType

[in] Tag type.Values: 8 = EPC Class 1, Generation 2 (Gen 2)

int pos

[in] read/write position of the tag (programming position)

int len

[in] length of void printout

int number

[in] number of labels to try encoding,The number of labels that will be attempted in case of read/encode failure.Values: 1 to 10

char err

[in] error handling, Values:

N = No action (printer drops the label format causing the error and moves to the next queued label)

P = Place printer in Pause mode (label format stays in the queue until the user cancels)

E = Place printer in Error mode (label format stays in the queue until the user cancels)

Return Value :

| Code | Value | Description |
|------------------------------|--------------|---|
| ERROR_CM_SUCCESS | 0 | success |
| ERROR_CM_INVALID_HANDLE | -2 | failed with invalid handle |
| ERROR_CM_INVALID_PARAMETER | -1 | Invalid argument |
| ERROR_CM_INSUFFICIENT_MEMORY | -4 | failed, out of memory |
| ERROR_IO_WRITE_FAILED | -9 | Failed to send data |
| ERROR_IO_READ_FAILED | -11 | Failed to read data |
| ERROR_IO_WRITE_TIMEOUT | -10 | Write data timed out |
| Other values | Other values | the error code returned by the Windows system |