条码打印机

ZMWIN

API 函数手册

Version: 2.0.0.1

2012
深圳致明兴科技有限公司

目　录

API 函数库文件说明

名称：ZMWIN.dll

版本编号：2.X.X.X

用途

本 API 函数库为深圳致明兴科技有限公司条码标签打印机的用户提供一组函数，为编写基于 Windows9X，NT，2000，XP,WIN X 等操作系统的应用程序提供便利。

本 API 函数库仅支持本公司产品。

本 API 函数与 ZMIN PCLE API 函数的区别是支持客户以打印机名称打开通讯端口传送打印机编程语言。

缩略语对照

PCLE：深圳致明兴科技有限公司的第一套打印机编程语言（Printer Command Language E）。

API：应用程序编程接口 (Application Program Interface)。

Dots：像素（pixel）是一种计算机科学技术尺寸单位，原指电视图像成像的最小单位，在打印机领域表示打印机的最小打印成像单位：1dot 等于一英寸除以打印机的最大分辨率。

- 对于 203DPI 的打印机来说，1dot = 25.4mm/203 = 0.125mm(1dot = 1000 / 203 = 5mil)；
- 对于 300DPI 的打印机来说，1dot = 25.4mm/300 = 0.085mm(1dot = 1000 / 300 = 3mil)。

TrueType Font：是基于 Windows 操作系统使用，可装卸的字体。

- 已经安装的 TrueType Font，都可以被本函数使用。

使用前须知

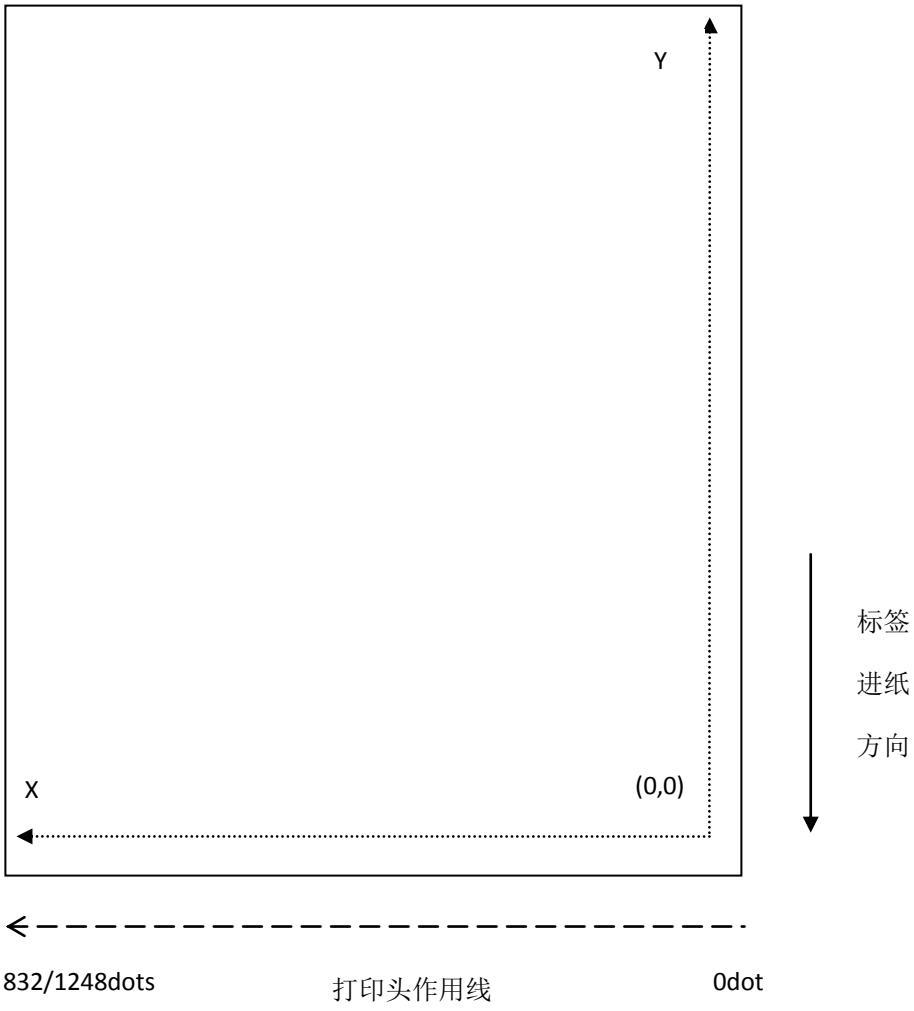
字符串

* 字符串以双引号(“)作为起始和结束标记;
* 所有打印指令和名称均区分大小写;
* <CR>为USASCII码十进制的”13”, 或十六进制的”0DH”,即”回车”符号;
* 反斜杠（\）有以下作用:

字符	输入
“	\ “
\	\\
0x00 - 0x7F	\x00 - \x7F

条码标签打印机的坐标系统

如下图所示:



函数概述列表

函数名称	说明
OpenPort	打开通讯端口。
ClosePort	关闭使用 OpenPort 函数打开的通讯端口。
ZM_ClearBuffer	清除打印机缓冲内存的内容。
SetPCComPort	设置 PC 机上串口的传输波特率。
GetErrState	检测使用 ZMWIN.DLL 里的其它函数后是否有错误产生；
ZM_GetInfo	得到本 API 函数库的版本信息。
ZM_DrawBarcode	打印一个条码。
ZM_DrawBarcodeEx	打印一个条码，内容可以是常量、序列号、变量或组合字符串.
ZM_DrawTextTrueTypeW	打印一行 TrueType Font 文字，并且文字宽度和高度可以微调。
ZM_DrawText	打印一行文本文字，内容可以是常量、序列号、变量或组合字符串。
ZM_DefineCounter	定义一个序列号变量。
ZM_SetDarkness	设置打印头发热温度
ZM_SetPrintSpeed	设置打印速度。
ZM_SetLabelHeight	设置标签的高度和定位间隙\黑线\穿孔的高度。
ZM_SetLabelWidth	设置标签的宽度。
ZM_DrawPcxGraphics	打印指定的图形。
ZM_PcxGraphicsList	打印已存储在打印机 RAM 或 FLASH 存储器里的图形名称清单。
ZM_PcxGraphicsDel	删除存储在打印机里的一个或所有图形。
ZM_PcxGraphicsDownload	存储一个 PCX 格式的图形到打印机。
ZM_BmpGraphicsDownload	先转换 BMP 到 PCX 格式，然手将 PCX 格式的图形到打印机。
ZM_PrintPCX	打印一个 PCX 格式的图形。
ZM_DrawBinGraphics	打印二进制格式的图形。
ZM_DrawLineXor	画直线(两直线相交处作”异或”处理)。
ZM_DrawLineOr	画直线(两直线相交处作”或”处理)。
ZM_DrawDiagonal	画斜线。
ZM_DrawWhiteLine	画白色直线。
ZM_SetPrinterState	设置打印机的工作状态。
ZM_PrintLabel	命令打印机执行打印工作。

ZM_SetCoordinateOrigin	设置/改变坐标原点。
ZM_PrintConfigunation	打印机器当前的设置/工作状态。
ZM_SoftFontList	打印存储在 RAM 或 FLASH 存储器里的软字体的名称清单。
ZM_SoftFontDel	删除存储在 RAM 或 FLASH 存储器里的一个或所有的软字体。
ZM_FormEnd	结束存储表格(Form)，此函数与 ZM_FormDownload 配对使用。
ZM_FormList	打印存储在打印机里的表格名称清单。
ZM_FormDel	删除存储在打印机里的一个或所有的表格。
ZM_ExecForm	运行指定的表格。
ZM_FormDownload	存储一个表格到打印机；此命令与 ZM_FormEnd 函数配对使用。
ZM_PrintLabelAuto	自动执行打印工作。
ZM_DefineVariable	定义变量。
ZM_DrawRectangle	画距形。
ZM_SetDirection	设置标签打印方向。
ZM_EnableFlASH	选择 FLASH 存储器。
ZM_DisableFLASH	取消选择 FLASH 存储器。
ZM_Download	下载变量或系列号变量。
ZM_DownloadInitVar	初始化变量或系列号变量。
ZM_FeedMedia	命令打印机走一行标签
ZM_MediaDetect	校准纸张探测器
ZM_Reset	将打印机复位。
ZM_CutPage	设置切刀的工作周期（即每打印多少页标签后，切刀才切一次纸）。
ZM_BinGraphicsList	打印已存储在打印机 RAM 或 FLASH 存储器里的图形名称清单（包括 Bin 格式和 PCX 格式的图形）。
ZM_BinGraphicsDel	删除已存储在打印机里的一个或所有图形（此图形可是 Bin 格式或 PCX 格式的）。
ZM_BinGraphicsDownload	存储一个 Bin 格式的图形到打印机。
ZM_RecallBinGraphics	打印一个已保存在打印机里的 Bin 格式图形。
ZM_DisableErrorReport	取消错误反馈。
ZM_EnableErrorReport	设置错误反馈 。
ZM_ErrorReport	发送错误查询指令到打印机并且指定串口接收和分析打印机当前错误代码。
ZM_UserFeed	令打印机进纸给定的长度。

ZM_UserBackFeed	令打印机退纸给定的长度。
ZM_SetPagePrintCount	设置某一打印页面的打印数量。
ZM_WritePrinter	送出数据到打印机。

OPENPORT

说明：

OpenPort 函数的作用是打开通讯端口。

使用本函数库其它函数之前，必须首先正确执行 OpenPort 函数。

原型：

int OpenPort(LPTSTR xxxx);

参数：

xxxx：当前所使用的打印机在WINDOWS下的名称；

返回值：

0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

```
int return = OpenPort("ZMIN X1");        //表示打开 X1 当前所选用的打印端口。
```

CLOSEPORT

说明：

ClosePort函数的作用是关闭使用OpenPort函数打开的通讯端口。

用户在对打印机操作完成之后，建议调用ClosePort关闭通讯端口；

否则用户的程序一直占用打开的通讯端口，直到程序被关闭。

原型：

void ClosePort(void);

参数：无

返回值：

无

范例：

```
ClosePort( );
```

ZM_CLEARBUFFER

说明：

ZM_ClearBuffer 函数的作用是清除打印机缓冲内存的内容。

当发送新的一张标签内容到打印机前，建议使用此命令先清空打印机图形缓存里已有的数据内容。

请不要在 FORM 的编排过程中使用此函数。

原型：

int ZM_ClearBuffer (void);

参数：无

返回值：0　　-> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_ClearBuffer ( );

SETPCCOMPORT

说明：

SetPCComPort 函数的作用是设置 PC 机上串口的传输波特率。

这个函数只有在使用串口进行通讯时才有效。

注意： 必须对应打印机上所选择串口波特率（通过调整 DIP 开关的 7，8PIN，请参阅用户手册）

原型：

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

参数：

BaudRate：要设置的串口波特率，可取值：

9600，19200，38400，57600；

HandShake：是否使用硬件握手（HandShaking）；

TRUE：硬件握手（HandShaking）有效，

FALSE：硬件握手（HandShaking）无效。

返回值：

0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

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

GETERRSTATE

说明：

GetErrState 函数的作用是检测使用 ZMWIN.DLL 里的其它函数后是否有错误产生；

错误代码请参阅“ZMWIN.dll 错误返回值解析”

这个函数必须在 ClosePort()函数前使用！

原型：

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

参数：无

返回值：

0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

示范：

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

ZM_GETINFO

说明：

ZM_GetInfo 函数作用是得到本 API 函数库的版本信息。

原型：

int ZM_GetInfo(void)

参数： 无

返回值： 0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例:

ZM_GetInfo(void)

ZM_DRAWBARCODE

说明：

ZM_DrawBarcode 函数作用是打印一个条码。

原型：

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

参数：

px：设置 X 坐标，以点(dots)为单位.

py：设置 Y 坐标，以点(dots)为单位.

pdirec:选择条码的打印方向. 0—不旋转;1—旋转 90°；2—旋转 180°；3—旋转 270°.

pCode：选择要打印的条码类型.

P4 值	条码类型
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：设置条码中窄单元的宽度,以点(dots)为单位.

pHorizontal：设置条码中宽单元的宽度,以点(dots)为单位.

pVertical：设置条码高度,以点(dots)为单位.

ptext：选’N’则不打印条码下面的人可识别文字,

选’B’则打印条码下面的人可识别文字.

pstr：一个长度为 1-100 的字符串。用户可以用”DATA”,Cn,Vn 自由排列组合成一个组合字符串,

“DATA”：常量字符串，必须用‘”’作为起始和结束符号，如“ZMIN Printer”。

Cn：序列号数值，此序列号必须已经定义，请参考 ZM_DrawText 函数范例。

Vn：变量字符串，此变量字符串必须已经定义，请参考 ZM_DrawText 函数范例。

如：“data1”CnVn“data2”.

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例:

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

ZM_DrawBarcode (50,30,0,'1A',1,1,10,'N',C2);

ZM_DrawBarcode (50,30,0,'1A',1,1,10,'N',V1);

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

ZM_DRAWBARCODEEX

说明：

ZM_DrawBarcodeEx 函数作用是打印一个条码，内容可以是常量、序列号、变量或组合字符串。

原型：

int ZM_DrawBarcodeEx (unsigned int px, unsigned int py,

unsigned int pdirec, LPTSTR pCode,

nsigned int NarrowWidth, unsigned int pHorizontal,

unsigned int pVertical, char ptext, LPTSTR pstr ,BOOL Varible);

参数:

px: 设置 X 坐标,以点(dots)为单位.

py: 设置 Y 坐标,以点(dots)为单位.

pdirec:选择条码的打印方向. 0—不旋转;1—旋转 90°; 2—旋转 180°; 3—旋转 270°.

pCode: 选择要打印的条码类型.

P4 值	条码类型
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：设置条码中窄单元的宽度,以点(dots)为单位.

pHorizontal：设置条码中宽单元的宽度,以点(dots)为单位.

pVertical：设置条码高度,以点(dots)为单位.

ptext：选’N’则不打印条码下面的人可识别文字,

选’B’则打印条码下面的人可识别文字.

pstr：一个长度为 1-100 的字符串。用户可以用”DATA”,Cn,Vn 自由排列组合成一个组合字符串,

“DATA”：常量字符串，必须用‘”’作为起始和结束符号，如“ZMIN Printer”。

Cn：序列号数值，此序列号必须已经定义，请参考 ZM_DrawText 函数范例。

Vn：变量字符串，此变量字符串必须已经定义，请参考 ZM_DrawText 函数范例。

如：“data1”CnVn“data2”.

BOOL Varible：

TRUE 表示当前字符串当中包含有变量,,需加‘\”’将打印数据包含。

例如：ZM_DrawBarcodeEx (50,30,0,’1A’,1,1,10,’N’,”\”123456\””,TRUE);

FALSE 表示当前字符串当中不包含有变量操作，不需加‘\”’.

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

返回值： 0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例:

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

说明：

ZM_DrawBar2D_DATAMATRIX **函数作用是打印一个 DataMatrix 二维条码。**

原型：

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

*参数：

int	x;	●X 座标。
int	y;	●Y 座标。备注：1 dot = 0.125 mm。
int	w;	●最大列印宽度，单位 dots。
int	v;	●最大列印高度，单位 dots。
int	o;	●设置旋转方向, 范围：0～3。
int	m;	●设置放大倍数，以点(dots)为单位,范围值：(1 - 9).
LPCTSTR	pstr;	●资料字串。

传回值： 0 -> OK.

Reference Error.txt file.

ZM_DRAWBAR2D_QR

功能：打印一个 QR 条码

```
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 )
```

*参数：

	int	x;	●X 座标。
	int	y;	●Y 座标。备注：1 dot = 0.125 mm。
	int	w;	●最大列印宽度，单位 dots。
	int	v;	●最大列印高度，单位 dots。
	int	o;	●设置旋转方向，范围：0～3。 （0--0°，1--90°，2--180°，3--270°）
	int	r;	●设置放大倍数，以点(dots)为单位,范围值：(1 - 9)。 （1--放大 1 倍， 2--放大 2 倍，3--放大 3 倍…… ）
	int	m;	●QR 码编码模式选择,范围值(0 - 4)。 0 是选择数字模式 1 是选择数字字母模式 2 是选择字节模式 0~256 3 是选择中国汉字模式 4 是选择混合模式
	int	g;	●QR 码纠错等级选择,范围值(0 - 3)。 0 是'L'等级 1 是'M'等级

2 是'Q1'等级

3 是'H1'等级

int s; ●QR 码掩模图形选择,范围值(0 - 8)。

0 - 是掩模图形 000

1 - 是掩模图形 001

2 - 是掩模图形 010

3 - 是掩模图形 011

4 - 是掩模图形 100

5 - 是掩模图形 101

6 - 是掩模图形 110

7 - 是掩模图形 111

8 - 是自动选择掩模图形

LPCTSTR pstr; ●资料字串。

传回值： 0 -> OK.

Reference Error.txt file.

ZM_DRAWBAR2D_MAXICODE

说明：

ZM_DrawBar2D_MaxiCode 函数作用是打印一个 MaxiCode 条码.

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

*参数：

	int	x;	●X 座标。
	int	y;	●Y 座标。备注：1 dot = 0.125 mm。
	int	m;	●Mode [2 - 4];
	int	u;	●是否是 UPS 格式
	LPCTSTR pstr;		●资料字串。

传回值： 0 -> OK.

Reference Error.txt file.

ZM_DRAWBAR2D_PDF417

说明：

ZM_DrawBar2D_Pdf417 函数作用是打印一个 PDF417 二维条码。

原型：

```
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)
```

参数：

unsigned int	x;	X 座标。
unsigned int	y;	Y 座标。备注：1 dot = 0.125 mm。
unsigned int	w;	最大列印宽度，单位 dots。
unsigned int	v;	最大列印高度，单位 dots。
unsigned int	s;	错误校正等级，范围：0～8。
unsigned int	c;	资料压缩等级，范围：0 或 1。
unsigned int	px;	模组宽度，范围：2～9 dots。
unsigned int	py;	模组高度，范围：4～99 dots。
unsigned int	r;	最大 row count。
unsigned int	l;	最大 column count。
unsigned int	t;	Truncation flag， ’0’ 是 normal 和 ’1’ 是 truncated.
unsigned int	o;	列印方向定位，’0’是 0°，’1’是 90°、

’2’是 180°，’3’是 270°

LPCTSTR pstr; 资料字串。

传回值： 0 -> OK.

Reference Error.txt file.

范例： 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_DRAWBAR2D_HANXIN

说明：

ZM_DrawBar2D_HANXIN **函数作用是打印一个汉信码二维条码。**

```
ZM_DrawBar2D_HANXIN( 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 )
```

参数： unsigned int x; ●X 座标。

unsigned int y; ●Y 座标。

unsigned int w: 最大打印宽度, 以点(dots)为单位.

unsigned int v: 最大打印高度,以点(dots)为单位.

unsigned int o: 设置旋转方向. 范围值(0 到 3)

(0-0°,1--90°,2--180°, 3--270°)

unsigned int r : 设置放大倍数，以点(dots)为单位.范围值：(0 - 30)

(0-放大 1 倍， 1-放大 2 倍 2-放大 3 倍……依此类推)。

unsigned int m: 汉信码编码模式选择. 范围值(0 到 6)

0 是选择数字模式，

1 是选择 TEXT 模式，

2 是选择二进制模式，

3 是选择常用汉字 1 区模式编码

4 是选择常用汉字 2 区模式编码

5 是 GB 18030 双字节区模式

6 是 GB 18030 四字节模式编码

unsigned int g: 汉信码纠错等级选择. 范围值(0 到 3)

0 是'L1'等级

1 是'L2'等级

2 是'L3'等级

3 是'L4'等级

unsigned int s: 汉信码掩模图形选择.范围值(0 到 3)

0 是掩模图形 00

1 是掩模图形 01

2 是掩模图形 10

3 是掩模图形 11

LPCTSTR pstr; ●资料字串。

传回值： 0 -> OK.

Reference Error.txt file.

ZM_DRAWTEXTTRUETYPEW

注意：

必须先正确安装 ZMIN 打印机驱动程序，才能使用这个 ZM_DrawTextTrueTypeW 函数。

说明：

ZM_DrawTextTrueTypeW 作用是打印一行 TrueType Font 文字，并且文字宽度和高度可以微调。

原型：

```
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 )
```

参数：

x：设置 X 坐标，以点(dots)为单位；

y：设置 X 坐标，以点(dots)为单位；

FHeight：字型高度，以点(dots)为单位；

FWidth：字型宽度，以点(dots)为单位；

＊ 如果想打印正常比例的字体，需将 FWidth 设置为 0；

FType：字型名称；

Fspin：字体旋转角度及文字相对坐标对齐方式；

1 -> 居左 0 度，2 -> 居左 90 度，3 ->居左 180 度，4 ->居左 270 度，5 ->居中 0 度，6 ->居中 90 度，

7 ->居中 270 度，8 ->居中 270 度

Fweight：字体粗细。

0 and 400 -> 400 标准、

100 -> 非常细、200 -> 极细、

300 -> 细　　、500 -> 中等、

600 -> 半粗　、700 -> 粗　、

800 -> 特粗　、900 -> 黑体。

Fitalic：斜体，0 -> FALSE、1 -> TRUE；

Funline：文字加底线，0 -> FALSE、1 -> TRUE；

FstrikeOut：文字加删除线，0 -> FALSE、1 -> TRUE；

id_name：识别名称，因为一行 TrueType 文字将被转换成 PCX 格式数据以 id_name 作为 PCX 格式图形的名称存放到打印机内，在关机前都可以多次通过 ZM_DrawPcxGraphics()调用 id_name 打印这行文字；

(当 data 参数或其他参数不同时，请务必设定不同的 id_name 值)

data：字符串内容。

返回值：　0　　-> OK；

其它返回值请参考章节：　**ZMWIN.dll 错误返回值解析**。

范例：打印 3mm 高度的汉字：

203DPI 打印机需将 FHeight 设置为 3 / 0.125 = 24 个点；

300DPI 打印机需将 FHeight 设置为 3 / 0.08 = 38 个点(四舍五入).

```
ZM_DrawTextTrueTypeW (30,35,24,0,"宋体",4,400,0,0,0,"A1","机要绝密");
```

ZM_DRAWTEXT

说明：

ZM_DrawText 函数作用是打印一行文本文字。

原型：

```
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 );
```

参数：

px：设置 X 坐标，以点(dots)为单位.

py：设置 Y 坐标，以点(dots)为单位.

pdirec：选择文字的打印方向. 0—不旋转;1—旋转 90°；2—旋转 180°；3—旋转 270°.

pFont：选择内置字体或软字体. 1—5：为打印机内置字体；‘A’—‘Z’：为下载的软字体.

a 为打印机内置 24*24 简体汉字.

取值	描述
1	西文字体1
2	西文字体2
3	西文字体3
4	西文字体4
5	西文字体5
‘a’	24点阵中文宋体

‘A’~’Z’	软字体

pHorizontal：设置点阵水平放大系数. 可选择:1—24.

pVertical：设置点阵垂直放大系数. 可选择:1—24.

ptext：选’N’则打印正常文本(如黑字白底文本),

选’R’则打印文本反色文本(如白字黑底文本).

pstr：一个长度为 1-100 的字符串。用户可以用”DATA”,Cn,Vn 自由排列组合成一个组合字符串,

“DATA”：常量字符串，必须用‘”’作为起始和结束符号，如“ZMIN Printer”。

Cn：序列号数值，此序列号必须已经定义，请参考范例。

Vn：变量字符串，此变量字符串必须已经定义，请参考范例。

如:“data1”CnVn“data2”.

返回值： 0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例:

```
ZM_DrawText (50,30,0,2,1,1,’N’,”123456789”);
```

打印序列号和变量字符串,（一般在 Form 中使用）

```
ZM_FormDel("TEST");                          //删除 Form "TEST"
ZM_FormDownload("TEST");                     //下载 Form "TEST"
ZM_DefineCounter(0, 6, 'N', "+1", "Enter Start");//定义序列号数值 C0，命名为：'C'+第一个参数
ZM_DefineVariable(0,16,'N',"Variable");      //定义变量字符串 V0，命名为：'V'+第一个参数
ZM_DrawText (50,30,0,5,1,1,'N',"V0");        //打印变量字符串 V0
```

```
ZM_DrawText (100,70,0,5,1,1,'N',"C0");          //打印序列号数值 C0
ZM_FormEnd();


ZM_ClearBuffer();
ZM_ExecForm("TEST");                            //执行 Form
ZM_Download();                                   //下载数据
ZM_DownloadInitVar("123");                       //为 C0 赋值
ZM_DownloadInitVar("456");                       //为 V0 赋值
ZM_PrintLabel(2, 1);
```

ZM_DRAWTEXTEX

说明：

ZM_DrawTextEx 函数作用是打印一行文本文字，内容可以是常量、序列号、变量或组合字符串。

原型：

```
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 Varible);
```

参数：

px：设置 X 坐标，以点(dots)为单位.

py：设置 Y 坐标，以点(dots)为单位.

pdirec：选择文字的打印方向. 0—不旋转;1—旋转 90°；2—旋转 180°；3—旋转 270°.

pFont：选择内置字体或软字体. 1—5：为打印机内置字体；‘A’—‘Z’：为下载的软字体.

a 为打印机内置 24*24 简体汉字.

取值	描述
1	西文字体1
2	西文字体2
3	西文字体3
4	西文字体4
5	西文字体5

‘a’	24点阵中文宋体
‘A’~’Z’	软字体

pHorizontal：设置点阵水平放大系数. 可选择:1—24.

pVertical：设置点阵垂直放大系数. 可选择:1—24.

ptext：选’N’则打印正常文本(如黑字白底文本),

选’R’则打印文本反色文本(如白字黑底文本).

pstr：一个长度为 1-100 的字符串。用户可以用”DATA”,Cn,Vn 自由排列组合成一个组合字符串,

“DATA”：常量字符串，必须用‘”’作为起始和结束符号，如“ZMIN Printer”。

Cn：序列号数值，此序列号必须已经定义，请参考 ZM_DrawText 函数范例。

Vn：变量字符串，此变量字符串必须已经定义，请参考 ZM_DrawText 函数范例。

如:“data1”CnVn“data2”.

BOOL Varible：

TRUE 表示当前字符串当中包含有变量操作,需加‘\”’将打印数据包含。

例如：ZM_DrawTextEx (50,30,0,2,1,1,’N’,”\”123456789\””,TRUE);

FALSE 表示当前字符串当中不包含有变量操作，不需加‘\”’.

例如：ZM_DrawTextEx (50,30,0,2,1,1,’N’,” 123456789”,FALSE);

返回值： 0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例:

```
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);

ZM_DEFINECOUNTER

说明：

ZM_DefineCounter 函数作用是定义一个序列号变量。

原型：

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

参数：

id： 系列号 ID，取值范围：0—9；

maxNum：序列号最大数字位数；取值范围：1—40；

ptext：对齐方式；L—左对齐，R—右对齐，C—居中，N—不对齐；

Pstr：序列号的变化规律；由”+”或”-”加上一个数字，再加上一个变化标志（D －十进制，B －二进制，O － 八进制，H －十六进制，X-自定义模式，允许用户设置最多64个字符）组成：

“+1”=每次增加 1，默认按照十进制计算：如 1234，1235，1236，…；

“+3D”=每次增加 3，按照十进制计算，同上；

“-1B”=每次减少 1，按照二进制计算：如 1111，1110，1101，…；

“-4O”=每次减少 4，按照八进制计算：如 1234，1230，1224，…；

“-6H”=每次减少 6，按照十六进制计算：如 1234，122E，1228，…；

“+3X”=如变化规律表内容为：TE2DOKLUO46MNY37，起始值是”T062”，

则 T062，T06K，T060，…；

pMsg：提示信息字符串；可在打印机 LCD 上或可编程键盘(KDU)的显示屏上显示。

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例:

```
ZM_DefineCounter (0, 6, ’ N’ ,” +1”, ″\” Enter\” Code:″);
```

ZM_SETDARKNESS

说明：

ZM_SetDarkness 函数的作用是设置打印头发热温度。

原型：

int ZM_SetDarkness (unsigned int id);

参数：

id：取值范围：0—20，缺省为 10；

此值并不是真正意义的温度数值,而是相对数值,0 表示打印头工作在最低发热状态,20 表示打印工作在最高发热状态.

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_SetDarkness (10);

ZM_SETPRINTSPEED

说明：

ZM_SetPrintSpeed 函数的作用是设置打印速度。

原型：

int ZM_SetPrintSpeed (unsigned int px);

参数：

px：取值范围为0 – 6，或者10 – 80。

p1值	速度	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)	

返回值：

0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：ZM_SetPrintSpeed (40);

ZM_SETLABELHEIGHT

说明：

ZM_SetLabelHeight 函数的作用是设置标签的高度和定位间隙\黑线\穿孔的高度。

原型：

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

参数：

lheight：标签的高度，以点(dots)为单位，取值范围：0-65535；

gapH：标签间的定位间隙/黑线/穿孔的高度，以点(dots)为单位，

取值范围：0-65535；

gapH 的取值与标签定位方式相关：

间隙模式(GAP MODE):缺省模式,gapH 设置为间隙的高度，

穿孔定位属于间隙模式的特例；

黑线模式(BLACK LINE MODE)：gapH 设置为黑线的高度；

连续纸模式(CONTINUOUS MODE):gapH 设置为 0.这时候,纸张探测器只检测纸张是否用尽。

返回值：0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_SetLabelHeight (160, 24);

ZM_SETLABELWIDTH

说明：

ZM_SetLabelWidth 函数的作用是设置标签的宽度。

原型：

int ZM_SetLabelWidth (unsigned int lwidth);

参数：

lwidth：标签的宽度，以点(dots)为单位。

返回值：0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

```
ZM_SetLabelWidth (250);
```

ZM_DRAWPCXGRAPHICS

说明：

ZM_DrawPcxGraphics 函数作用是打印指定的图形。

注：被打印的图形必须预先使用 ZM_PcxGraphicsDownload（）存储到打印机里。

原型：

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

参数：

px：设置 X 坐标;以点(dots)为单位；

py：设置 Y 坐标;以点(dots)为单位；

game：即将打印的图形名称，最大长度为 16 个字符，必须是在 ZM_PcxGraphicsDownload（）中自定义的图形名称。

返回值： 0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_DrawPcxGraphics (100, 50, “PCX1”);

ZM_PCXGRAPHICSLIST

说明：

ZM_PcxGraphicsList 函数的作用是打印已存储在打印机 RAM 或 FLASH 存储器里的图形名称清单。

原型：

int ZM_PcxGraphicsList (void);

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

```
ZM_PcxGraphicsList ( );
```

ZM_PCXGRAPHICSDEL

说明：

ZM_PcxGraphicsDel 函数作用是删除存储在打印机里的一个或所有图形。

原型：

int ZM_PcxGraphicsDel (LPTSTR pid);

参数：

pid：即将删除的图形名称，最大长度为 16 个字符；

如果 pid = “*”,则将删除所有存储在 RAM 或 FLASH 存储器里的图形。

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_PcxGraphics Del (“PCX2”);

ZM_PCXGRAPHICSDOWNLOAD

说明：

ZM_PcxGraphicsDownload 函数的作用是存储一个 PCX 格式的图形到打印机。

原型：

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

参数：

pcxname：自定义图形的名称，最大长度为 16 个字符；当图形存储到打印机后，用户在 ZM_DrawPcxGraphics（）中使用此名称才能将图形读取出来打印。

pcxpath: PCX 图形文件在 PC 机存储器里的路径;

返回值：　0　　-> OK;

其它返回值请参考章节：　**ZMWIN.dll 错误返回值解析**。

范例：

ZM_PcxGraphicsDownload（“PCXA”，“c:\\test1111.pcx”);

ZM_BMPGRAPHICSDOWNLOAD

说明：

ZM_BmpGraphicsDownload 函数的作用是先转换 BMP 到 PCX 格式，然手将 PCX 格式的图形到打印机。

原型：

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

参数：

pcxname：自定义图形的名称，最大长度为 16 个字符；当图形存储到打印机后，用户在 ZM_DrawPcxGraphics（）中使用此名称才能将图形读取出来打印。

pcxpath:　BMP 图形文件在 PC 机存储器里的路径;

iDire　：旋转方向 0->0° 1->90° 2->180° 3->270° 其他->0°

返回值：　0　　-> OK;

其它返回值请参考章节：　**ZMWIN.dll 错误返回值解析**。

范例：

ZM_BmpGraphicsDownload（“PCXA”，“c:\\test1111.bmp”，0);

ZM_PRINTPCX

说明：

ZM_PrintPCX 函数是打印一个 PCX 格式的图形。

这个函数将 ZM_PcxGraphicsDownload（）和 ZM_DrawPcxGraphics（）组合封装到一起使用。

原型：

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

参数：

px：设置 X 坐标;以点(dots)为单位；

py：设置 Y 坐标;以点(dots)为单位；

filename：PCX 图形文件名称，可包含文件路径。

格式如：“XXXXXXXX.XXX” 或 “X:\\XXX\\XXX.PCX”。

返回值：0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

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

ZM_DRAWBINGRAPHICS

说明：

ZM_DrawBinGraphics 函数的作用是打印二进制格式的图形。

二进制格式图形是不压缩的图形数据;每一个比特(bit)表示一个点;比特值为 0 时此点将打印,为 1 时此点不打印。

原型：

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

参数:

px：设置 X 坐标，以点(dots)为单位；

py：设置 Y 坐标，以点(dots)为单位；

pbyte：一行数据的字节数(1Byte = 8bits)；如果一行数据的点数不能整除 8，则其字节数应该等于商加上 1；如：一行是 14bit 的数据的字节数是 2；

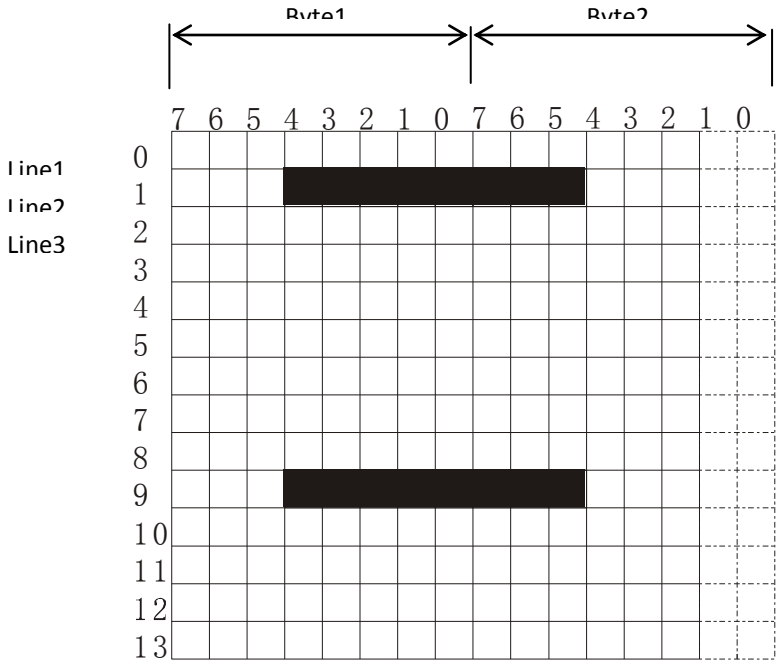
Ph：图形的高度，以点(dots)为单位；

Gdata([…raster data…])：二进制图形数据,数据量大小= pbyte * pH (Bytes)。**Bit 值为 0 是打印内容，为 1 是空白内容。**

二进制数据传输顺序是从左到右，从上到下，以下图为例：

数据传输顺序为：Line1 的 Byte1(0xff)，Line1 的 Byte2(0xff)，Line2 的 Byte1(0xe0),Line2 的 Byte2(0x1f)，Line3 的 Byte1(0xff)，Line3 的 Byte2(0xff)，…

其中虚线部分是非图形区域，对应它们的 bit 值为 1。



返回值：0　　-> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

```
char buf[] = {0xff, 0xff, 0xe0, 0x1f, 0xff, 0xff…};
ZM_DrawBinGraphics (20, 30, 4, 14, buf);
```

ZM_DRAWLINEXOR

说明：

ZM_DrawLineXor 函数作用是画直线(两直线相交处作"异或"处理)。

原型：

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

参数：

px：X 坐标，以点(dots)为单位；

py：Y 坐标，以点(dots)为单位；

pbyte：设置直线的水平长度，以点(dots)为单位；

pH：设置直线的垂直高度，以点(dots)为单位。

返回值：0　　-> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

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

ZM_DRAWLINEOR

说明：

ZM_DrawLineOr 函数作用是画直线(两直线相交处作"或"处理)。

原型：

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

参数:

px: 设置 X 坐标，以点(dots)为单位；

py: 设置 Y 坐标，以点(dots)为单位；

plength：设置直线的水平长度，以点(dots)为单位；

pH：设置直线的垂直高度，以点(dots)为单位。

返回值：0　　-> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

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

ZM_DRAWDIAGONAL

说明：

ZM_DrawDiagonal 函数的作用是画斜线。

原型：

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

参数：

px：设置斜线起始 X 坐标，以点(dots)为单位；

py：设置斜线起始 Y 坐标，以点(dots)为单位；

thickness：设置斜线粗细，以点(dots)为单位；

pEx：设置斜线终止 X 坐标，以点(dots)为单位；

pEy：设置斜线终止 Y 坐标，以点(dots)为单位。

返回值：0　　-> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

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

ZM_DRAWWHITELINE

说明：

ZM_DrawWhiteLine函数的作用是画白色直线。

原型：

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

参数：

px：设置 X 坐标，以点(dots)为单位；

py：设置 Y 坐标，以点(dots)为单位；

plength：设置直线的水平长度，以点(dots)为单位；

pH：设置直线的垂直高度，以点(dots)为单位。

返回值：0　　-> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

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

ZM_SETPRINTERSTATE

说明：

ZM_SetPrinterState函数的作用是设置打印机的工作状态。

原型：

int ZM_SetPrinterState (char state);

参数:

state 为以下几种字符：

D：设置打印机为热感印(热传导)状态；

P：设置打印机为连续送纸状态(缺省)；

L：设置打印机为打印一张标签后，暂停等待用户确定再打印下一张标签；

(确定方式：1.按”FEED”键；2.在安装剥纸器情况下，当用户取走标签后自动打印下一张标签)

C：设置打印机为安装切纸刀状态；

N：设置打印机为安装剥纸器状态。

注意:

1. 切纸刀与剥纸器不能同时安装；
2. 如果打印机状态设置不正确时，打印机前面板的 READY 指示灯将闪烁，请参考打印机用户手册的故障排除章节。

返回值：0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_SetPrinterState (‘D’);

ZM_PRINTLABEL

说明：

ZM_PrintLabel 函数的作用是命令打印机执行打印工作。

注：此函数不能在 FORM 的编排过程中，而用 ZM_PrintLabelAuto（）函数代替。

原型：

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

参数：

number：打印标签的数量，取值范围：1—65535；

cpnumber：每张标签的复制份数，取值范围：1—65535；

如果 cpnumber 没有设置，那么默认为 1。

返回值：0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_PrintLabel (2,3);

ZM_SETCOORDINATEORIGIN

说明：

ZM_SetCoordinateOrigin 函数的作用是设置/改变坐标原点。通过改变坐标原点可以实现一行多列的打印。

原型：

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

参数：

px：X 坐标移动的距离，确以点(dots)为单位；

Py：Y 坐标移动的距离，确以点(dots)为单位。

返回值：

0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_SetCoordinateOrigin (12,23);

ZM_PRINTCONFIGUNATION

说明：

ZM_PrintConfigunation 函数的作用是打印机器当前的设置/工作状态。

原型：

int ZM_PrintConfigunation ( );

参数：无

返回值：

0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_PrintConfigunation ( );

ZM_SOFTFONTLIST

说明：

ZM_SoftFontList 函数的作用是打印存储在 RAM 或 FLASH 存储器里的软字体的名称清单。

原型：

int ZM_SoftFontList (void);

参数：无

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_SoftFontList ();

ZM_SOFTFONTDEL

说明：

ZM_SoftFontDel 函数作用是删除存储在 RAM 或 FLASH 存储器里的一个或所有的软字体。

原型：

int ZM_SoftFontDel (char pid);

参数:

pid: 软字体 ID，取值范围：A—Z 或 * ；

如果 pid = ‘*’,打印机将删除存储在 RAM 或 FLASH 存储器里所有的软字体.

返回值： 0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_SoftFontDel (‘A’);

ZM_FORMEND

说明：

ZM_FormEnd 函数:作用是结束存储表格(Form)，此函数与 ZM_FormDownload 配对使用。

原型：

int ZM_FormEnd (void);

返回值： 0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

```
ZM_FormDownload（“Form1”）;
…
ZM_FormEnd ( );
```

ZM_FORMLIST

说明：

ZM_FormList 函数作用是打印存储在打印机里的表格名称清单。

原型：

int ZM_FormList (void);

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

```
ZM_FormList ( );
```

ZM_FORMDEL

说明：

ZM_FormDel 函数是删除存储在打印机里的一个或所有的表格。

原型：

int ZM_FormDel (LPTSTR pid);

参数：

pid：即将删除的软字体的名称，最大长度为 16 个字符；

如果 pid = “*”,打印机将删除存储在 RAM 或 FLASH 存储器里所有的表格。

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_FormDel (“FORMNAME”);

ZM_EXECFORM

说明：

ZM_ExecForm 函数的作用是运行指定的表格。

原型：

int ZM_ExecForm (LPTSTR pid);

参数：

pid：即将运行的表格的名称，最大长度为 16 个字符。

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_ExecForm (“FORM1”);

ZM_FORMDOWNLOAD

说明：

ZM_FormDownload 函数的作用是存储一个表格到打印机；此命令与 ZM_FormEnd 函数配对使用；

如果在 EnableFlASH（）函数后使用,表格的内容则存储到 FLASH 存储器；

如果在默认状态下或在 DisableFlASH（）函数后使用，表格的内容则存储到 RAM 存储器。

原型：

int ZM_FormDownload (LPTSTR pid);

参数：

pid：自定义的表格名称，最大长度为 16 个字符；此表格内容存储到打印机后，用户必须使用才能运行它。

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_FormDownload (“FORMNAME”);

ZM_PRINTLABELAUTO

说明：

ZM_PrintLabelAuto 函数的作用是自动执行打印工作。

当 FORM 中存在变量或者序列号时，建议使用此函数，当用户输入全部的变量内容,打印机将立刻开始打印标签.

注：只能在 FORM 里使用。

原型：

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

参数：

number：打印标签的数量，取值范围：1—65535；

cpnumber：每张标签的复制份数，取值范围：1—65535；

如果 cpnumber 没有设置，那么默认为 1。

返回值：0　　-> OK；

其它返回值请参考章节：　**ZMWIN.dll 错误返回值解析**。

范例：

ZM_PrintLabelAuto (2,3);

原型：

int ZM_ EnableErrorReport (void);

参数：无

返回值：

0　　-> OK；

其它返回值请参考章节：　**ZMWIN.dll 错误返回值解析**。

范例：

ZM_ EnableErrorReport ( );

ZM_DEFINEVARIABLE

说明：

ZM_DefineVariable 函数的作用是定义变量。

在 FORM 里使用此函数来定义一个变量。

原型：

int ZM_DefineVariable (unsigned int pid, unsigned int pmax,

char porder, LPTSTR pmsg);

参数：

pid：变量 ID 号码，取值范围：00—99；

pmax：最大字符个数，取值范围：1—99；

如果使用 KDU，只能在 16 以内；

porder：对齐方式，L—左对齐，R—右对齐，C—居中，N—不对齐；

pmsg：提示内容，将会在 KDU 或打印机的 LCD 显示。

返回值：

0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

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

ZM_DRAWRECTANGLE

说明：

ZM_DrawRectangle 函数的作用是画距形。

原型：

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

参数：

px：起始点的 X 坐标，以点(dots)为单位；

py：起始点的 Y 坐标，以点(dots)为单位；

thickness：边框的粗细，以点(dots)为单位；

pEx：终止点的 X 坐标，以点(dots)为单位；

pEy：终止点的 Y 坐标，以点(dots)为单位。

返回值：

0　　-> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

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

ZM_SETDIRECTION

说明：

ZM_SetDirection 函数的作用是设置标签打印方向。

注：此函数将改变整张标签上所有内容的方向，如文本、条码、直线、距形。

原型：

int ZM_SetDirection (char direct);

参数：

direct：方向，取值为 B 或 T，缺省值为 T。

B：将从标签右下角开始打印；

T：从标签左上角开始正常打印。

返回值：

0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_SetDirection (‘B’);

ZM_ENABLEFLASH

说明：

ZM_EnableFlASH 函数的作用是选择 FLASH 存储器。

当使用此函数后，发送到打印机的数据将被存储到 FLASH 里。

原型：

int ZM_EnableFlASH (void);

参数：无

返回值：

0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_EnableFlASH ( );

ZM_DISABLEFLASH

说明：

ZM_ DisableFLASH 函数的作用是取消选择 FLASH 存储器;

当使用此函数后，发送到打印机的数据将被存储到 SDRAM 里。

原型：

int ZM_ DisableFLASH (void);

参数：无

返回值：

0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_ DisableFLASH ( );

ZM_DOWNLOAD

说明：

ZM_Download 函数的作用是下载变量或系列号变量。

请参阅 PPL Ⅰ的“？”命令。

原型：

int ZM_Download(void);

参数：无

返回值：

0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

```
ZM_Download( );
```

ZM_DOWNLOADINITVAR

说明：

ZM_DownloadInitVar 函数的作用是初始化变量或系列号变量。

需跟在 ZM_Download()函数后使用。

原型：

int ZM_DownloadInitVar(LPTSTR pstr)；

参数：无

返回值：

0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

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

ZM_RESET

说明：

ZM_Reset 函数的作用是将打印机复位。

这个命令对打印机复位，会将打印机设置恢复到出厂状态

不能在打印函数序列的开头或者中间使用此指令，否则会将该指令后面的内容清空，导致打印机不执行该函数后面的打印函数。

原型：

int ZM_Reset(void);

参数：无

返回值：

0 -> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_Reset( );

ZM_FEEDMEDIA

说明：

ZM_ FeedMedia **函数的作用是命令打印机走一行标签。**

原型：

int ZM_FeedMedia (void);

参数：无

返回值：

0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_ FeedMedia ( );

ZM_MEDIADETECT

说明：

ZM_ MediaDetect **函数的作用校准纸张探测器。**

原型：

int ZM_MediaDetect (void);

参数：无

返回值：

0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_ MediaDetect ( );

ZM_CUTPAGE

说明：

ZM_CutPage 函数的作用是设置切刀的工作周期.（即每打印多少页标签后，切刀才切一次纸）。

原型：

int ZM_CutPage (UINT page);

参数：

page：页数,取值范围：1-999；默认是 1。

返回值：

0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_CutPage(1);

ZM_BINGRAPHICSLIST

说明：

ZM_BinGraphicsList 函数的作用是打印已存储在打印机 RAM 或 FLASH 存储器里的图形名称清单，**包含 Bin 格式和 PCX 格式的图形名称。**

原型：

int ZM_BinGraphicsList (void);

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

```
ZM_BinGraphicsList ( );
```

ZM_BINGRAPHICSDEL

说明：

ZM_BinGraphicsDel 函数作用是删除存储在打印机里的一个或所有图形，此图形可以是 Bin 格式或 PCX 格式的。

原型：

int ZM_BinGraphicsDel (LPTSTR pid);

参数：

pid：即将删除的图形名称，最大长度为 16 个字符；

如果 pid = “*”，则将删除所有存储在 RAM 或 FLASH 存储器里的图形。

返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_BinGraphics Del (“Bin2”);

ZM_BINGRAPHICSDOWNLOAD

说明：

ZM_BinGraphicsDownload 函数的作用是存储一个 Bin 格式的图形到打印机。

原型：

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

参数：

name：自定义图形的名称，最大长度为 16 个字符；当图形存储到打印机后，用户在 ZM_RecallBinGraphics（）中使用此名称才能将图形读取出来打印。

pbyte：一行数据的字节数(1Byte = 8bits)；如果一行数据的点数不能整除 8，则其字节数应该等于商加上 1；如：一行是 14bit 的数据的字节数是 2；

pH：图形的高度，以点(dots)为单位；

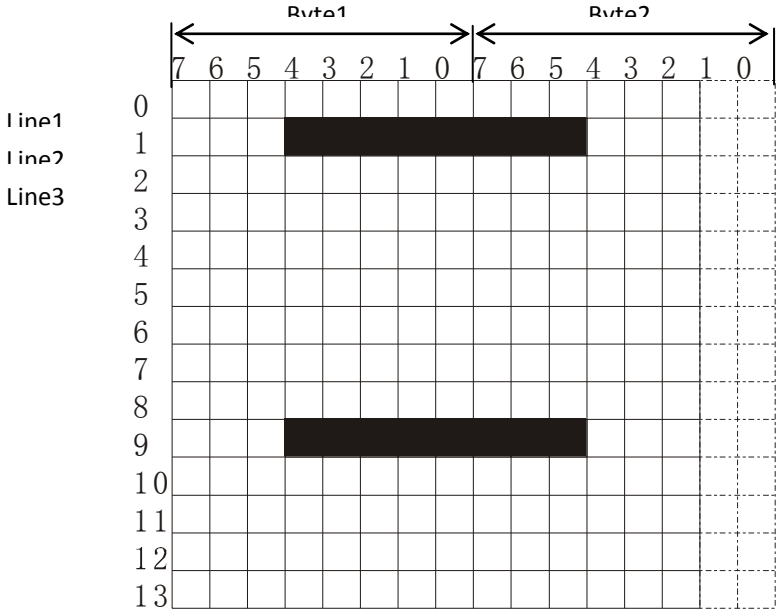
Gdata([…raster data…])：二进制图形数据,数据量大小= pbyte * pH (Bytes)。

Bit 值为 1 是打印内容，为 0 是空白内容。

二进制数据传输顺序是从左到右，从上到下，以下图为例：

数据传输顺序为：Line1 的 Byte1(0x00)，Line1 的 Byte2(0x00)，Line2 的 Byte1(0x1f),Line2 的 Byte2(0xe0)，Line3 的 Byte1(0x00)，Line3 的 Byte2(0x00)，…

其中虚线部分是非图形区域，对应它们的 bit 值为 0。



返回值： 0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

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

ZM_RECALLBINGRAPHICS

说明：

ZM_RecallBinGraphics 函数是打印一个 Bin 格式的图形。

原型：

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

参数：

px：设置 X 坐标;以点(dots)为单位；

py：设置 Y 坐标;以点(dots)为单位；

name：Bin 图形文件名称；

返回值：0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

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

ZM_DISABLEERRORREPORT

说明：

ZM_DisableErrorReport 函数的作用是取消错误反馈。

原型：

int ZM_DisableErrorReport(void);

参数：无

返回值：

0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_DisableErrorReport( );

ZM_ENABLEERRORREPORT

说明：

ZM_ EnableErrorReport函数的作用是设置错误反馈 。

打印机的反馈数据从 RS232 串口返回电脑.

如果打印中发生错误,打印机将先发送一个 NACK(15H)字符回电脑,跟着发送出错编号.

如果没有错误发生,打印机将在接收到 P 命令后发送 ACK(06H)字符.

错误代码	说明
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 to long
0x0c	PDF-417 coded data to large to fit in bar code
0x0d	
0x0e	

ZM_ERRORREPORT

说明：

ZM_ErrorReport 函数的作用是发送错误查询指令到打印机并且从指定串口接收和分析打印机当前错误代码。必须在 ZM_ClearBuffer()函数调用之后使用。

在执行此函数之前，如果程序正在打开一个发送端口，必须首先执行 ClosePort()关闭已打开的端口！

（建议使用 ZM_ErrorReportEx，该函数支持串口超时反馈）

用户可以使用此命令立刻确定打印机的当前错误状态，打印机将传回 4 个字节到主机：

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

表<一>

Error/Status code	解释	ZM_ErrorReport()返回值	ErrorCode
00	无错误	0	“00”
01	语法错误	1	“01”
82	碳带探测出错	82	“82”
83	标签探测出错	83	“83”
86	切刀检测出错	86	“86”
87	打印头未关闭	87	“87”
88	暂停状态	88	“88”
99	其它错误	99	“99”

原型：

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

参数：

wPort：发送数据的端口；(**此参数主要为编程扩展预留，默认 0 即可，无作用。**)

0： 表示打印到文件 PBuffi.txt（在执行程序目录下建立文件）；（此端口请勿使用！）

1： 表示打开LPT1；

2： 表示打开 LPT2；

3： 表示打开 LPT3；

4： 表示打开 COM1；

5： 表示打开 COM2；

6： 表示打开 COM3。

rPort：接收打印机当前错误状态代码的端口；（此参数可以支持 COM1 到 COM255）

1： 表示打开 COM1 作为接收端口；

2： 表示打开 COM2 作为接收端口；

3： 表示打开 COM3 作为接收端口；

BaudRate：要设置的串口波特率，可取值：

9600，19200，38400，57600；

HandShake：是否使用硬件握手（HandShaking）；

TRUE：硬件握手（HandShaking）有效，

FALSE：硬件握手（HandShaking）无效。

返回值：

>=0，请参考表<一>中的 Error/Status code 值

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

1. 通过设置 X1DPI 打印机驱动中的打印端口，从 COM1 读取当前错误状态代码：

```
OpenPort("ZMIN X1DPI ");
....
```

```
ZM_ClearBuffer();
char buff[5] = {0};
ZM_ ErrorReport (1,    1,38400, FALSE);
（或 ZM_ ErrorReport (1,    1, 38400, FALSE );）


…….
ClosePort();
```

ZM_ERRORREPORTEX

说明：

ZM_ErrorReportEx 函数的作用是发送错误查询指令到打印机并且从指定串口接收和分析打印机当前错误代码。必须在 ZM_ClearBuffer()函数调用之后使用。

在执行此函数之前，如果程序正在打开一个发送端口，必须首先执行 ClosePort()关闭已打开的端口！

用户可以使用此命令立刻确定打印机的当前错误状态，打印机将传回 4 个字节到主机：

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

表<一>

Error/Status code	解释	ZM_ErrorReport()返回值	ErrorCode
00	无错误	0	“00”
01	语法错误	1	“01”
82	碳带探测出错	82	“82”
83	标签探测出错	83	“83”
86	切刀检测出错	86	“86”
87	打印头未关闭	87	“87”
88	暂停状态	88	“88”
99	其它错误	99	“99”

原型：

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

参数：

wPort：发送数据的端口；(**此参数主要为编程扩展预留，默认 0 即可，无作用。**)

0： 表示打印到文件 PBuffi.txt（在执行程序目录下建立文件）；（此端口请勿使用！）

1： 表示打开LPT1；

2： 表示打开 LPT2；

3： 表示打开 LPT3；

4： 表示打开 COM1；

5： 表示打开 COM2；

6： 表示打开 COM3。

rPort：接收打印机当前错误状态代码的端口；（此参数可以支持 COM1 到 COM255）

1： 表示打开 COM1 作为接收端口；

2： 表示打开 COM2 作为接收端口；

3： 表示打开 COM3 作为接收端口；

BaudRate：要设置的串口波特率，可取值：

9600，19200，38400，57600；

HandShake：是否使用硬件握手（HandShaking）；

TRUE：硬件握手（HandShaking）有效，

FALSE：硬件握手（HandShaking）无效。

TimeOut： 接收串口超时等待时间；单位为：100ms

返回值：

>=0，请参考表<一>中的 Error/Status code 值

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

2. 通过设置 X1DPI 打印机驱动中的打印端口，从 COM1 读取当前错误状态代码：

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

```
.....

ClosePort();
```

ZM_SETPAGEPRINTCOUNT

说明：

ZM_SetPagePrintCount 函数的作用是设置某一打印页面的打印数量。

原型：

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

参数:

number：打印标签的数量，取值范围：1—65535；

cpnumber：每张标签的复制份数，取值范围：1—65535；

如果 cpnumber 没有设置，那么默认为 1。

返回值：0 -> OK；

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

ZM_SetPagePrintCount(1,1);

ZM_WRITEPRINTER

说明：

ZM_WritePrinter 函数的作用是送出数据到打印机。此函数和 ZM_PrintLabel 作用不一样，ZM_PrintLabel 相当于实现了 ZM_SetPagePrintCount 和 ZM_WritePrinter 两个功能。

原型：

int ZM_WritePrinter();

参数：无

返回值：0　　-> OK;

其它返回值请参考章节： **ZMWIN.dll 错误返回值解析**。

范例：

```
ZM_WritePrinter();
```

ZMWIN.DLL 错误返回值解析

-1000

to

-1030 ： 打开端口操作出错;

-1031

to

-1037 ： 串口读取操作出错;

-1040 ： 从串口读取数据出错;

-1041 ： 从串口读取数据出错;

-1042 ： 串口波特率值有误;

-2992 ： OpenPort 执行出错,请检查打印机是否安装正常!

-3001 ： ZM_GetInfo 执行出错;

-3002 ： ZM_DrawText 执行出错;

-3003 ： ZM_DrawText 的参数错误;

-3004 ： ZM_DrawText 或者 ZM_DrawBarcode 的 pdirec 参数错误;

-3005 ： ZM_DrawText 的 pFont 参数错误;

-3006 ： ZM_DrawText 的 pHorizontal 或者 pVertical 参数错误;

-3007 ： NULL;

-3008 ： ZM_DrawBarcode 执行出错;

-3009 ： ZM_DrawBarcode 参数错误;

-3010 ： ZM_DefineCounter 执行出错;

-3011 ： ZM_DefineCounter 参数错误;

-3012 ： ZM_SetDarkness 执行出错;

-3013 ： ZM_DS 执行出错;

-3014 ： ZM_SoftFontList 执行出错；

-3015 ： ZM_SoftFontDel 参数错误；

-3016 ： ZM_SoftFontDel 执行出错；

-3017 ： ZM_FormEnd 执行出错；

-3018 ： ZM_FormList 执行出错；

-3019 ： ZM_FormDel 分配内存出错；

-3020 ： ZM_FormDel 参数错误；

-3021 ： ZM_FormDel 执行出错；

-3022 ： ZM_ExecForm 分配内存出错；

-3023 ： ZM_ExecForm 参数错误；

-3024 ： ZM_ExecForm 执行出错；

-3025 ： ZM_FormDownload 分配内存出错；

-3026 ： ZM_FormDownload 参数错误；

-3027 ： ZM_FormDownload 执行出错；

-3028 ： ZM_DrawPcxGraphics 分配内存出错；

-3029 ： ZM_DrawPcxGraphics 参数错误；

-3030 ： ZM_DrawPcxGraphics 执行出错；

-3031 ： ZM_PcxGraphicsList 执行出错；

-3032 ： ZM_PcxGraphicsDel 分配内存出错；

-3033 ： ZM_PcxGraphicsDel 参数错误；

-3034 ： ZM_PcxGraphicsDel 执行出错；

-3035 ： ZM_PcxGraphicsDownload 内存分配错误；

-3036 ： ZM_PcxGraphicsDownload 打开文件错误；

-3037 ： ZM_PcxGraphicsDownload 执行出错；

-3038 ： ZM_DrawBinGraphics 执行出错；

-3039 ： (同上)；

-3040 ： NULL；

-3041 : ZM_DisableCircumgyrate 执行出错;

-3042 : ZM_EnableCircumgyrate 执行出错;

-3043 : ZM_DrawLineXor 执行出错;

-3044 : ZM_DrawLineOr 执行出错;

-3045 : ZM_DrawDiagonal 执行出错;

-3046 : ZM_DrawWhiteLine 执行出错;

-3047 : ZM_ClearBuffer 执行出错;

-3048 : ZM_SetPrinterState 执行出错;

-3049 : ZM_SetPrinterState 参数错误;

-3050 : ZM_PrintLabel 执行出错;

-3051 : ZM_PrintLabel 参数错误;

-3052 : ZM_PrintLabelAuto 执行出错;

-3053 : ZM_PrintLabelAuto 参数错误;

-3054 : ZM_SetLabelHeight 执行出错;

-3055 : ZM_SetLabelWidth 执行出错;

-3056 : ZM_SetCoordinateOrigin 执行出错;

-3057 : ZM_SetPrintSpeed 执行出错;

-3058 : ZM_SetPrintSpeed 参数错误;

-3059 : ZM_PrintConfigunation 执行出错;

-3060 : ZM_DisableErrorReport 执行出错;

-3061 : ZM_EnableErrorReport 执行出错;

-3062 : ZM_DefineVariable 执行出错;

-3063 : ZM_DefineVariable 参数错误;

-3064 : ZM_DefineVariable 参数错误;

-3065 : ZM_DrawRectangle 执行出错;

-3066 : ZM_Y 执行出错;

-3067 : ZM_Y 参数错误;

-3068 ： ZM_SetDirection 执行出错;

-3069 ： ZM_SetDirection 参数错误;

-3070 ： ZM_EnableFlASH 执行出错;

-3071 ： ZM_DisableFLASH 执行出错;

-3072 ： ZM_Download 执行出错;

-3073 ： ZM_Reset 执行出错;

-3074 ： ZM_BackFeed 执行出错;

-3075 ： 分配保存 PCX HEAD 文件结构内存出错;

-3076 ： 分配保存 PCX data 内存出错;

-3077 ： 分配保存当前程序运行的文件路径内存出错;

-3078 ： 创建 PCX 文件出错;

-3079 ： 保存 PCX data 出错;

-3080 ： ZM_PrintPCX 执行出错;

-3081 ： 创建 PrinterDC 失败,进行出错处理;

-3082 ： 分配保存 bitmap 内存出错;

-3083 ： ZM_BinGraphicsDownload(执行出错;

-3084 ： ZM_BinGraphicsDel 执行出错;

-3085 ： ZM_BinGraphicsList 执行出错;

-3087 ： ZM_RecallBinGraphics 执行出错;

-3088 ： ZM_RecallBinGraphics 的"name"参数字符长度=0 或>16;

-3089 ： ZM_UserFeed 执行出错;

-3090 ： ZM_UserBackFeed 执行出错;

-3092 ： ZM_ErrorReport 函数中，写端口错误;

-3100 ： SetCommPort 中更改串口出错;

-3101 ： ZM_CutPage 执行出错；

-3102 ： ZM_FeedMedia 执行出错；

-3103 ： ZM_MediaDetect 执行出错；

-3200 to -3400 ： 端口未打开或已经关闭；

-3250 ： 请检查 ZM_DrawBar2D_Pdf417( )函数输入参数；

-3251 to -3257 ：端口未打开或已经关闭；

-3261 ： 在操作 ZM_ErrorReport()之前，端口打开异常

-4000 ： 在操作 OpenPort()函数时,端口打开异常

-4001 ： 在操作 ZM_ErrorReport()时，串口配置错误

-4002 ： 在操作 ZM_ErrorReportEx()时，串口超时