

# Label Printer

ZMWIN

**API FUNCTIONS MANUAL**

Version: 2.0.0.2

2012@  
ZMIN TECHNOLOGIES CO.,LTD.



# Contents

Description of API functions file .....	1
Purpose .....	2
Abbreviations contrast.....	3
Notice .....	4
Coordinates system.....	5
Functions lists.....	6
OpenPort.....	8
ClosePort .....	9
ZM_ClearBuffer.....	10
SetPCComPort .....	11
GetErrState.....	12
ZM_GetInfo.....	13
ZM_SetDirection .....	14
ZM_SetDarkness.....	15
ZM_SetPrintSpeed .....	16
ZM_SetLabelHeight .....	17
ZM_SetLabelWidth.....	18
ZM_SetCoordinateOrigin.....	19
ZM_DrawBarcode.....	20
ZM_DrawBarcodeEx .....	23
ZM_DrawBar2D_DATAMATRIX .....	26
ZM_DrawBar2D_QR.....	27
ZM_DrawBar2D_MaxiCode .....	29
ZM_DrawBar2D_Pdf417 .....	30
ZM_DrawTextTrueTypeW .....	32
ZM_DrawText.....	35
ZM_DrawTextEx .....	37
ZM_PcxGraphicsList.....	40
ZM_PcxGraphicsDel .....	41
ZM_PcxGraphicsDownload .....	42
ZM_BmpGraphicsDownload.....	43
ZM_DrawPcxGraphics .....	44
ZM_PrintPCX .....	45
ZM_DrawBinGraphics.....	46
ZM_DrawLineXor.....	48
ZM_DrawLineOr .....	49
ZM_DrawDiagonal.....	50
ZM_DrawWhiteLine.....	51
ZM_DrawRectangle .....	52
ZM_PrintLabel.....	53
ZM_SoftFontList .....	54
ZM_SoftFontDel.....	55
ZM_FormEnd .....	56
ZM_FormList .....	57
ZM_FormDel .....	58
ZM_FormDownload.....	59
ZM_ExecForm.....	60
ZM_PrintLabelAuto.....	61
ZM_DefineCounter .....	62
ZM_DefineVariable.....	64

ZM_Download .....	65
ZM_DownloadInitVar .....	66
ZM_PrintConfiguration .....	67
ZM_Reset .....	68
ZM_SetPrinterState .....	69
ZM_FeedMedia .....	70
ZM_MediaDetect .....	71
ZM_CutPage .....	72
ZM_EnableFIASH .....	73
ZM_DisableFLASH .....	74
ZM_BinGraphicsList .....	75
ZM_BinGraphicsDel .....	76
ZM_BinGraphicsDownload .....	77
ZM_RecallBinGraphics .....	79
ZM_DisableErrorReport .....	80
ZM_EnableErrorReport .....	81
ZM_ErrorReport .....	82
ZM_ErrorReportEx .....	84
ZM_SetPagePrintCount .....	86
ZM_WritePrinter .....	87
ZMWIN.dll error run return code lists .....	88

## Description of API functions file

File Name: ZMWIN.dll

Version: 2.X.X.X

Copyright: ©2012 ZMIN TECHNOLOGIES CO.,LTD.All Right Reserved.

---

## Purpose

This API function library is used with ZMIN Printers, for compiling the applications based on the operating system of Windows9X, NT, 2000, XP, Win7, Win8.

This API function library only supports products of ZMIN Technologies Co., Ltd.

This API function supports customers send the command language to printer through the ZMIN windows driver.

---

## Abbreviations contrast

**PCLE:** Printer Command Language E of ZMIN.

**API:** Application Program Interface.

**Dots:** Pixel is a kind dimensional unit used in computer science technology. Originally means the minimum unit of the TV imaging. The minimum imaging unit for a printer: Dots are equal to one inch divided by the maximum resolution of the printer.

For example: 1 inch = 25.4mm or 1000mil

For the 203 DPI Printers: 1 dot =  $25.4\text{mm} / 203 = 0.125\text{mm}$  (1dot =  $1000\text{mil} / 203 = 5\text{mil}$ )

For the 300 DPI printers: 1 dot =  $25.4\text{mm} / 300 = 0.085\text{mm}$  (1dot =  $1000\text{mil} / 300 = 3\text{mil}$ )

**TrueType Font:** TrueType is an outline font standard developed by Apple and Microsoft in the late 1980s. It has become the most common format for fonts on both the Microsoft Windows operating systems.

**The TrueType Font have installed in windows can be used by this API functions.**

---

**Notice****String**

In command sets, we can use data strings with the following characteristics:

Name: for graphics, soft fonts and forms.

Data: for fonts and bar code

The quotation mark character ( " ) designates the beginning and ending of a string.

The backslash (\) character designates that the following character(s) is literal and will be encoded into the data field. Please refer to the following

Examples:

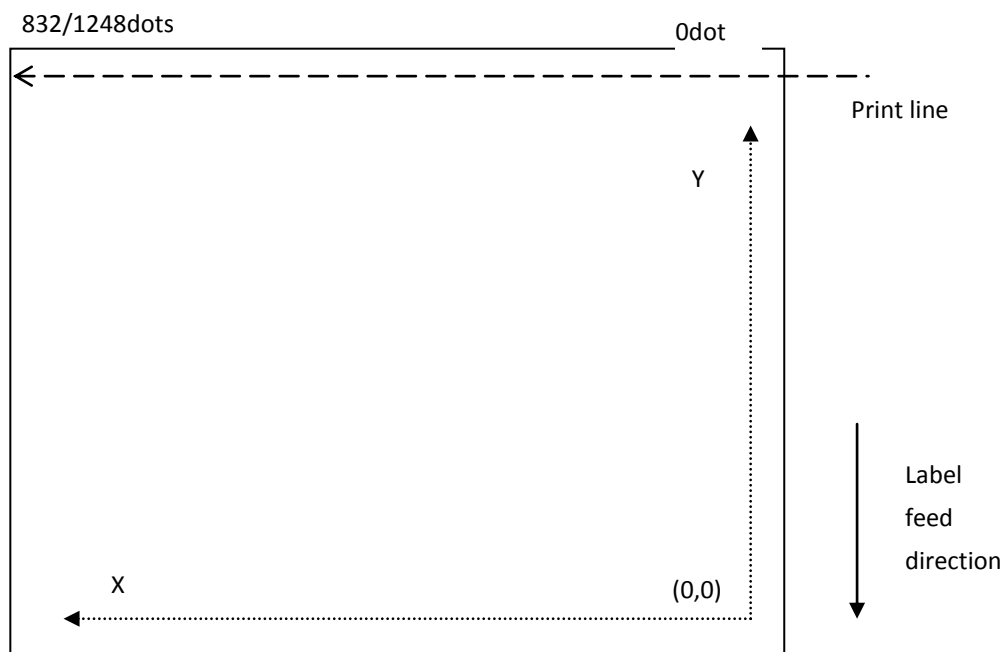
Character input

To Print	Input
"	\ "
\	\\
0x00 - 0x7F	\x00 - \x7F

Note: All commands and names are case sensitive.

## Coordinates system

The coordinates system for the barcode label printer is shown below:





## Functions lists

Name	Description
<a href="#"><u>OpenPort</u></a>	Open the windows driver's communication port, prepare to send commands.
<a href="#"><u>ClosePort</u></a>	Close the communication port opened by OpenPort(),End to transfer.
<a href="#"><u>ZM_ClearBuffer</u></a>	Clear the printer's ram buffer.
<a href="#"><u>SetPCComPort</u></a>	Set the windows's COM port.
<a href="#"><u>GetErrState</u></a>	Get the error run return code if there has any error when run the API.
<a href="#"><u>ZM_GetInfo</u></a>	Get the DLL's information of version.
<a href="#"><u>ZM_SetDirection</u></a>	Set label print orientation.
<a href="#"><u>ZM_SetDarkness</u></a>	Set print darkness.
<a href="#"><u>ZM_SetPrintSpeed</u></a>	Set print speed.
<a href="#"><u>ZM_SetLabelHeight</u></a>	Set the height and gap of the label.
<a href="#"><u>ZM_SetLabelWidth</u></a>	Set the width of the label.
<a href="#"><u>ZM_SetCoordinateOrigin</u></a>	Set the coordinate reference point.
<a href="#"><u>ZM_DrawBarcode</u></a>	Print 1D barcode which content is a constant string.
<a href="#"><u>ZM_DrawBarcodeEx</u></a>	Print 1D barcode, which content can be combined with the fixed data, Counter,Variable.
<a href="#"><u>ZM_DrawTextTrueTypeW</u></a>	Print text with TrueType Font.
<a href="#"><u>ZM_DrawText</u></a>	Print text with printer's internal font, which content is a constant string.
<a href="#"><u>ZM_DrawTextEx</u></a>	Print text with printer's internal font., which content can be combined with the fixed data, Counter,Variable.
<a href="#"><u>ZM_PcxGraphicsList</u></a>	Print PCX graphics name list which have been stored in the printer.
<a href="#"><u>ZM_PcxGraphicsDel</u></a>	Delete one or all of PCX graphics which have been stored in the printer.
<a href="#"><u>ZM_PcxGraphicsDownload</u></a>	Save PCX graphics to the printer.
<a href="#"><u>ZM_BmpGraphicsDownload</u></a>	Converting BMP graphics to PCX graphics and save it to the printer.
<a href="#"><u>ZM_DrawPcxGraphics</u></a>	Print PCX graphics.
<a href="#"><u>ZM_PrintPCX</u></a>	Print PCX graphics.
<a href="#"><u>ZM_DrawBinGraphics</u></a>	Print Binary graphics.
<a href="#"><u>ZM_DrawLineXor</u></a>	Draw line with "Exclusive OR" function.
<a href="#"><u>ZM_DrawLineOr</u></a>	Draw line with "OR" function.
<a href="#"><u>ZM_DrawDiagonal</u></a>	Draw diagonal line.
<a href="#"><u>ZM_DrawWhiteLine</u></a>	Draw white line.
<a href="#"><u>ZM_DrawRectangle</u></a>	Draw rectangle.
<a href="#"><u>ZM_PrintLabel</u></a>	Print out label.
<a href="#"><u>ZM_SoftFontList</u></a>	Print out soft font name list which have been stored in the printer.
<a href="#"><u>ZM_SoftFontDel</u></a>	Delete one or all of soft font which have been stored in the printer.
<a href="#"><u>ZM_FormEnd</u></a>	Ends Form store. It must be used together with ZM_FormDownload().

<a href="#"><u>ZM_FormList</u></a>	Print out Form name list which have been stored in the printer.
<a href="#"><u>ZM_FormDel</u></a>	Delete one or all of Form which have been stored in the printer.
<a href="#"><u>ZM_FormDownload</u></a>	Save Form to the printer. It must be used together with ZM_FormEnd().
<a href="#"><u>ZM_ExecForm</u></a>	Run Form.
<a href="#"><u>ZM_PrintLabelAuto</u></a>	Auto print the form as soon as all variable data has been supplied.( Use this command only in a stored form)
<a href="#"><u>ZM_DefineCounter</u></a>	Define Counter.
<a href="#"><u>ZM_DefineVariable</u></a>	Define Variable.
<a href="#"><u>ZM_EnableFIASH</u></a>	Enable the data are stored to FLASH.(Depend on model of printer)
<a href="#"><u>ZM_DisableFLASH</u></a>	Disable the data are stored to FLASH.
<a href="#"><u>ZM_Download</u></a>	Download Variable or Counter.
<a href="#"><u>ZM_DownloadInitVar</u></a>	Set the initial value of Variable or Counter.
<a href="#"><u>ZM_PrintConfiguration</u></a>	Print out self test label.
<a href="#"><u>ZM_Reset</u></a>	Reset the printer.
<a href="#"><u>ZM_SetPrinterState</u></a>	Set the printer's work state.
<a href="#"><u>ZM_FeedMedia</u></a>	Feed out a label.
<a href="#"><u>ZM_MediaDetect</u></a>	Perform the label calibration.
<a href="#"><u>ZM_CutPage</u></a>	Sets cutter work status.
<a href="#"><u>ZM_BinGraphicsList</u></a>	Print out binary graphics name list which have been stored in the printer.
<a href="#"><u>ZM_BinGraphicsDel</u></a>	Delete one or all of binary graphics name list which have been stored in the printer.
<a href="#"><u>ZM_BinGraphicsDownload</u></a>	Save a binary graphics to the printer.
<a href="#"><u>ZM_RecallBinGraphics</u></a>	Print binary graphics which have been stored in the printer.
<a href="#"><u>ZM_DisableErrorReport</u></a>	Disable the printer's status report via the COM port.
<a href="#"><u>ZM_EnableErrorReport</u></a>	Enable the printer's status report via the COM port.
<a href="#"><u>ZM_ErrorReport</u></a>	Get the printer's status code.
<a href="#"><u>ZM_ErrorReportEx</u></a>	Get the printer's status code with timeout.
<a href="#"><u>ZM_SetPagePrintCount</u></a>	Set the copies of page.
<a href="#"><u>ZM_WritePrinter</u></a>	Send the data to printer.

## OpenPort

### Description:

This function is used to open the communications port.

The OpenPort function must be executed correctly prior to being able to use the other functions in this directory.

### Syntax:

```
int OpenPort(  
    LPTSTR xxxx  
);
```

### Parameters:

xxxx

The name of the current printer used by windows.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
int return = OpenPort("ZMIN X1");    //Open the communication port currently chosen by ZMIN X1.
```

## ClosePort

**Description:**

This function is used to close the communication port which opened by the OpenPort function. It suggests that call the ClosePort to close the communication port after completing all of the operation processed by other functions. Otherwise, your program will always take up the opened communication port until the program be closed.

**Syntax:**

**void ClosePort**(void);

**Parameters:**

None

**Return Value:**

None

**Example:**

ClosePort( );

## ZM\_ClearBuffer

**Description:**

This function is used to clear the contents in printer buffer.

It suggest that use this function to clear the contents in printer buffer before sending a new label to the printer.

**Note: Please do not use this function in the FORM.**

**Syntax:**

```
int ZM_ClearBuffer (void);
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_ClearBuffer ( );
```

## SetPCComPort

**Description:**

This function is used to set the baud rate of the COM port in PC.

It is only available when use the COM port.

**Note:**

Be sure to correspond with the serial baud rate setting in your printer (by adjusting the DIP switch pin7&pin8 on or off, please refer to the user's manual).

**Syntax:**

```
int SetPCComPort(  
    DWORD BaudRate,  
    BOOL HandShake  
);
```

**Parameters:**

BaudRate

Set baud rate value.

value: 9600,19200,38400,57600;

HandShake

HandShaking flag;

TRUE: Enable HandShaking;

FALSE: Disable HandShaking.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
SetPCComPort ( 9600, TRUE);
```

## GetErrState

### Description:

This function is used to check the correctness/validity of the other functions in ZMWIN.DLL.

Please refer to the section of ZMWIN.dll description of returning errors for error codes.

This function must use before the Closeport().

### Syntax:

```
int GetErrState(void);
```

### Parameters:

None

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
int state = 0;  
OpenPort("ZMIN X1");  
...  
state = GetErrState();  
...  
ClosePort();
```

## ZM\_GetInfo

**Description:**

This function is used to get the edition information of this API DLL.

**Syntax:**

```
int ZM_GetInfo(void);
```

**Parameters:**

None.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_GetInfo(void)
```



## ZM\_SetDirection

**Description:**

This function is used to set print orientation for all graphics, text, bar codes, lines and rectangles.  
this function will change the direction of contents, such as text, barcode, straight line, and rectangle.

**Syntax:**

```
int ZM_SetDirection (  
    char direct  
);
```

**Parameters:**

direct

Orientation; Acceptable values are B or T. The default value is T.

B: Print from bottom right corner.

T: Print from top left corner.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_SetDirection ('B');
```

## ZM\_SetDarkness

**Description:**

This function is used to set darkness of the TPH.

**Syntax:**

```
int ZM_SetDarkness (  
    unsigned int id  
);
```

**Parameters:**

id

Acceptable values are from 0 to 20, the default value is 10.

This value is not in deed the temperature of the TPH, it is a relative value. The lightest print darkness is achieved with a value of 0 and the greatest print darkness is achieved with a value of 20.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_SetDarkness (10);
```

## ZM\_SetPrintSpeed

### Description:

This function is used to set the print speed.

### Syntax:

```
int ZM_SetPrintSpeed (  
    unsigned int px  
);
```

### Parameters:

px

Value: 10 – 80.

px value	speed	PPLB(compatible)
10	1.0 ips (25 mm/s)	0 or 1
15	1.5 ips (37 mm/s)	
20	2.0 ips (50 mm/s)	2
25	2.5 ips (63 mm/s)	
30	3.0 ips (75 mm/s)	3
35	3.5 ips (83 mm/s)	
40	4.0 ips (100 mm/s)	4
50	5.0 ips (125 mm/s)	5
60	6.0 ips (150 mm/s)	6
70	7.0 ips (175 mm/s)	
80	8.0 ips (200 mm/s)	

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_SetPrintSpeed (40);
```

## ZM\_SetLabelHeight

### Description:

This function is used to set the label's height and the height of media gap/black line/bore hole.

### Syntax:

```
int ZM_SetLabelHeight (  
    unsigned int  lheight,  
    unsigned int  gapH  
);
```

### Parameters:

lheight

label height in dots. Value range: 0-65535.

gapH

the height of media gap/black line/bore hole in dots.

Value: 0-65535.

The value of gapH is related to the position of labels.

**Gap mode:** By default, set gapH to gap length. The printer is in Gap mode and parameters are set with the media AutoSense.

**Black Line Mode:** Set gapH to B plus black line thickness in dots.

**Continuous Media Mode:** Set gapH to 0 (zero). The transmissive (gap) sensor will be used to detect the end of media.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_SetLabelHeight (160, 24);
```

## ZM\_SetLabelWidth

**Description:**

This function is used to set the label's width.

**Syntax:**

**int ZM\_SetLabelWidth** (unsigned int lwidth);

**Parameters:**

lwidth

label width in dots .

**Return Value**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

ZM\_SetLabelWidth (250);

## ZM\_SetCoordinateOrigin

**Description:**

This function is used to set/change the coordinate origin point.

**Syntax:**

```
int ZM_SetCoordinateOrigin(  
    unsigned int  px,  
    unsigned int  py  
);
```

**Parameters:**

px:

X coordinate moved distance in dots.

Py

Y coordinate moved distance in dots.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_SetCoordinateOrigin (12,23);
```

## ZM\_DrawBarcode

### Description:

This function is used to print a 1D barcode, the content must be a constant string.

### Syntax:

```
int ZM_DrawBarcode (  
    unsigned int  px,  
    unsigned int  py,  
    unsigned int  pdirec,  
    LPTSTR  pCode,  
    unsigned int  NarrowWidth,  
    unsigned int  pHorizontal,  
    unsigned int  pVertical,  
    char ptext,  
    LPTSTR pstr  
);
```

### Parameters:

px

X coordinate in dots.

py

Y coordinate in dots.

pdirec

Select printing Direction.

0 - No rotation; 1 - 90°rotation; 2 - 180°rotation; 3 - 270°rotation.

pCode

1D Barcode selection.

P4 Value	Barcode Type
0	Code 128 UCC (shipping container code)
1	Code 128 AUTO
1A	Code 128 subset A
1B	Code 128 subset B
1C	Code 128 subset C
1E	UCC/EAN
1F	EAN 128 subset A
1G	EAN 128 subset B
1H	EAN 128 subset C

2D	Interleaved 2 of 5 with human readable check digit
2G	German Postcode
2M	Matrix 2 of 5
2U	UPC Interleaved 2 of 5
3	Code 3 of 9
3C	Code 3 of 9 with check sum digit
3E	Extended Code 3 of 9
3F	Extended Code 3 of 9 with check sum digit
9	Code93
E30	EAN-13
E32	EAN-13 2 digit add-on
E35	EAN-13 5 digit add-on
E80	EAN-8
E82	EAN-8 2 digit add-on
E-85	EAN-8 5 digit add-on
K	Codabar
P	Postnet
UA0	UPC-A
UA2	UPC-A 2 digit add-on
UA5	UPC-A 5 digit add-on
UE0	UPC-E
UE2	UPC-E 2 digit add-on
UE5	UPC-E 5 digit add-on

NarrowWidth

Narrow bar width in dots.

pHorizontal

Wide bar width in dots.

pVertical

Barcode height in dots.

ptext

N - no print the readable characters under barcode;

B - print the readable characters under barcode.

pstr

A character string (length is 1 to 100) **It must begin and ended with ""**.

Example: "123456".

**Return Value:**

0 : Succeeds.



Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_DrawBarcode (50,30,0,'1A',1,1,10,'N',"123456");
```

## ZM\_DrawBarcodeEx

### Description:

This function is used to print a 1D barcode, the content can be constant, counter value, variable or combined string.

### Syntax:

**int ZM\_DrawBarcodeEx (**

    unsigned int px,  
    unsigned int py,  
    unsigned int pdirec,  
    LPTSTR pCode,  
    unsigned int NarrowWidth,  
    unsigned int pHorizontal,  
    unsigned int pVertical,  
    char ptext,  
    LPTSTR pstr,  
    BOOL Variable

**);**

### Parameters:

px

X coordinate in dots.

py

Y coordinate in dots.

pdirec

Select printing Direction.

0 - No rotation; 1 - 90°rotation; 2 - 180°rotation; 3 - 270°rotation.

pCode

1D Barcode selection.

P4 Value	Barcode Type
0	Code 128 UCC (shipping container code)
1	Code 128 AUTO
1A	Code 128 subset A
1B	Code 128 subset B
1C	Code 128 subset C
1E	UCC/EAN
1F	EAN 128 subset A

1G	EAN 128 subset B
1H	EAN 128 subset C
2D	Interleaved 2 of 5 with human readable check digit
2G	German Postcode
2M	Matrix 2 of 5
2U	UPC Interleaved 2 of 5
3	Code 3 of 9
3C	Code 3 of 9 with check sum digit
3E	Extended Code 3 of 9
3F	Extended Code 3 of 9 with check sum digit
9	Code93
E30	EAN-13
E32	EAN-13 2 digit add-on
E35	EAN-13 5 digit add-on
E80	EAN-8
E82	EAN-8 2 digit add-on
E-85	EAN-8 5 digit add-on
K	Codabar
P	Postnet
UA0	UPC-A
UA2	UPC-A 2 digit add-on
UA5	UPC-A 5 digit add-on
UE0	UPC-E
UE2	UPC-E 2 digit add-on
UE5	UPC-E 5 digit add-on

NarrowWidth

Narrow bar width in dots.

pHorizontal

Wide bar width in dots.

pVertical

Barcode height in dots.

pText

N - no print the readable characters under barcode;

B - print the readable characters under barcode.

pStr

A combined character string (length is 1 to 100), it can be combined with "DATA", Cn, Vn.

"DATA": A data field string, **it must begin and ended with ""**.

Example: "123456".

Cn: A counter value. Refer to C command.

Vn: A variable string. Refer to V command.

Example: `"data1"CnVn"data2"`.

#### Variable

TRUE: Indicating the string contain the Counter value or Variables string data.

Example: ZM\_DrawBarcodeEx (50,30,0,'1A',1,1,10,'N',C1" is "V2,TRUE);

FALSE: It's a fixed data string.

Example: ZM\_DrawBarcodeEx (50,30,0,'1A',1,1,10,'N',"123456",FALSE);

#### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

#### Example:

```
ZM_DrawBarcodeEx (50,30,0,'1A',1,1,10,'N',"123456",TRUE);
```

```
ZM_DrawBarcodeEx (50,30,0,'1A',1,1,10,'N',"123456",FALSE);
```

```
ZM_DrawBarcodeEx (50,30,0,'1A',1,1,10,'N',C2,TRUE);
```

```
ZM_DrawBarcodeEx (50,30,0,'1A',1,1,10,'N',V1,TRUE);
```

```
ZM_DrawBarcodeEx (50,30,0,'1A',1,1,10,'N',C1" is "V2,TRUE);
```

## ZM\_DrawBar2D\_DATAMATRIX

### Description:

This function is used to print a DataMatrix barcode.

### Syntax:

```
int ZM_DrawBar2D_DATAMATRIX(  
    unsigned int x,  
    unsigned int y,  
    unsigned int w,  
    unsigned int v,  
    unsigned int o,  
    unsigned int m,  
    LPTSTR pstr  
);
```

### Parameters:

x	X coordinate in dots.
y	Y coordinate in dots.
w	Maximum print width in dots.
v	Maximum print height in dots.
o	Print direction positioning. '0' means 0°, '1' means 90°, '2' means 180°, '3' means 270°
m	Module width in dots. Value: 1 - 9.
pstr	ASCII data or Binary data bytes Represents a fixed data field.

### Return Value:

0 : Succeeds.  
Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

## ZM\_DrawBar2D\_QR

### Description:

This function is used to print a DataMatrix barcode.

### Syntax:

```
int ZM_DrawBar2D_QR(  
    unsigned int x,  
    unsigned int y,  
    unsigned int w,  
    unsigned int v,  
    unsigned int o,  
    unsigned int r,  
    unsigned int m,  
    unsigned int g,  
    unsigned int s,  
    LPTSTR pstr  
);
```

### Parameters:

x	X coordinate in dots.
y	Y coordinate in dots.
w	Maximum print width in dots.
v	Maximum print height in dots.
o	Print direction positioning. '0' means 0°, '1' means 90°, '2' means 180°, '3' means 270°
r	Module width in dots. Value: 1 - 9.
m	Start mode. Value: 0 – 4. 0: Numeric

---

- 1: AlphaNumeric
- 2: Binary
- 3: Kanji
- 4: AUTO

g

ECC.

Value: 0 – 3.

- 0: L
- 1: M
- 2: Q1
- 3: H1

s

Mask

Value: 0 – 8.

- 0: mask 000
- 1: mask 001
- 2: mask 010
- 3: mask 011
- 4: mask 100
- 5: mask 101
- 6: mask 110
- 7: mask 111
- 8: AUTO

pstr

ASCII data or Binary data bytes Represents a fixed data field.

#### **Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

## ZM\_DrawBar2D\_MaxiCode

### Description:

This function is used to print a MatrixCode barcode.

### Syntax:

```
int ZM_DrawBar2D_MaxiCode(  
    unsigned int x,  
    unsigned int y,  
    unsigned int m,  
    unsigned int u,  
    LPTSTR pstr  
);
```

### Parameters:

x

X coordinate in dots.

y

Y coordinate in dots.

m

Mode.

Value: 2 - 4

u

UPS format selection.

Value: 1 It's UPS format, other number means It isn't UPS format.

pstr

Mode Dependent Data Format.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.



## ZM\_DrawBar2D\_Pdf417

### Description:

This function is used to print a PDF417 barcode.

### Syntax:

```
int ZM_DrawBar2D_Pdf417(  
    unsigned int x, unsigned int y,  
    unsigned int w, unsigned int v,  
    unsigned int s, unsigned int c,  
    unsigned int px, unsigned int py,  
    unsigned int r, unsigned int l,  
    unsigned int t, unsigned int o,  
    LPTSTR pstr  
);
```

### Parameters:

x

X coordinate in dots.

y

Y coordinate in dots.

w

Maximum printing width, in dots.

v

Maximum printing height, in dots.

s

correction degrees.

Value: 0 - 8

c

compression degrees.

Value: 0 or 1.

px

Module width

Value: 2 - 9

py

Module height.

Value: 4 – 99.

r

Max row count.

l

Max column count.

t

Truncation flag

'0' means normal and

'1' means truncated.

o

Select printing Direction.

0 - No rotation; 1 - 90°rotation; 2 - 180°rotation; 3 - 270°rotation.

pstr

A character string (length is 1 to 100), using

"DATA", Cn and Vn parameters.

"DATA": A data fixed string, it must begin and end with ""

Example: "123456".

Cn: A counter value. Refer to C command.

Vn: A variable string. Refer to V command.

Example: "data1"CnVn"data2".

#### **Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

#### **Example:**

```
unsigned int x,y,w,v,s,c,px,py,r,l,t,o;
```

```
LPCTSTR pstr = "ZMININFO";
```

```
x=10;y=10;w=400;v=300;s=0;c=0;px=3;py=7;r=10;l=2;t=0;o=0;
```

```
ZM_DrawBar2D_Pdf417 (x,y,w,v,s,c,px,py,r,l,t,o,pstr);
```

## ZM\_DrawTextTrueTypeW

### Note:

**Be sure to install the ZMIN printer driver prior to using ZM\_DrawTextTrueTypeW()**

### Description:

**This function is used to print a line of TrueType Font string, and the character width and height can be adjusted.**

### Syntax:

```
int ZM_DrawTextTrueTypeW(  
    int x,  
    int y,  
    int FHeight,  
    int FWidth,  
    LPCTSTR FType,  
    int Fspin,  
    int FWeight,  
    BOOL FItalic,  
    BOOL FUnline,  
    BOOL FStrikeOut,  
    LPCTSTR id_name,  
    LPCTSTR data  
);
```

### Parameters:

x

X coordinate in dots.

y

Y coordinate in dots.

FHeight

Font type height in dots.

FWidth

Font type width in dots.

**\* If you want to print the normal scale font, set FWidth to 0.**

FType

Font type name.

Fspin

Font rotation.

1 - No rotation; 2 - 90°rotation; 3 - 180°rotation; 4 - 270°rotation.

Fweight

Font thickness.

0 and 400 : 400 standard,

100 : extra thin,

200 : tiny thin,

300 : thin,

500 : middling,

600 : half thick,

700 : thick,

800 : extra thick,

900 : bolder.

Fitalic:

Italic.

0: FALSE,

1: TRUE.

Funline

Add base line.

0: FALSE,

1: TRUE.

FstrikeOut

Add strikethrough

0: FALSE,

1: TRUE.

id\_name

Identify the name. True type characters will be transferred to PCX data and store to the printer as the name of id\_name. users can Call id\_name to print this true type characters by ZM\_DrawPcxGraphics( )for several times before powering off.

**Note: It don't allow setting the same id\_name for different strings.**

data

The contents of string.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

Print "Arial" fonts for 3mm height:

---

For the 203 DPI models printer, FHeight is  $3 / 0.125 = 24\text{dots}$ ;

For the 300 DPI models printer, FHeight is  $3 / 0.08 = 38\text{dots}$  (round).

```
ZM_DrawTextTrueTypeW (30,35,24,0,"Arial",4,400,0,0,0,"A1","AaBbCc");
```

## ZM\_DrawText

### Description:

This function is used to print text , the content must be a constant string.

### Syntax:

```
int ZM_DrawText (
    unsigned int  px,
    unsigned int  py,
    unsigned int  pdirec,
    unsigned int  pFont,
    unsigned int  pHorizontal,
    unsigned int  pVertical,
    char ptext,
    LPTSTR pstr
);
```

### Parameters:

x

X coordinate in dots.

y

Y coordinate in dots.

pdirec

pdirec

Select printing Direction.

0 - No rotation; 1 - 90°rotation; 2 - 180°rotation; 3 - 270°rotation.

pFont

Selects internal fonts or soft fonts.

1 - 5: internal fonts; 'a': 24\*24 simple Chinese fonts in printer.

Value	Description
1	Font 1
2	Font 2
3	Font 3
4	Font 4
5	Font 5
'a'	24*24 dot matrix Chinese font

pHorizontal

Horizontal multiplier expands the text horizontally.

Value : 1-24

pVertical

Vertical multiplier expands the text vertically.

Value: 1-24

pText

Choosing 'N' prints normal text (i.e. black characters on a white background)

Choosing 'R' prints reversed text (i.e. white characters on a black background)

pstr

A character string (length is 1 to 100) **It must begin and ended with ""**.

Example: "123456".

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_DrawText (50,30,0,2,1,1,'N',"123456789");
```

## ZM\_DrawTextEx

### Description:

This function is used to print text, the content can be constant, counter value, variable or combined string.

Syntax:

```
int ZM_DrawTextEx(
    unsigned int px,
    unsigned int py,
    unsigned int pdirec,
    unsigned int pFont,
    unsigned int pHorizontal,
    unsigned int pVertical,
    char ptext,
    LPTSTR pstr ,
    BOOL Variable
);
```

### Parameters:

x

X coordinate in dots.

y

Y coordinate in dots.

pdirec

pdirec

Select printing Direction.

0 - No rotation; 1 - 90°rotation; 2 - 180°rotation; 3 - 270°rotation.

pFont

Selects internal fonts or soft fonts.

1 - 5: internal fonts; 'a': 24\*24 simple Chinese fonts in printer.

Value	Description
1	Font 1
2	Font 2
3	Font 3
4	Font 4
5	Font 5
'a'	24*24 dot matrix Chinese font



**pHorizontal**

Horizontal multiplier expands the text horizontally.

Value : 1-24

**pVertical**

Vertical multiplier expands the text vertically.

Value: 1-24

**ptext**

Choosing 'N' prints normal text (i.e. black characters on a white background)

Choosing 'R' prints reversed text (i.e. white characters on a black background)

**pstr**

A combined character string (length is 1 to 100) , it can be combined with "DATA" , Cn, Vn.

"DATA": A data field string, **it must begin and ended with ""**.

Example: **"123456"**.

Cn: A counter value. Refer to C command.

Vn: A variable string. Refer to V command.

Example: **"data1"CnVn"data2"**.

**Variable**

TRUE: Indicating the string contain the Counter value or Variables string data.

Example: ZM\_DrawTextEx (50,30,0,2,1,1,'N',"Printer"C2V1"is ok.",TRUE);

FALSE: It's a fixed data string.

Example: ZM\_DrawTextEx (50,30,0,2,1,1,'N',"123456789",FALSE);

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

ZM\_DrawTextEx (50,30,0,2,1,1,'N',"123456789",TRUE);

ZM\_DrawTextEx (50,30,0,2,1,1,'N',"123456789",FALSE);

ZM\_DrawTextEx (50,30,0,2,1,1,'N',C1,TRUE);

ZM\_DrawTextEx (50,30,0,2,1,1,'N',V3,TRUE);

ZM\_DrawTextEx (50,30,0,2,1,1,'N',"Printer"C2V1"is ok.",TRUE);

**Print out combined character string which contain the counter value and variable string.(Be used in Form format)**

ZM_FormDel("TEST");	//Delete the Form "TEST"
ZM_FormDownload("TEST");	//Download the Form "TEST"
ZM_DefineCounter(0, 6, 'N', "+1", "Enter Start");	//Define a Counter name C0
ZM_DefineVariable(0,16,'N',"Variable");	//Define a Variable name V0

```
ZM_DrawTextEx (50,30,0,5,1,1,'N',V0,TRUE);    //Print the string of V0
ZM_DrawText (100,70,0,5,1,1,'N',C0,TRUE);      //Print the value of C0
ZM_FormEnd();                                  //Ends Form downloading.

ZM_ClearBuffer();
ZM_ExecForm("TEST");                          //Run Form
ZM_Download();                                //Beginning to downloading data.
ZM_DownloadInitVar("123");                    //Set the value of C0.
ZM_DownloadInitVar("456");                    //Set the string of V0
ZM_PrintLabel(2, 1);                          //Print out label.
```

## ZM\_PcxGraphicsList

**Description:**

This function force the printer to print out the list of PCX graphics that stored to RAM or flash memory from host.

**Syntax:**

```
int ZM_PcxGraphicsList (void );
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_PcxGraphicsList ( );
```

## ZM\_PcxGraphicsDel

**Description:**

This function force the printer to delete PCX graphics currently stored in RAM or flash memory.

**Syntax:**

```
int ZM_PcxGraphicsDel (LPTSTR pid);
```

**Parameters:**

pid:

Graphics name with a maximum of 16 characters or "\*".

if pid = "\*", all graphics will be deleted from RAM or flash memory.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_PcxGraphics Del ("PCX2" );
```

## ZM\_PcxGraphicsDownload

### Description:

This function force the printer to store a PCX graphics to printer.

Syntax:

```
int ZM_PcxGraphicsDownload (  
    char* pcxname,  
    char* pcxpath  
);
```

### Parameters:

pcxname

User-defined graphics name with a maximum of 16 characters. graphics can only be printed by using this name in ZM\_DrawPcxGraphics ( ) after the graphics being stored to the printer

pcxpath

The path of the PCX graphics in memory.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_PcxGraphicsDownload ("PCXA", "c:\\test1111.pcx");
```

## ZM\_BmpGraphicsDownload

### Description:

This function Convert a BMP graphics to PCX graphics, then store this PCX graphics to printer.

Syntax:

```
int ZM_BmpGraphicsDownload (  
    char* pcxname,  
    char* pcxpath,  
    int iDire  
);
```

### Parameters:

pcxname

User-defined graphics name with a maximum of 16 characters. graphics can only be printed by using this name in ZM\_DrawPcxGraphics ( ) after the graphics being stored to the printer

pcxpath

The path of the BMP graphics in memory.

iDire

Select BMP graphics' rotation.

0 - No rotation; 1 - 90°rotation; 2 - 180°rotation; 3 - 270°rotation.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_BmpGraphicsDownload ("PCXA", "c:\\test1111.bmp", 0);
```

## ZM\_DrawPcxGraphics

### Description:

This function is used to print the designated PCX graphics.

**Note: The graphics must store in the printer by using ZM\_PcxGraphicsDownload before it to be printed.**

### Syntax:

```
int ZM_DrawPcxGraphics (  
    unsigned int  px,  
    unsigned int  py,  
    LPTSTR  gname  
);
```

### Parameters:

px

X coordinate in dots.

py

Y coordinate in dots.

game

Graphics name with a maximum of 16 characters, it must be user-defined name in the ZM\_PcxGraphicsDownload().

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_DrawPcxGraphics (100,50,"PCX1");
```

## ZM\_PrintPCX

### Description:

This function is used to print a PCX graphics.

Using this function is equal to use together with ZM\_PcxGraphicsDownload() and ZM\_DrawPcxGraphics().

### Syntax:

```
int ZM_PrintPCX (  
    unsigned int px,  
    unsigned int py,  
    char* filename  
);
```

### Parameters:

px

X coordinate in dots.

py

Y coordinate in dots.

filename

The path of the PCX graphics in memory.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_PrintPCX(10,100,"c:\\phone.pcx");
```



## ZM\_DrawBinGraphics

### Description:

This function is used to print binary graphics.

Binary graphics is non-compressed graphics data. Each bit represents a dot., 0 represent print. a value of "0" prints a dot; a value of "1" does not print a dot.

### Syntax:

```
int ZM_DrawBinGraphics (  
    unsigned int  px,  
    unsigned int  py,  
    unsigned int  pbyte,  
    unsigned int  pH,  
    UCHAR* Gdata  
);
```

### Parameters:

px

X coordinate in dots.

py

Y coordinate in dots.

pbyte

Bytes for one line data. If 8 cannot divide the bits of one line data, then the bytes equal to the result add 1.

**Example:** the bytes of one line data is 2(14 bits data),

pH

Height of graphic, in dots.

Gdata

**([...raster data...])**

Binary graphic data; data size = pbyte \* pH (Bytes).

Binary data transmission sequence is left to right, up to down, example:

data transmission sequence: Line1's Byte1(0xff), Line1's Byte2(0xff), Line2's Byte1(0xe0),Line2's Byte2(0x1f), Line3's Byte1(0xff), Line3's Byte2(0xff)...

The part of the broken line is non-graphic area, and then the corresponding bit is 0.



## ZM\_DrawLineXor

### Description:

This function is used to draw straight-line (If two straight-lines intersects, use an exclusive or operation).

### Syntax:

```
int ZM_DrawLineXor(  
    unsigned int  px,  
    unsigned int  py,  
    unsigned int  pbyte,  
    unsigned int  pH  
);
```

### Parameters:

px

X coordinate in dots.

py

Y coordinate in dots.

pbyte

Horizontal length in dots.

pH

Vertical length in dots.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_DrawLineXor (100,20,5,110);
```

## ZM\_DrawLineOr

### Description:

This function is used to draw straight-line (If two straight-lines intersects, use Or operation).

Syntax:

```
int ZM_DrawLineOr(  
    unsigned int  px,  
    unsigned int  py,  
    unsigned int  plength,  
    unsigned int  pH  
);
```

### Parameters:

px

X coordinate in dots.

py

Y coordinate in dots.

plength

Horizontal length in dots.

pH

Vertical length in dots.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_DrawLineOr (100,20,5,110);
```

## ZM\_DrawDiagonal

### Description:

This function is to draw diagonal.

### Syntax:

```
int ZM_DrawDiagonal (  
    unsigned int  px,  
    unsigned int  py,  
    unsigned int  thickness,  
    unsigned int  pEx,  
    unsigned int  pEy  
);
```

### Parameters:

px

X coordinate in dots.

py

Y coordinate in dots.

thickness

Set Thickness in dots.

pEx

End X coordinate in dots.

pEy

End Y coordinate in dots.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_DrawDiagonal (50,30,10,100,80);
```

## ZM\_DrawWhiteLine

### Description:

This function is used to draw a whit line.

### Syntax:

```
int ZM_DrawWhiteLine(  
    unsigned int  px,  
    unsigned int  py,  
    unsigned int  plength,  
    unsigned int  pH  
);
```

### Parameters:

px

X coordinate in dots.

py

Y coordinate in dots.

plength

Horizontal length in dots.

pH

Vertical length in dots.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_DrawWhiteLine (100,20,5,110);
```

## ZM\_DrawRectangle

### Description:

This function is used to draw rectangle.

### Syntax:

```
int ZM_DrawRectangle (  
    unsigned int  px,  
    unsigned int  py,  
    unsigned int  thickness,  
    unsigned int  pEx,  
    unsigned int  pEy  
);
```

### Parameters:

px

Horizontal start position (X) in dots.

py

Vertical start position (Y) in dots.

thickness

Line thickness in dots.

pEx

Horizontal end position (X) in dots.

pEy

Vertical end position (Y) in dots.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_DrawRectangle (50,120,5,250,150);
```

## ZM\_PrintLabel

**Description:**

This function is used to order the printer to print out label.

**Notes:**

Please don't use this function during editing the FORM, use ZM\_PrintLabelAuto() instead of it.

**Syntax:**

```
int ZM_PrintLabel (  
    unsigned int    number,  
    unsigned int    cpnumber  
);
```

**Parameters:**

number

number of labels to print.

Value:1 - 65535

cpnumber

number of copies pre label. the default value is 1 if this cpnumber have no set value.

Value:1 - 65535

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_PrintLabel (2,3);
```



## ZM\_SoftFontList

**Description:**

This function will cause the printer to print out a list of all soft fonts that are stored in RAM or FLASH memory.

**Syntax:**

```
int ZM_SoftFontList (void);
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_SoftFontList ();
```

## ZM\_SoftFontDel

**Description:**

This function force the printer to remove one or all, soft fonts stored in RAM and/or Flash memory.

**Syntax:**

```
int ZM_SoftFontDel (char pid);
```

**Parameters:**

pid:

Soft font ID.

Value: A—Z or \*

If pid = '\*',all fonts will be deleted from RAM or flash memory.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_SoftFontDel ('A');
```

## ZM\_FormEnd

**Description:**

This function is used to end a form store, it must be used together with the ZM\_FormDownload.

**Syntax:**

```
int ZM_FormEnd (void );
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_FormDownload ( "Form1" ) ;  
...  
ZM_FormEnd ( );
```

## ZM\_FormList

**Description:**

This function force the printer to print out a list of forms that have been stored.

**Syntax:**

```
int ZM_FormList (void );
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_FormList ( );
```

## ZM\_FormDel

**Description:**

This function force the printer to delete forms currently stored.

**Syntax:**

```
int ZM_FormDel (  
    LPTSTR pid  
);
```

**Parameters:**

pid

Form name with a maximum of 16 characters.

If pid = "\*",all forms will be deleted from RAM or flash memory.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_FormDel ("FORMNAME");
```

## ZM\_FormDownload

**Description:**

This function is used to store a form to the printer, it must be used together with the ZMIN\_FormEnd.

**Syntax:**

```
int ZM_FormDownload (  
    LPTSTR pid  
);
```

**Parameters:**

pid

User-defined form name with a maximum of 16 characters. Use this form prior to execute this function after storing the form to the printer

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_FormDownload ("FORMNAME");
```

## ZM\_ExecForm

**Description:**

This function is used to execute the designated form.

**Syntax:**

```
int ZM_ExecForm (  
    LPTSTR pid  
);
```

**Parameters:**

pid

Form name with a maximum of 16 characters.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_ExecForm ("FORM1");
```

## ZM\_PrintLabelAuto

**Description:**

This function is used to execute printing job automatically. It suggests that use this function when variables or counter values exist. The printer prints the form, as soon as all variable data have been input.

**Notes: Only use in FORM.**

**Syntax:**

```
int ZM_PrintLabelAuto (  
    unsigned int  number,  
    unsigned int  cpnumber  
);
```

**Parameters:**

number

number of labels to print.

Value:1 - 65535

cpnumber

number of copies pre label. the default value is 1 if this cpnumber have no set value.

Value:1 - 65535

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_PrintLabelAuto (2,3);
```



## ZM\_DefineCounter

### Description:

This function is used to define a counter variable.

### Syntax:

```
int ZM_DefineCounter (
    unsigned int id,
    unsigned int maxNum,
    char ptext,
    LPTSTR pstr,
    LPTSTR pMsg
);
```

### Parameters:

id

Counter ID, acceptable values are from 0 to 9.

maxNum

Maximum digit number, acceptable values are from 1 to 40.

ptext

Justification code. L for left justification, R for right justification, C for centralization and N for no justification.

Pstr

Counter change rule : "+" or "-" sign followed by a single digit of 1 – 9, then followed by a change symbol (i.e. D - decimal base, B - binary system, O - octonary number system, H - hexadecimal system, X - user defined pattern, to a maximum of 64 characters. )

#### Example Step values:

" +1 " = Increases each time by 1, according to Decimal base computation. Example: 1234, 1235, 1236, ....

" +3D " = Increases each time by 3, according to Decimal base computation. Example: 1234, 1235, 1236, ....

" -1B " = Decreases each time by 1, according to Binary computation. Example: 1111, 1110, 1101, ....

" -4O " = Decreases each time by 4, according to Octonary number system computation.  
Example: 1234, 1230, 1224, ....

" -6H " = Decreases each time by 1, according to hexadecimal base computation. Example: 1234, 122E, 1228, ....

“+3X”= Increase each time by 3, according to a user-defined pattern. Example: In user-defined rule: TE2DOKLU046MNY37, the starting value is “T062”, followed by T062, T06K, T060,....

pMsg

A text string that will be sent to LCD or KDU.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_DefineCounter (0,6,'N','+1','\nEnter\ Code:');
```

## ZM\_DefineVariable

### Description:

This function is used to define variable in the FORM.

**Note: It use this function only when editing FORM.**

### Syntax:

```
int ZM_DefineVariable (  
    unsigned int  pid,  
    unsigned int  pmax,  
    char  porder,  
    LPTSTR  pmsg  
);
```

### Parameters:

pid

Variable ID number, value range: 00—99;

pmax

Maximum number of characters, value range: 1—99.

If you use KDU, the length should limited to under 16 characters.

porder

Field Justification; L-left justification, R- right justification, C-center, N-no justification.

pmsg

Remind contents. Will be sent to KDU or displayed on LCD display of the printer.

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_DefineVariable (0,16,L,"Enter Title:");
```

## ZM\_Download

**Description:**

This function is used to download variable or counter variable to the printer.  
Please refer to the command “?” of the PCLE.

**Syntax:**

**int ZM\_Download**(void);

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

ZM\_Download( );

## ZM\_DownloadInitVar

**Description:**

This function is used to initialize the Variable or Counter value.

Note: It must be execute ZM\_Download() before use this function.

**Syntax:**

```
int ZM_DownloadInitVar (  
    LPTSTR pst  
);
```

**Parameters:**

pst

a string for Variable;

a numeric string for Counter.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_DownloadInitVar("123456");
```

## ZM\_PrintConfiguration

**Description:**

This function is used to print out Self Test label.

**Syntax:**

```
int ZM_PrintConfiguration ( );
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_PrintConfiguration ( );
```

## ZM\_Reset

**Description:**

This function is used to reset the printer.

**Note: The reset function cannot be used within a stored form.**

**Syntax:**

```
int ZM_Reset (void);
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_Reset( );
```

## ZM\_SetPrinterState

**Description:**

This function is used to set the printer's working state.

**Syntax:**

```
int ZM_SetPrinterState(char state);
```

**Parameters:**

state

D: Enable direct thermal printing (without ribbon).

P: Enable continuous printout.(default)

L: After printing a label the printer will stop, requiring user input to print the next label. Input determined by:

1. By pressing the "feed" button for each label to be printed.

2. Will continue automatically after previously printed label is removed (with peeler kit installed)

C: Enable Cutting mode. (Only with cutter kit installed)

N: Enable Peeler mode. (Only with peeler kit installed)

**Notes:**

1. The cutter and dispenser cannot be enabled at the same time.
2. Once the options are incorrectly selected, the READY light in front panel will flash. Please refer to the trouble-shooting section to correct the errors.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_SetPrinterState ('D');
```



## ZM\_FeedMedia

**Description:**

This Function force the printer to feed out a label.

**Syntax:**

```
int ZM_FeedMedia (void);
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_FeedMedia ( );
```

## ZM\_MediaDetect

**Description:**

This function is used to force a label calibration.

**Syntax:**

```
int ZM_MediaDetect (void);
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_MediaDetect ( );
```

## ZM\_CutPage

**Description:**

This function is used to set cutter's working circle (The cutter will start to cut labels once the numbers of labels have been printed)

**Syntax:**

```
int ZM_CutPage (  
    UINT page  
);
```

**Parameters:**

page

number of label have printed before cutter work.

Value:1 – 999; default value is 1.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_CutPage(1);
```

## ZM\_EnableFIASH

**Description:**

This function is used to enable Flash memory.

the data sent to the printer will be stored to the flash memory when use this function.

**Note: This function is enable Depend on model of printer.**

**Syntax:**

```
int ZM_EnableFIASH ( void);
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_EnableFIASH ( );
```

## ZM\_DisableFLASH

**Description:**

This function is used to disable Flash memory.

The data sent to the printer will be stored to the SDRAM when use this function.

**Syntax:**

```
int ZM_DisableFLASH (void);
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_DisableFLASH ( );
```

## ZM\_BinGraphicsList

**Description:**

This function force the printer to print out the list of Binary graphics that stored to RAM or flash memory from host.

**Syntax:**

```
int ZM_BinGraphicsList (void );
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_BinGraphicsList ( );
```

## ZM\_BinGraphicsDel

**Description:**

This function force the printer to delete one or all of Binary graphics currently stored in RAM or flash memory.

**Syntax:**

```
int ZM_BinGraphicsDel (  
    LPTSTR pid  
);
```

**Parameters:**

pid

Graphics name with a maximum of 16 characters or “\*”.

if pid = “\*”, all graphics will be deleted from RAM or Flash memory.

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_BinGraphicsDel (“Bin2” );
```

## ZM\_BinGraphicsDownload

### Description:

This function force the printer to store a Binary graphics to printer.

### Syntax:

```
int ZM_BinGraphicsDownload (  
    char* name,  
    unsigned int pbyte,  
    unsigned int pH,  
    UCHAR * Gdata  
);
```

### Parameters:

name

User-defined graphics name with a maximum of 16 characters. graphics can only be printed by using this name in ZM\_RecallBinGraphics() after the graphics being stored to the printer

pbyte

Bytes for one line data. If 8 cannot divide the bits of one line data, then the bytes equal to the result add 1.

**Example:** the bytes of one line data is 2(14 bits data),

pH

Height of graphic, in dots.

Gdata

**([...raster data...])**

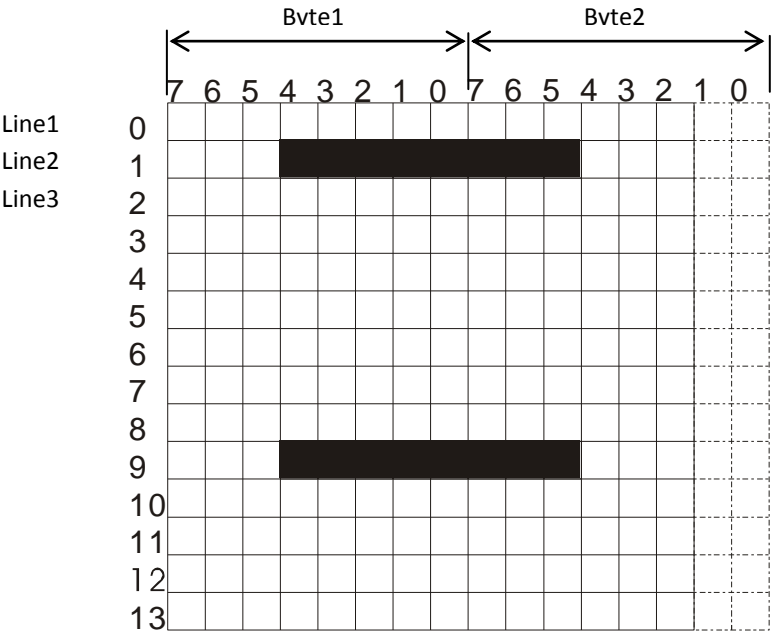
Binary graphic data; data size = pbyte \* pH (Bytes).

Binary data transmission sequence is left to right, up to down, example:

data transmission sequence: Line1's Byte1(0xff), Line1's Byte2(0xff), Line2's Byte1(0xe0),Line2's Byte2(0x1f), Line3's Byte1(0xff), Line3's Byte2(0xff), ...

The part of the broken line is non-graphic area, and then the corresponding bit is 0.





**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
char buf[] = {0xff,0xff,0xe0,0x1f,0xff,0xff...};  
ZM_BinGraphicsDownload ("BinA", 3, 24, buf);
```

## ZM\_RecallBinGraphics

### Description:

This function is used to print the downloaded Binary graphics.

**Note: The graphics must store in the printer by using ZM\_BinGraphicsDownload before it to be printed.**

### Syntax:

```
int ZM_RecallBinGraphics (  
    unsigned int px,  
    unsigned int py,  
    char* name  
);
```

### Parameters:

px

X coordinate in dots.

py

Y coordinate in dots.

name

Graphics name with a maximum of 16 characters, it must be user-defined name in the ZM\_BinGraphicsDownload().

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_RecallBinGraphics(10,100,"BinA");
```

## ZM\_DisableErrorReport

**Description:**

This function is used to Disable the printer's status report via the COM port.

**Syntax:**

```
int ZM_DisableErrorReport (void);
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_DisableErrorReport( );
```

## ZM\_EnableErrorReport

**Description:**

This function is used to Enable the printer's status report via the COM port.

**Syntax:**

```
int ZM_EnableErrorReport (void);
```

**Parameters:**

None

**Return Value:**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_DisableErrorReport( );
```

**Remark:**

The printer sends its feedback to the PC via the COM port.

If an error occurs, the printer will send a NACK (15H), followed by the error number, to the host. If no errors occur, the printer will echo ACK (06H) after each P command.

Error Code	Description
0x00	No Error
0x01	Object Exceeded Label Border
0x02	Bar Code Data Length Error
0x03	Insufficient Memory to Store Data
0x04	Memory Configuration Error
0x05	RS-232 Interface Error
0x06	Paper or Ribbon Empty
0x07	Duplicate Name: Form, Graphic or Soft Font
0x08	Name Not Found: Form, Graphic or Soft Font
0x09	Not in Data Entry Mode
0x0a	Print Head Up (Open)
0x0b	Pause Mode or Paused in Peel mode
0x0c	Does not fit in area specified
0x0d	Data length too long
0x0c	PDF-417 coded data too large to fit in bar code
0x0d	
0x0e	

## ZM\_ErrorReport

### Description:

Use this function to get printer error/status code immediately. This function must be use the COM port to receive send back status code.

This function must be execute after ZM\_ClearBuffer().

Before use the function, please confirm to use ClosePort() to close COM port if other application have opened the COM port.

It strong suggested to use ZM\_ErrorReportEx() instead of this function.

### Syntax:

**int ZM\_ErrorReport** (int wPort, int rPort, DWORD BaudRate, BOOL HandShake);

### Parameters:

wPort

must be 0.

rPort

the COM port of PC which receive the Error/Status code.

Value: 1 – 255

Example:

1 : Receiving codes from COM1;

2 : Receiving codes from COM2;

...

255: Receiving codes from COM255;

BaudRate

Set baud rate value.

value: 9600,19200,38400,57600;

HandShake

HandShaking flag;

TRUE: Enable HandShaking;

FALSE: Disable HandShaking.

### Return Value:

>=0 : Refer to <Tabl 1>.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

**Getting the status code of ZMIN X1DPI printer from COM1 port:**

```

OpenPort("ZMIN X1DPI ");
....
ZM_ClearBuffer();
char buff[5] = {0};
ZM_ErrorReport (0, 1, 38400, FALSE);
.....
ClosePort();

```

**Remark:**

The printer will report 4 bytes back to host in the following format:

0xXX XX 0x0d 0x0a : Error/Status code <CR><LF>

**<Tabl 1>**

Error/Status code	Description	ZM_ErrorReport() return value	ErrorCode
00	No Error	0	"00"
01	Error Command	1	"01"
82	Out of Ribbon	82	"82"
83	Out of Media	83	"83"
86	Cutter aren't installed	86	"86"
87	Print Head Up (Open)	87	"87"
88	Print Pause	88	"88"
99	Other Error	99	"99"

## ZM\_ErrorReportEx

### Description:

Use this function to get printer's error/status code , it will return when timeout , whether the printer send back the codes or no. This function must be use the COM port to receive send back status code.

Before use the function, please confirm to use ClosePort() to close COM port if other application have opened the COM port.

This function must be execute after ZM\_ClearBuffer().

### Syntax:

```
int ZM_ErrorReport (  
    int wPort,  
    int rPort,  
    DWORD BaudRate,  
    BOOL HandShake,  
    int TimeOut  
);
```

### Parameters:

wPort

must be 0.

rPort

the COM port of PC which receive the Error/Status code.

Value: 1 – 255

Example:

1 : Receiving codes from COM1;

2 : Receiving codes from COM2;

...

255: Receiving codes from COM255;

BaudRate

Set baud rate value.

value: 9600,19200,38400,57600;

HandShake

HandShaking flag;

TRUE: Enable HandShaking;

FALSE: Disable HandShaking.

Timeout

Set the receiving time out timer in 100ms.

### Return Value:

>=0 : Refer to <Tabl 1>.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

**Getting the status code of ZMIN X1DPI printer from COM1 port, waiting 2 second then timeout:**

```
OpenPort("ZMIN X1DPI ");
....
ZM_ClearBuffer();
char buff[5] = {0};
ZM_ErrorReport (0, 1, 38400, FALSE,20);
.....
ClosePort();
```

### Remark:

The printer will report 4 bytes back to host in the following format:

0xXX XX 0x0d 0x0a : Error/Status code <CR><LF>

<Tabl 1>

Error/Status code	Description	ZM_ErrorReport() return value	ErrorCode
00	No Error	0	"00"
01	Error Command	1	"01"
82	Out of Ribbon	82	"82"
83	Out of Media	83	"83"
86	Cutter aren't installed	86	"86"
87	Print Head Up (Open)	87	"87"
88	Print Pause	88	"88"
99	Other Error	99	"99"



## ZM\_SetPagePrintCount

### Description:

**Notice: Don't use this function when send PCLE Compatible commands.**

This function is used to set windows driver's copies of page.

This function is different to ZM\_PrintLable(),ZM\_PrintLable().

Use this function only when you want to print image of printer which have encode from other applications.

### Syntax:

```
int ZM_SetPagePrintCount(  
    unsigned int  number,  
    unsigned int  cpnumber  
);
```

### Parameters:

number

number of labels to print.

Value:1 - 65535

cpnumber

number of copies pre label. the default value is 1 if this cpnumber have no set value.

Value:1 - 65535

### Return Value:

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

### Example:

```
ZM_SetPagePrintCount(1,1);
```

## ZM\_WritePrinter

**Description:**

This Function force to start send data to printer. The data can be any ascii codes have stored in DLL's buffer.

**Syntax:**

```
int ZM_WritePrinter();
```

**Parameters:**

None

**Return Value**

0 : Succeeds.

Other returns: Please refer to chapter: ZMWIN.DLL erroneous return value specification.

**Example:**

```
ZM_WritePrinter();
```

## ZMWIN.dll error run return code lists

- 1000  
to  
-1030 : It can't open the communication port.
- 1031  
to  
-1037 : Error occurs while reading data from the COM port.
- 1040 : Error occurs while reading data from the COM port.  
-1041 : Error occurs while reading data from the COM port.  
-1042 : The Baud of COM port was an error value.
- 2992 : Fail to executed OpenPort(), Please confirm whether the printer driver installed or no!  
-3001 : ZM\_GetInfo execution error;  
-3002 : ZM\_DrawText execution error;  
-3003 : ZM\_DrawText parameter error;  
-3004 : ZM\_DrawText or ZM\_DrawBarcode's pdirec parameter error;  
-3005 : ZM\_DrawText pFont parameter error;  
-3006 : ZM\_DrawText's pHorizontal or pVertical parameter error;  
-3007 : NULL;  
-3008 : ZM\_DrawBarcode execution error;  
-3009 : ZM\_DrawBarcode parameter error;  
-3010 : ZM\_DefineCounter execution error;  
-3011 : ZM\_DefineCounter parameter error;  
-3012 : ZM\_SetDarkness execution error;  
-3013 : ZM\_DS execution error;  
-3014 : ZM\_SoftFontList execution error;  
-3015 : ZM\_SoftFontDel parameter error;  
-3016 : ZM\_SoftFontDel execution error;  
-3017 : ZM\_FormEnd execution error;  
-3018 : ZM\_FormList execution error;  
-3019 : ZM\_FormDel Allocate memory error;  
-3020 : ZM\_FormDel parameter error;  
-3021 : ZM\_FormDel execution error;  
-3022 : ZM\_ExecForm Allocate memory error;  
-3023 : ZM\_ExecForm parameter error;  
-3024 : ZM\_ExecForm execution error;  
-3025 : ZM\_FormDownload Allocate memory error;

- 3026 : ZM\_FormDownload parameter error;
  - 3027 : ZM\_FormDownload execution error;
  - 3028 : ZM\_DrawPcxGraphics Allocate memory error;
  - 3029 : ZM\_DrawPcxGraphics parameter error;
  - 3030 : ZM\_DrawPcxGraphics execution error;
  - 3031 : ZM\_PcxGraphicsList execution error;
  - 3032 : ZM\_PcxGraphicsDel Allocate memory error;
  - 3033 : ZM\_PcxGraphicsDel parameter error;
  - 3034 : ZM\_PcxGraphicsDel execution error;
  - 3035 : ZM\_PcxGraphicsDownload Allocate memory error;
  - 3036 : ZM\_PcxGraphicsDownload can't open the file;
  - 3037 : ZM\_PcxGraphicsDownload execution error;
  - 3038 : ZM\_DrawBinGraphics execution error;
  - 3039 : ZM\_DrawBinGraphics execution error;
  - 3040 : NULL;
  - 3041 : ZM\_DisableCircumgyrate execution error;
  - 3042 : ZM\_EnableCircumgyrate execution error;
  - 3043 : ZM\_DrawLineXor execution error;
  - 3044 : ZM\_DrawLineOr execution error;
  - 3045 : ZM\_DrawDiagonal execution error;
  - 3046 : ZM\_DrawWhiteLine execution error;
  - 3047 : ZM\_ClearBuffer execution error;
  - 3048 : ZM\_SetPrinterState execution error;
  - 3049 : ZM\_SetPrinterState parameter error;
  - 3050 : ZM\_PrintLabel execution error;
  - 3051 : ZM\_PrintLabel parameter error;
  - 3052 : ZM\_PrintLabelAuto execution error;
  - 3053 : ZM\_PrintLabelAuto parameter error;
  - 3054 : ZM\_SetLabelHeight execution error;
  - 3055 : ZM\_SetLabelWidth execution error;
  - 3056 : ZM\_SetCoordinateOrigin execution error;
  - 3057 : ZM\_SetPrintSpeed execution error;
  - 3058 : ZM\_SetPrintSpeed parameter error;
  - 3059 : ZM\_PrintConfiguation execution error;
  - 3060 : ZM\_DisableErrorReport execution error;
  - 3061 : ZM\_EnableErrorReport execution error;
  - 3062 : ZM\_DefineVariable execution error;
  - 3063 : ZM\_DefineVariable parameter error;
  - 3064 : ZM\_DefineVariable parameter error;
-

- 
- 3065 : ZM\_DrawRectangle execution error;
  - 3066 : ZM\_Y execution error;
  - 3067 : ZM\_Y parameter error;
  - 3068 : ZM\_SetDirection execution error;
  - 3069 : ZM\_SetDirection parameter error;
  - 3070 : ZM\_EnableFIASH execution error;
  - 3071 : ZM\_DisableFLASH execution error;
  - 3072 : ZM\_Download execution error;
  - 3073 : ZM\_Reset execution error;
  - 3074 : ZM\_BackFeed execution error;
  - 3075 : Allocate memory of structure PCX HEAD error;
  - 3076 : Allocate memory of PCX data error;
  - 3077 : Can't save current path of file;
  - 3078 : Can't create PCX graphics file;
  - 3079 : Can't save PCX data;
  - 3080 : ZM\_PrintPCX execution error;
  - 3081 : Can't create the PrinterDC;
  - 3082 : Allocate memory of bitmap error;
  
  - 3083 : ZM\_BinGraphicsDownload execution error;
  - 3084 : ZM\_BinGraphicsDel execution error;
  - 3085 : ZM\_BinGraphicsList execution error;
  - 3087 : ZM\_RecallBinGraphics execution error;
  - 3088 : ZM\_RecallBinGraphics: The length of "name" was too long;
  
  - 3089 : ZM\_UserFeed execution error;
  - 3090 : ZM\_UserBackFeed execution error;
  - 3092 : ZM\_ErrorReport can't write data to the communication port;
  - 3100 : SetCommPort can't modify the parameter of COM port.
  
  - 3101 : ZM\_CutPage execution error;
  - 3102 : ZM\_FeedMedia execution error;
  - 3103 : ZM\_MediaDetect execution error;
  
  - 3200 to -3400 : The communication port isn't opened or has closed;
  
  - 3250 : ZM\_DrawBar2D\_Pdf417() parameter error;
  - 3251 to -3257 : The communication port isn't opened or has closed;
-

- 3261 : The communication port was opened abnormal before execute ZM\_ErrorReport();
- 4000 : The communication port was opened abnormal when execute OpenPort ();
- 4001 : The COM port was set incorrectly when execute ZM\_ErrorReport();
- 4002 : Time out error occurred when The ZM\_ErrorReportEx() use the COM port.

