

标签打印机 中文编程手册

使用指南

文档字体规则

文件规则	描述
[表示内容]	在中括号的内容表示该参数为可选项
<ESC>	ASCII 27，当打印机收到以该控制字符为起始的指令将立即响应（即使打印机在错误状态时也将实时回应）
~	ASCII 126，该字符起始的指令用于询问打印机状态
<i>注：200 DPI:1 mm = 8 dots</i>	粗斜体，用于表示批注

设计标签

以下代码内容为一个最简单标签的必备要素，以此为例，详解设计标签时必备的内容和要点。

```
SIZE 58 mm,30 mm
GAP 2 mm
CLS
TEXT 50,50,"4",0,1,1,"DEMO FOR TEXT"
PRINT 1
```

一张标签通常包含三个部分，即系统设定（蓝色部分）、打印内容设定（绿色部分）和执行打印指令（红色部分）。

系统设定包括标签尺寸（SIZE、GAP）和清除缓冲区数据指令（CLS）等。
打印内容设定可以参考本文档卷标内容设计指令内容，本例中系打印文本。
执行打印指令用于打印出设计好的标签，在此指令发送后打印机才执行打印动作。
需要特别注意，在每一条指令结尾需要加入换行符，即字符串“\r\n”或16进制 0x0D 0x0A

系统设定指令

● SIZE

该指令用于设定卷标纸的宽度和长度

指令语法

(1) 英制系统 (英寸 inch)

SIZE m,n

(2) 公制系统 (毫米 mm)

SIZE m mm,n mm

参数	说明
m	标签纸的宽度 (不含背纸)
n	标签纸的长度 (不含背纸)

注:

200 DPI: 1 mm = 8 dots

300 DPI: 1 mm = 12 dots

使用公制单位, 在单位与数字之间必须添加一个空格

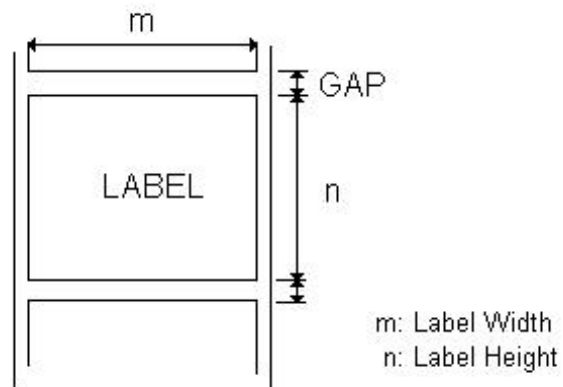
范例

(1) 英制系统 (inch)

SIZE 3.5,3.00

(2) 公制系统 (mm)

SIZE 100 mm,100 mm



● CASHDRAWER

该指令用于产生钱箱控制脉冲

指令语法 :

CASHDRAWER m,t1,t2

ESC p m,t1,t2

参数	说明
m	0, 48 钱箱插座的引脚 2 1, 49 钱箱插座的引脚 5
t1,t2	$0 \leq t1 \leq 255, 0 \leq t2 \leq 255$ 输出由 t1 和 t2 设定的钱箱开启脉冲到由 m 指定的引脚

注:

钱箱开启脉冲高电平时间为 $[t1 \times 2 \text{ ms}]$, 低电平时间为 $[t2 \times 2 \text{ ms}]$

如果 $t2 < t1$, 低电平时间为 $[t1 \times 2 \text{ ms}]$

● GAP

该指令用于定义两张卷标纸间的垂直间距距离

指令语法

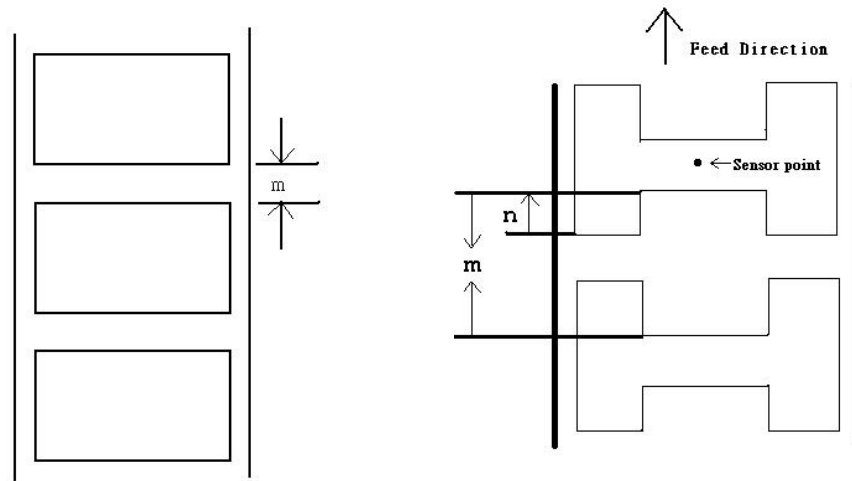
(1) 英制系统 (英寸 inch)

GAP m,n

(2) 公制系统 (毫米 mm)

GAP m mm,n mm

参数	说明
m	两标签纸中间的垂直距离 $0 \leq m \leq 1 \text{ (inch)}, 0 \leq m \leq 25.4 \text{ (mm)}$
n	垂直间距偏移 $n \leq \text{标签纸纸张长度 (inch 或 mm)}$



注:

200 DPI: 1 mm = 8 dots

使用公制单位，在单位与数字之间必须添加一个空格

设置为 0 时表示使用连续纸，使用黑标纸时请使用 *BLINE* 指令

范例

一般垂直间距设定

- (1) 英制系统 (inch)

GAP 0.12, 0

- (2) 公制系统 (mm)

GAP 3 mm, 0 mm

特殊垂直间距设定

- (1) 英制系统 (inch)

GAP 0.30, -0.10

- (2) 公制系统 (mm)

GAP 7.62 mm, -2.54 mm

● *BLINE*

该指令用于设定黑标高度及定义标签印完后标签额外送出的长度

指令语法

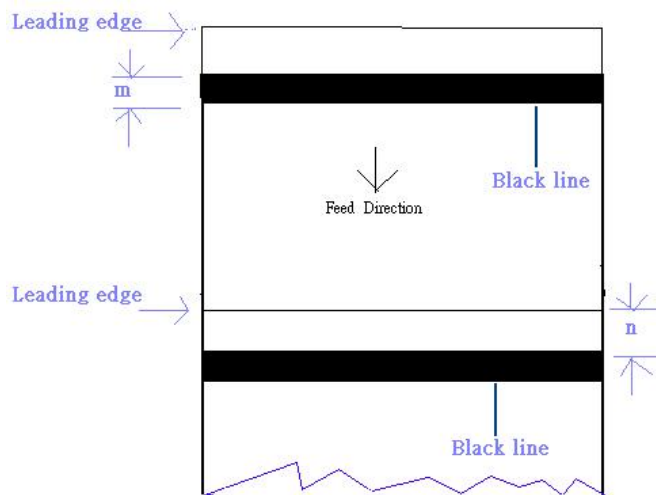
- (1) 英制系统 (inch)

BLINE m, n

- (2) 公制系统 (mm)

BLINE m mm,n mm

参数	说明
m	黑标高度 $0 \leq m \leq 1$ (inch) , $0 \leq m \leq 25.4$ (mm)
n	额外送出纸张长度 $n \leq$ 标签纸纸张长度 (inch 或 mm)



范例:

BLINE 6 mm,3 mm

注:

BLINE 指令不与 GAP 指令同时使用

使用公制单位，在单位与数字之间必须添加一个空格

部分型号发送上述指令，自检页信息 GAP 6 mm、BLINE 3 mm 为正常现象

● OFFSET

该指令用于控制在剥离模式时 (peel-off mode) 每张卷标停止的位置，在打印下一张时打印机会将原先多推出或少推出的部分以回拉方式补偿回来。该指令仅适用于剥离模式。

指令语法

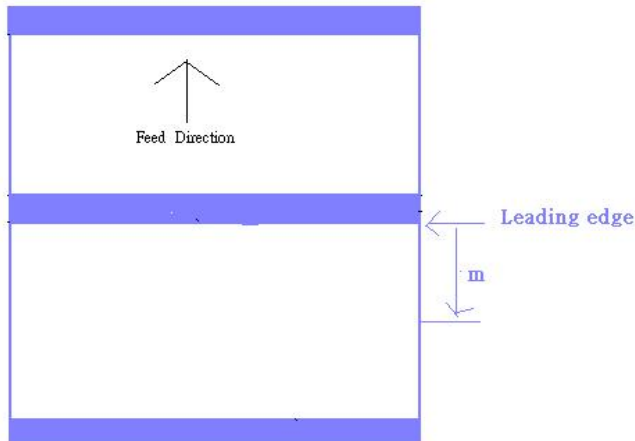
(1) 英制系统 (inch)

OFFSET m

(2) 公制系统 (mm)

OFFSET m mm

参数	说明
m	纸张停止的距离 (inch 或 mm) $0 \leq m \leq 1$ (inch) $0 \leq m \leq 25.4$ (mm)



注:

设定不合理的数值可能引起卡纸

200 DPI: 1 mm = 8 dots

使用公制单位, 在单位与数字之间必须添加一个空格

范例

(1) 英制系统 (inch)

OFFSET 0.5

(2) 公制系统 (mm)

OFFSET 12.7 mm

● SPEED

该指令用于控制打印速度

指令语法 :

SPEED n

参数	说明
n	1.5 设定打印速度为 1.5 " /sec
	2 设定打印速度为 2.0 " /sec
	3 设定打印速度为 3.0 " /sec
	4 设定打印速度为 4.0 " /sec
	5 设定打印速度为 4.0 " /sec
	6 设定打印速度为 4.0 " /sec

范例：

SPEED 4

● DENSITY

该指令用于控制打印时的浓度

指令语法：

DENSITY n

参数	说明
n	0~15
	0：使用最淡的打印浓度
	15：使用最深的打印浓度

范例

DENSITY 7

● DIRECTION

该指令用于定义打印时出纸和打印字体的方向

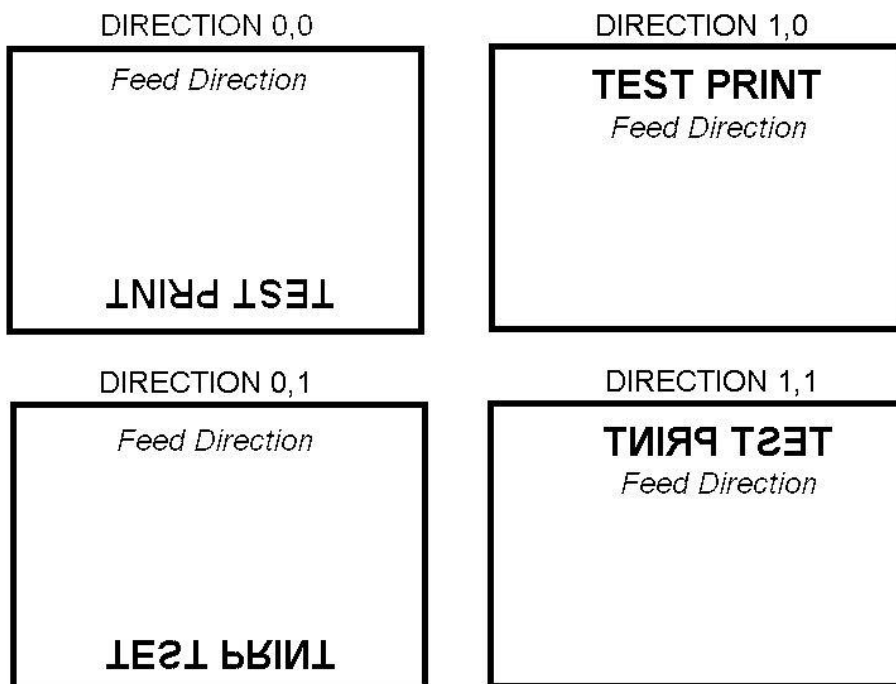
指令语法

DIRECTION n

参数	说明
----	----

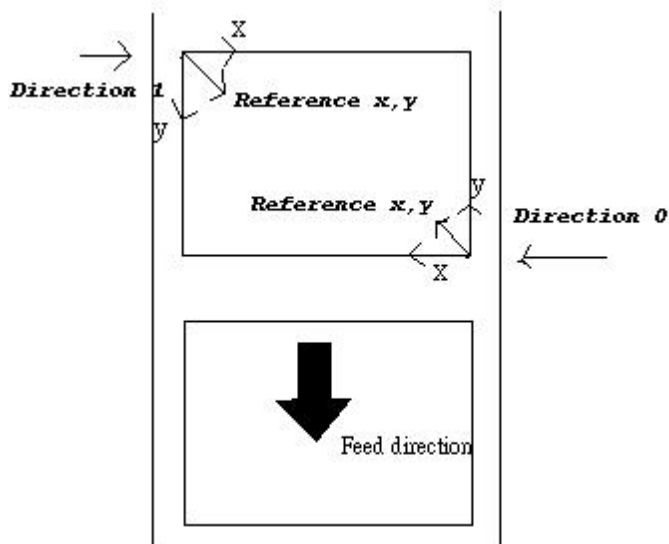
n

0 或 1, 请参考图示



● REFERENCE

该指令用于定义卷标的参考坐标原点。坐标原点位置和打印方向有关，如图所示



指令语法：

REFERENCE x,y

参数	说明
x	水平方向的坐标位置, 单位 dot
y	垂直方向的坐标位置, 单位 dot

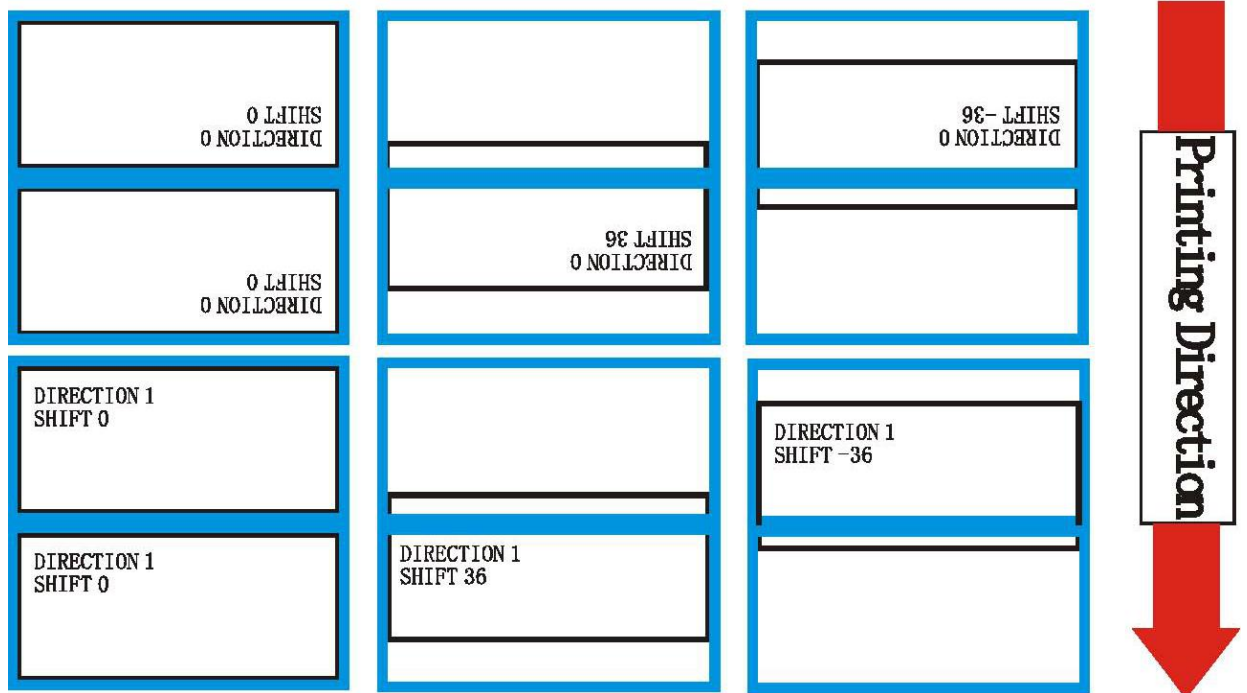
● SHIFT

该指令表示标签打印偏移量多少设置

指令语法

SHIFT n

参数	说明
n	0 或 1, 请参考图示



● COUNTRY

该指令用于选择对应的国际字符集

指令语法 :

COUNTRY n

参数	说明
n	001:USA 002:French 003:Latin America 034:Spanish 039:Italian 044:United Kingdom 046:Swedish 047:Norwegian 049:German

范例：

COUNTRY 001

● CODEPAGE

该指令用于选择对应的国际代码页

指令语法：

CODEPAGE n

参数	说明
n	8-bit codepage 字符集代表 437:United States 850:Multilingual 852:Slavic 860:Portuguese 863:Canadian/French 865:Nordic Windows code page 1250:Central Europe 1252:Latin I 1253:Greek

1254:Turkish

以下代码页仅限于 12×24 dot 英数字体

WestEurope:WestEurope

Greek:Greek

Hebrew:Hebrew

EastEurope:EastEurope

Iran: Iran

IranII: IranII

Latvian:Latvian

Arabic:Arabic

Vietnam:Vietnam

Uygur:Uygur

Thai:Thai

1252:Latin I

1257:WPC1257

1251:WPC1251

866:Cyrillic

858:PC858

747:PC747

864:PC864

1001:PC1001

范例：

CODEPAGE 437

● CLS

该指令用于清除图像缓冲区 (image buffer) 的数据

指令语法：

CLS

参数	说明
----	----

N/A	N/A
-----	-----

注：此项指令必须置于 *SIZE* 指令之后

范例：

CLS

● FEED

该指令用于将标签纸向前推送指定的长度

指令语法：

FEED n

参数	说明
n	$1 \leq n \leq 9999$ ，单位 dot

注：

设定不合理的数值可能引起卡纸或褶皱

200 DPI: 1 mm = 8 dots

300 DPI: 1 mm = 12 dots

范例：

FEED 40

● BACKFEED & BACKUP

该指令用于将标签纸向后回拉指定的长度

指令语法：

BACKFEED n

BACKUP n

参数	说明
n	$1 \leq n \leq 9999$ ，单位 dot

注：

设定不合理的数值可能引起卡纸或碳带褶皱

200 DPI: 1 mm = 8 dots

300 DPI: 1 mm = 12 dots

范例：

BACKFEED 40

BACKUP 40

● FORMFEED

该指令用于控制打印机进一张标签纸

参数	说明
N/A	N/A

范例：

FORMFEED

● HOME

在使用含有间隙或黑标的标签纸时，若不能确定第一张标签纸是否在正确打印位置时，此指令可将标签纸向前推送至下一张标签纸的起点开始打印。标签尺寸和间隙需要在本条指令前设置。

参数	说明
N/A	N/A

注：使用该指令时，纸张高度大于或等于 30 mm

范例：

HOME

● PRINT

该指令用于打印出存储于影像缓冲区内的数据

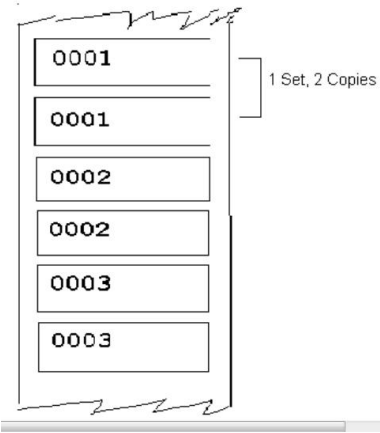
指令语法

PRINT m[,n]

参数	说明
m	指定打印的份数（set） $1 \leq m \leq 65535$
n	每张标签需重复打印的张数 $1 \leq n \leq 65535$

范例

SIZE 60 mm,20 mm
SET COUNTER @1 1
@1="0001"
CLS
TEXT 10,10,"3",0,1,1,@1
PRINT 3,2



● SOUND

该指令用于控制蜂鸣器的频率，可设定 10 阶的声音，每阶声音的长短由第二个参数控制

指令语法：

SOUND level interval

参数	说明
----	----

level	音阶:0-9
interval	间隔时间:1-4095

范例:

SOUND 1,300

SOUND 2,400

SOUND 3,600

● LIMITFEED

该指令用于设定打印机进纸时，若经过所设定的长度仍无法侦测到垂直间距，则打印机在连续纸模式工作。

指令语法：

(1) 英制系统 (inch)

LIMITFEED n

(2) 公制系统 (mm)

LIMITFEED n mm

参数	说明
n	可使用 inch 或 mm

注:

该项设定会存于打印机内存

当打印机初始化时，该设定值会被定为 4 inch

范例

LIMITFEED 12

LIMITFEED 0.5 mm

● SELFTEST

不经自测动作，直接打印自检页信息。

参数	说明
N/A	N/A

范例：

SELFTEST

卷标内容设计指令

● BAR

该指令用于在标签上画线

指令语法

BAR x,y,width,height

参数	说明
x	线条左上角 X 坐标，单位 dot
y	线条左上角 Y 坐标，单位 dot
width	线宽，单位 dot
height	线高，单位 dot

注：200 DPI: 1 mm = 8 dots

范例

BAR 100,100,300,200



● BARCODE

该指令用来画一维条码

指令语法

BARCODE x,y,"code type",height,human readable,rotation,narrow,wide,"content"

参数	说明
x	左上角水平坐标起点，以点（dot）表示
y	左上角垂直坐标起点，以点（dot）表示
height	条形码高度，以点（dot）表示
human readable	0 表示人眼不可识，1 表示人眼可识
rotation	条形码旋转角度，顺时针方向

	0, 不旋转
	90, 顺时针方向旋转 90 度
	180, 顺时针方向旋转 180 度
	270, 顺时针方向旋转 270 度
narrow	窄 bar 宽度, 以点 (dot) 表示
wide	宽 bar 宽度, 以点 (dot) 表示
code type	详见表 1
content	

表 1 一维条码种类

Code Type	说明
128	Code 128, switching code subset A, B, C automatically
128M	Code 128, switching code subset A, B, C manually, 详见表 2
EAN128	Code 128, switching code subset A, B, C automatically
25	Interleaved 2 of 5
25C	Interleaved 2 of 5 with check digits
39	Code 39 full ASCII for TSPL2 printers Code 39 standard for TSPL printers
39C	Code 39 full ASCII with check digit for TSPL2 printers Code 39 standard with check digit for TSPL printers
39S	Code 39 standard for TSPL2 printers
93	Code 93
EAN13	EAN 13
EAN13+2	EAN 13 with 2 digits add-on
EAN13+5	EAN 13 with 5 digits add-on
EAN8	EAN 8
EAN8+2	EAN 8 with 2 digits add-on
EAN8+5	EAN 8 with 5 digits add-on
CODA	Codabar
UPCA	UPC-A
UPCA+2	UPC-A with 2 digits add-on

UPCA+5	UPC-A with 5 digits add-on
UPCE	UPC-E
UPCE+2	UPC-E with 2 digits add-on
UPCE+5	UPC-E with 5 digits add-on
CPOST	China post code
MSI	MSI code
MSIC	MSI code with check digital
PLESSEY	PLESSEY code
ITF14	ITF 14 code
EAN14	EAN 14 code

表 2 CODE128M 控制方法

Control code	A	B	C
096	FNC3	FNC3	NONE
097	FNC2	FNC2	NONE
098	SHIFT	SHIFT	NONE
099	CODE C	CODE C	NONE
100	CODE B	FNC4	CODE B
101	FNC4	CODE A	CODE A
102	FNC1	FNC1	FNC1
103	Start (CODE A)		
104	Start (CODE B)		
105	Start (CODE C)		

使用 "!" 为条形码 subset 的控制字符, 后面加三码(如上表所示), 若无指定 code 128M 的起始 subset, 系统定值为 subset B

表 3 不同类型一维条码支持窄宽比

	narrow : wide 1:1	narrow : wide 1:2	narrow : wide 1:3	narrow : wide 2:5	narrow : wide 3:7
128	10x	-	-	-	-
EAN128	10x	-	-	-	-

25	-	10x	10x	5x	-
25C	-	10x	10x	5x	-
39	-	10x	10x	5x	-
39C	-	10x	10x	5x	-
93	-	-	10x	-	-
EAN13	8x	-	-	-	-
EAN13+2	8x	-	-	-	-
EAN13+5	8x	-	-	-	-
EAN 8	8x	-	-	-	-
EAN 8+2	8x	-	-	-	-
EAN 8+5	8x	-	-	-	-
CODA	-	10x	10x	5x	-
POST	1x	-	-	-	-
UPCA	8x	-	-	-	-
UPCA+2	8x	-	-	-	-
UPCA+5	8x	-	-	-	-
UPCE	8x	-	-	-	-
UPCE+2	8x	-	-	-	-
UPCE+5	8x	-	-	-	-
CPOST	-	-	-	-	1x
MSI	-	-	10x	-	-
MSIC			10x		-
PLESSY	-	-	10x	-	-
ITF14	-	10x	10x	5x	-
EAN14	-	-	-	5x	-

表 4 最大字符数

Barcode type	Maximum bar code length
128	-
EAN128	-
25	-

25C	-
39	-
39C	-
93	-
EAN13	12
EAN13+2	14
EAN13+5	17
EAN 8	7
EAN 8+2	9
EAN 8+5	12
CODA	-
POST	5,9,11
UPCA	11
UPCA+2	13
UPCA+5	16
UPCE	6
UPCE+2	8
UPCE+5	11
CPOST	-
MSI	-
MSIC	
PLESSY	-
ITF14	13
EAN14	13

范例

BARCODE 100,100,"39",96,1,0,2,4,"1000"

BARCODE 10,10,"128M",48,1,0,2,2,"!104!096ABCD!101EFGH"

(上述 CODE 128M 的范例为使用 CODE B 起始的条码,其中!096(FNC3)与 ABCD 均以 CODE B 方式编码;!101 为将原编码方式由 CODE B 转换为 CODE A,后续的 EFGH 即为使用 CODE A 方式编码)

● BOX

该指令用于在卷标上绘制矩形方框

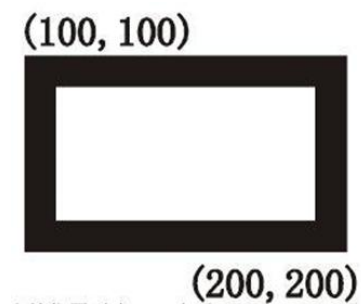
指令语法

BOX x_start,y_start,x_end,y_end,line thickness

参数	说明
x_start	方框左上角 X 坐标，单位 dot
y_start	方框左上角 Y 坐标，单位 dot
x_end	方框右下角 X 坐标，单位 dot
y_end	方框右下角 Y 坐标，单位 dot
line thickness	方框线宽，单位 dot

范例

BOX 100,100,200,200,5



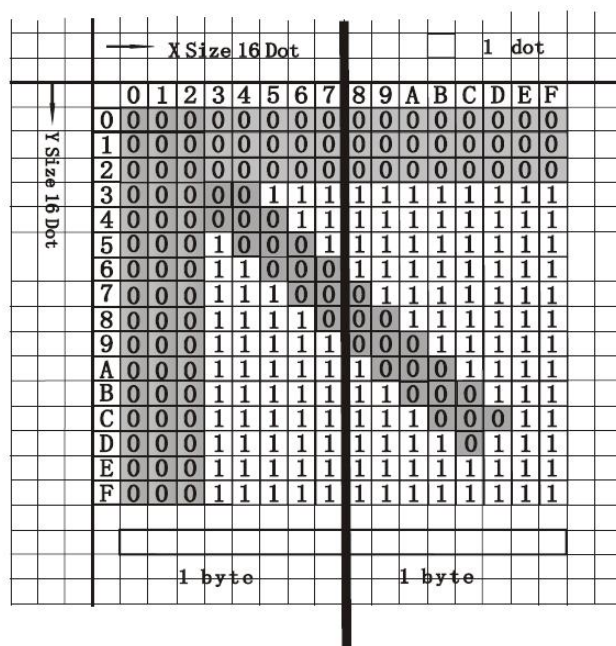
● BITMAP

该指令用于在卷标上绘制位图（非 BMP 格式图档）

指令语法

BITMAP x,y,width,height,mode,bitmap data

参数	说明
x	位图左上角 X 坐标
y	位图左上角 Y 坐标
width	位图的宽度，单位 byte
height	位图的高度，单位 dot
mode	位图绘制模式
0	OVERWRITE
1	OR
2	XOR



ROW (Y-axis)	L-Byte		R-Byte	
	Binary	Hexadecimal	Binary	Hexadecimal
0	0000 0000	00	0000 0000	00
1	0000 0000	00	0000 0000	00
2	0000 0000	00	0000 0000	00
3	0000 0111	07	1111 1111	00
4	0000 0011	03	11111111	FF
5	0001 0001	11	1111 1111	FF
6	0001 1000	18	1111 1111	FF
7	0001 1100	1C	0111 1111	7F
8	0001 1110	1E	0011 1111	3F
9	0001 1111	1F	0001 1111	1F
A	0001 1111	1F	1000 1111	8F
B	0001 1111	1F	1100 0111	C7
C	0001 1111	1F	1110 0011	E3
D	0001 1111	1F	1111 0111	F7
E	0001 1111	1F	1111 1111	FF
F	0001 1111	1F	1111 1111	FF

范例

16 进制指令代码	ASCII 指令代码
53 49 5A 45 20 34 2C 32 0D 0A 47 41	SIZE 4,2
50 20 30 2C 30 0D 0A 43 4C 53 0D	GAP 0,0
0A 42 49 54 4D 41 50 20 32 30 30 2C	CLS
32 30 30 2C 32 2C 31 36 2C 30 2C 00	BITMAP 200,200,2,16,0,data
00 00 00 00 00 07 FF 03 FF 11 FF 18	PRINT 1,1
FF 1C 7F 1E 3F 1F 1F 1F 8F 1F C7	
1F E3 1F E7 1F FF 1F FF 0D 0A 50	
52 49 4E 54 20 31 2C 31 0D 0A	

● PUTBMP

该指令用来打印单色 BMP 图片文件

指令语法

PUTBMP x,y,"filename"

参数	说明
x	水平方向起始点坐标，以点（dot）表示
y	垂直方向起始点坐标，以点（dot）表示
filename	欲打印的文件名称（需已下载于打印机缓存）

注：该指令仅支持单色 BMP 图片文件

```
C:\BMP-PCX>DIR
Volume in drive C is WIN98
Volume Serial Number is 4140-4735

Directory of C:\BMP-PCX

06/08/2008  03:06 PM    <DIR>
06/08/2008  03:06 PM    <DIR>
06/08/2008  03:56 PM             12,430 GP.bmp
06/08/2008  03:10 PM             1,181 GP.pcx
                2 File(s)          13,611 bytes
                2 Dir(s)  8,802,189,312 bytes free

C:\BMP-PCX>COPY CON LPT1
DOWNLOAD  "GP.BMP",12430,^Z
1 file(s) copied.

C:\BMP-PCX>COPY GP.BMP/B LPT1
1 file(s) copied.

C:\BMP-PCX>COPY CON LPT1
SIZE 3,2.5
GAP 0,0
CLS
PUTBMP 100,100,"GP.BMP"
PRINT 1,1
^Z
1 file(s) copied.
C:\BMP-PCX>_
```

范例

```
PUTBMP 100,100,"LOGO.BMP"
```

注：`^Z` 表示 `<Ctrl>+<z>` 或者是 `<F6>` 键

● PUTPCX

该指令用于打印单色 PCX 格式图片文件

指令语法

```
PUTPCX x,y,"filename"
```

参数	说明
x	水平方向起始点坐标，以点（dot）表示
y	垂直方向起始点坐标，以点（dot）表示
filename	欲打印的文件名称（需已下载于打印机缓存）

```
C:\BMP-PCX>DIR
Volume in drive C is WIN98
Volume Serial Number is 4140-4735

Directory of C:\BMP-PCX

01/03/2005  01:06 PM    <DIR>          .
01/03/2005  01:06 PM    <DIR>          ..
01/03/2005  01:52 PM             12,430 TSC.bmp
01/03/2005  01:10 PM             1,181 TSC.pcx
                2 File(s)             13,611 bytes
                2 Dir(s)   8,802,189,312 bytes free

C:\BMP-PCX>COPY CON LPT1
DOWNLOAD "TSC.PCX",1181,^Z
                1 file(s) copied.

C:\BMP-PCX>COPY TSC.PCX/B LPT1
                1 file(s) copied.

C:\BMP-PCX>COPY CON LPT1
SIZE 4,2.5
GAP 0,0
CLS
PUTPCX 100,100,"TSC.PCX"
PRINT 1,1
^Z
                1 file(s) copied.

C:\BMP-PCX>_
```

范例

```
PUTPCX 10,10,"TSC.PCX"
```

注：`^Z` 表示 `<Ctrl>+<z>` 或者是 `<F6>` 键

● ERASE

该指令用于清除影像缓冲区部分区域的数据

指令语法

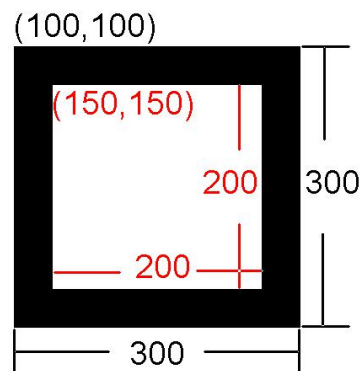
ERASE x_start,y_start,x_width,y_height

参数	说明
x_start	清除区域的左上角 X 坐标，单位 dot
y_start	清除区域的左上角 Y 坐标，单位 dot
x_width	清除区域宽度，单位 dot
y_height	清除区域高度，单位 dot

范例

BAR 100,100,300,300

ERASE 150,150,200,200



● REVERSE

将指定的区域反相打印

指令语法

REVERSE x_start,y_start,x_width,y_height

参数	说明
x_start	反相区域左上角 X 坐标，单位 dot
y_start	反相区域左上角 Y 坐标，单位 dot
x_width	反相区域宽度，单位 dot
y_height	反相区域高度，单位 dot

范例

REVERSE 100,100,200,200



● TEXT

该指令用于打印字符串

指令语法

TEXT x,y,"font",rotation,x-multiplication,y-multiplication,"content"

参数	说明
x	文字 X 方向起始点坐标
y	文字 Y 方向起始点坐标
font	字体名称
1	8×12 dot 英数字体
2	12×20 dot 英数字体
3	16×24 dot 英数字体
4	24×32 dot 英数字体
5	32×48 dot 英数字体
6	14×19 dot 英数字体 OCR-B
7	21×27 dot 英数字体 OCR-B
8	14×25 dot 英数字体 OCR-A
9	9×17 dot 英数字体
10	12×24 dot 英数字体
TSS16. BF2	简体中文 16×16 (GB 码)
TSS20. BF2	简体中文 20×20 (GB 码)
TST24. BF2	繁体中文 24×24 (大五码)
TSS24. BF2	简体中文 24×24 (GB 码)
K	韩文 24×24Font (KS 码)
TSS32. BF2	简体中文 32×32 (GB 码)
rotation	文字旋转角度 (顺时针方向)
0	0 度
90	90 度
180	180 度
270	270 度

x-multiplication	X 方向放大倍率 1-10
y-multiplication	Y 方向放大倍率 1-10

注:

五号字英文字母仅可打印大写字母

若要打印双引号时(")在程序中请使用\["]来打印双引号

若要打印 0D(hex) 字符时, 请在程序中使用\[R]来打印 CR

若要打印 0A(hex) 字符时, 请在程序中使用\[A]来打印 LF

打印机所支持的字体以具体型号为准

范例

TEXT 100,100,"4",0,1,1,"DEMO FOR TEXT"

● QRCODE

该指令用来打印二维码

程序语法:

QRCODE x,y,ECC level,cell width,mode,rotation,"data string"

参数	说明
x	二维码水平方向起始点坐标
y	二维码垂直方向起始点坐标
ECC level	选择 QRCODE 纠错等级
L	7%
M	15%
Q	25%
H	30%
cell width	二维码宽度 1-10
mode	手动/自动编码
A	Auto
M	Manual
rotation	旋转角度 (顺时针方向)
0	0 度

90	90 度
180	180 度
270	270 度
data string	编码的字符串

范例

SIZE 60 mm,30 mm

GAP 2 mm

CLS

QRCODE 20,20,L,4,A,0,"www.baidu.com"

PRINT 1,1

询问打印机状态指令

● <ESC>!?

询问打印机状态指令为立即响应型指令，该指令控制字符是以<ESC>(ASCII27,escape 字符)为控制字符.即使打印机在错误状态中仍能透过RS-232 回传一个 byte 资料来表示打印机状态，若回传值为 0x00 则表示打印机处于正常的状态

指令语法

<ESC>!?

16 进制：1B 21 3F

10 进制：27 33 63

返回值（16 进制）	打印机状态
00	正常待机
01	开盖
02	卡纸
03	卡纸、开盖
04	缺纸
05	缺纸、开盖
08	无碳带
09	无碳带、开盖
0A	无碳带、卡纸
0B	无碳带、卡纸、开盖
0C	无碳带、缺纸
0D	无碳带、缺纸、开盖
10	暂停打印
20	正在打印
80	其他错误

返回值可转换为二进制，按位（bit）进行观察判定，值为 1 时表示处于当前状态，状态参考下表

bit	打印机状态（值为 1 时）
0	开盖
1	卡纸
2	缺纸
3	无碳带
4	暂停打印
5	正在打印
6	外盖打开(option)
7	过热(option)

● <ESC>!R

该指令命令打印机重新开机

指令语法

<ESC>!R
16 进制： 1B 21 52
10 进制： 27 33 82

● ~!@

该指令回复打印机已打印的里程，以作为维护的参考，打印机仅回复整数部分的里程，小数的部分将会被忽略，传回值将以 ASCII 字符的格式传回，以 0x0d 作为结尾

指令语法

~!@

● ~!A

该指令用于询问打印机内存大小，回传值以 10 进制字符表示，以 0x0d 作为结尾

指令语法

~!A

● ~!C

该指令透过串口回传打印机是否有安装 RTC

返回值（16 进制）	打印机状态
00	未安装 RTC
01	已安装 RTC

● ~!D

该指令用于输入除错模式（Dump Mode），除错模式下所有指令会以 16 进制编码打出，重启打印机可退出该模式

指令语法

~!D

● ~!F

该指令用来询问打印机内存中所储存的文件名，打印机回复 ASCII 字符文件名，每个文件名以 0d(hex) 作为分隔，最后一个文件名以 0x0d,0x1a 作为结束

指令语法

~!F

● ~!I

该指令用于询问打印机所设定的 codepage code，回传格式如下：

code page, country code

ex: 8 bit: 437, 001

有关回传信息，请参考 CODEPAGE 指令

指令语法

~!I

● ~!T

该指令回复打印机的型号

指令语法

~!T

文件管理指令

● DOWNLOAD

定义文件可被储存于打印机的 DRAM 中。下载至打印机的文件可区分为两种：程序文件及资料文件(包括字符型文件、PCX 图形文件和 BMP 图形文件等)

指令语法

1. Download 程序文件

DOWNLOAD [n,]"FILENAME.BAS"

参数	说明
n	指定文件存储位置 未指定时文件被储存于 DRAM F: 文件存储于 FLASH
FILENAME. BAS	储存于打印机中的文件名称

注:

大小写将表示不同的文件名称

文件扩展名必须是 BAS

DRAM 的中的文件在打印机关机后丢失

若要将程序存在打印机内，程序的第一行须加 DOWNLOAD，程序最后一行需以 EOP 做结束

2. Download 资料文件

储存数据于打印机内存的格式如下

DOWNLOAD [n,]"FILENAME",DATA SIZE,DATA CONTENT...

参数	说明
n	指定文件存储位置 未指定时文件被储存于 DRAM F: 文件存储于 FLASH

FILENAME	储存于打印机中的文件名称
DATA SIZE	不含文件头的实际文件大小，以字节数计算
DATA CONTENT	下载到打印机的数据

注：

行与行之间的资料以 *CR (0x0D)* 和 *LF (0x0A)* 做分隔

若不指定储存的位置，则文件一律载至 *DRAM*，存于 *DRAM* 的文件会因电源关闭而消失

范例

```

DOWNLOAD "EXAMPLE.BAS"
SIZE 56 mm,40 mm
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 1
REFERENCE 0,0
SET PEEL OFF
CLS
TEXT 20,20,"3",0,1,1,"EXAMPLE PROGRAM"
PRINT 1
EOP
RUN "EXAMPLE.BAS"

```

● EOP

程序结束点，需将该指令放在程序的最后一行

程序语法

```
EOP
```

范例

```

DOWNLOAD "DEMO.BAS"
SIZE 3,4
GAP 0,0

```

```
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET PEEL OFF
CLS
TEXT 100,100,"3",0,1,1,"DEMO PROGRAM"
PRINT 1
EOP
```

● FILES

该指令可打印出储存在打印机内存的文件名称

程序语法：

```
FILES
```

范例

请依照下列步骤列出储存于打印机内存的文件名称

```
C:\>COPY CON LPT1<ENTER>
SET DEBUG LABEL<ENTER>
FILE<ENTER>
<CTRL>Z
C:\>
```

注：<ENTER>、<CTRL>、<CTRL>代表 PC 键盘上的相对按键

● KILL

该命令用来删除储存于打印机内存的文件

程序语法：

```
KILL [n,]"FILENAME"
```

参数	说明
FILENAME	删除的文件名称，注意有大小写

*	表示删除所有文件
n	未指定时：删除 DRAM 中的文件
F:	删除主板 FLASH 中的文件

范例

```
KILL "DEMO.BAS"
KILL "*.PCX"
KILL F,"FILENAME"
KILL ""
```

● MOVE

该指令可将存在 DRAM 的数据写到 flash memory

程序语法：

```
MOVE
```

● RUN

此命令是用来执行存贮在打印机内的文件

程序语法：

```
RUN "FILENAME.BAS"
```

范例

```
C:\>COPY CON LPT1<ENTER>
RUN "DEMO.BAS"<ENTER>
<CTRL><Z><ENTER>
C:\>
```

BASIC 指令和函数

● ABS ()

该函数回复整数或浮点数的绝对值

程序语法

ABS(-100)

ABS(-99.99)

ABS(VARIABLE)

范例

DOWNLOAD "TEST.BAS"

SIZE 3,4

GAP 0,0

DENSITY 8

SPEED 3

DIRECTION 0

REFERENCE 0,0

SET PEEL OFF

CLS

A=ABS(-100)

B=ABS(-50.98)

C=-99.99

TEXT 100,100,"3",0,1,1,STR\$(A)

TEXT 100,150,"3",0,1,1,STR\$(B)

TEXT 100,200,"3",0,1,1,STR\$(ABS(C))

PRINT 1

EOP

● **ASC()**

该函数返回字符的 ASCII 码

程序语法

ASC("A")

范例

```
DOWNLOAD "TEST.BAS"  
SIZE 3,4  
GAP 0,0  
DENSITY 8  
SPEED 3  
DIRECTION 0  
REFERENCE 0,0  
SET PEEL OFF  
CLS  
CODE1=ASC("A")  
TEXT 100,100,"3",0,1,1,STR$(CODE1)  
PRINT 1  
EOP
```

● **CHR\$()**

该函数返回所指定的 ASCII code 字符

程序语法

CHR\$(n)

范例

```
DOWNLOAD "TEST.BAS"  
SIZE 3,4  
GAP 0,0  
DENSITY 8
```



```
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET PEEL OFF
CLS
A=65
WORD$=CHR$(A)
TEXT 100,100,"3",0,1,1,WORD$
PRINT 1
EOP
```

● END

主程序的结束点

程序语法

```
END
```

范例

```
DOWNLOAD "DEMO.BAS"
SIZE 4,2
GAP 0,0
DIRECTION 1
CLS
TEXT 200,60,"4",0,1,1,"END COMMAND TEST"
X=300
Y=200
X1=500
Y1=400
GOSUB DR_LINE
PRINT 1
END
:DR_LINE
FOR I=1 TO 100 STEP 10
```

BOX X+I,Y+I,X1-I,Y1-I,5

NEXT

RETURN

EOP

DEMO

● EOF ()

该命令用来判断文件指针是否已到文件的结束位置

程序语法

EOF (File Handle)

参数	说明
File handle	0 或 1

返回值	说明
非零	指针已到文件结束位置
0	指针未到文件结束位置

范例

DOWNLOAD "DATA",16,COMPUTER

2000

DOWNLOAD "DEMO.BAS"

SIZE 3,3

GAP 0.0,0

DIRECTION 1

CLS

OPEN "DATA",0

SEEK 0,0

Y=110

TEXT 10,10,"3",0,1,1,"*****EOF TEST*****"

:A

```

Temp$=""
READ 0,ITEM$,P
TEXT
10,Y,"2",0,1,1,ITEM$+"$"+STR$(P)+"[EOF(0)="+STR$(EOF(0))+"]"
BARCODE 10,Y+25,"39",40,1,0,2,4,"PRICE-"+STR$(P)
Y=Y+100
IF EOF(0)=0 THEN
GOTO A
ENDIF
PRINT 1
EOP
RUN "DEMO.BAS"

```

● OPEN

该指令用于开启储存于打印机内存的文件，打印机最多能同时一次开启两个文件，使用该指令时文件需已储存于打印机内。

程序语法

```
OPEN "filename",file handle
```

参数	说明
filename	存储于内存的文件名称
file handle	0 或 1

范例

```

DOWNLOAD "DATA1",56,COMPUTER
2000
12
MOUSE
500
13
KEYBOARD
300

```

100

DOWNLOAD "DATA2",56,Computer

3000

32

Mouse

900

93

Keyboard

700

700

DOWNLOAD "DEMO.BAS"

SIZE 3,1

GAP 0,0

DENSITY 8

SPEED 4

DIRECTION 1

REFERENCE 0,0

SET CUTTER OFF

SET PEEL OFF

I=1

Y=100

GOSUB OpenData

:Start

CLS

TEXT 10,10,"3",0,1,1,"*****OPEN COMMAND TEST*****"

ITEM\$=""

READ 0,ITEM\$,P,Q

TEXT

10,Y,"2",0,1,1,ITEM\$+"\$"+STR\$(P)+"[EOF(0)=" +STR\$(EOF(0))+"]"

BARCODE

10,Y+25,"39",40,1,0,2,4,"PRICE*" +STR\$(Q)+"=" +STR\$(P*Q)

```

Y=Y+100
PRINT 1
Y=100
IF EOF(0)=1 THEN
GOSUB OpenData
ENDIF
IF EOF(0)=0 THEN
GOTO Start
ENDIF
END
:OpenData
IF I=1 THEN
OPEN "DATA1",0
ENDIF
IF I=2 THEN
OPEN "DATA2",0
ENDIF
SEEK 0,0
IF I>2 THEN
END
ENDIF
I=I+1
RETURN
EOP
DEMO

```

● READ

该指令用于读取已存于打印机内存的文件

程序语法

READ file handle,variables

参数	说明
----	----

file handle	0 或 1
variables	字符串、整数或浮点变量

范例

DOWNLOAD "DATA1",20,COMPUTER

2000

12

DOWNLOAD "DATA2",16,Mouse

900

93

DOWNLOAD "DEMO.BAS"

SIZE 3,1

GAP 0,0

DENSITY 8

SPEED 4

DIRECTION 1

REFERENCE 0,0

SET CUTTER OFF

SET PEEL OFF

I=0

Y=100

OPEN "DATA1",0

OPEN "DATA2",1

SEEK 0,0

SEEK 1,0

:Start

CLS

TEXT 10,10,"3",0,1,1,"*****READ COMMAND TEST*****"

TEXT 10,50,"3",0,1,1,"OPEN-READ DATA"+STR\$(I+1)

ITEM\$=""

READ I,ITEM\$,P,Q

```

TEXT 10,Y,"2",0,1,1,ITEM$+"$"+STR$(P)
BARCODE
10,Y+25,"39",40,1,0,2,4,"PRICE*"+STR$(Q)+"="+STR$(P*Q)
Y=Y+100
PRINT 1
Y=100
IF I<1 THEN
I=I+1
GOTO Start
ELSE
END
ENDIF
EOP
DEMO

```

● SEEK

该指令用来移动文件指针到某一特定的位置.

程序语法

```
SEEK file handle,offset
```

参数	说明
file handle	0 或 1
offset	文件指标的偏移量

范例

```

DOWNLOAD "DATA",12,1234567890
DOWNLOAD "TEST.BAS"
SIZE 4,1.5
GAP 0,0
DIRECTION 1
REFERENCE 0,0
CLS

```

```

OPEN "DATA",0
SEEK 0,4
READ 0,Num$
TEXT 100,10,"3",0,1,1,"SEEK COMMAND TEST"
BAR 100,40,300,4
TEXT 100,60,"3",0,1,1,"SHIFT 4 CHARACTERS"
TEXT 100,110,"3",0,1,1,Num$
BAR 100,140,300,4
SEEK 0,0
READ 0,Num$
TEXT 100,160,"3",0,1,1,"SHIFT 0 CHARACTERS"
TEXT 100,210,"3",0,1,1,Num$
PRINT 1
EOP
TEST

```

● LOF ()

该指令可回传已打开文件的文件大小，以 byte 表示

程序语法

```
LOF("FILENAME")
```

参数	说明
FILENAME	已下载在打印机内存的文件名称

范例

```

DOWNLOAD "DATA1",10,1234567890
DOWNLOAD "DATA2",15,ABCDEFGHIJKLMNO
DOWNLOAD "LoFTest.BAS"
SIZE 4,1.5
GAP 0,0
DIRECTION 1
CLS

```



```

OPEN "DATA1",0
OPEN "DATA2",1
TEXT 10,20,"4",0,1,1,"LOF() FUNCTION TEST"
J=LOF("DATA1")
K=LOF("DATA2")
TEXT 10,140,"3",0,1,1,"DATA1 IS: "+STR$(J)+" Bytes"
TEXT 10,200,"3",0,1,1,"DATA2 IS: "+STR$(K)+" Bytes"
PRINT 1
EOP
LofTest

```

● FREAD\$()

该指令用于读取已打开文件内所指定 byte 数的数据

程序语法

```
FREAD$(file handle,byte)
```

参数	说明
file handle	0 或 1
byte	欲读取数据的 byte 数

范例

```

DOWNLOAD "DATA1",10,1234567890
DOWNLOAD "DATA2",15,ABCDEFGHIJKLMNO
DOWNLOAD "OPEN2.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
CLS
OPEN "DATA1",0
OPEN "DATA2",1
SEEK 0,0
SEEK 1,0

```

```

Y$=FREAD$(0,6)
Z$=FREAD$(1,6)
TEXT 10,100,"3",0,1,1,"FREAD$(0,6) IS: "+Y$
TEXT 10,150,"3",0,1,1,"FREAD$(1,6) IS: "+Z$
PRINT 1
EOP
RUN "OPEN2.BAS"

```

● FOR...NEXT LOOP

循环指令可自动执行的程序，直到条件满足为止。请勿由循环外部直接跳到循环内部执行，否则将发生无法预期的错误。

程序语法

```

FOR variable = start TO end STEP increment
statement; start < end
[EXITFOR]
NEXT

```

参数	说明
variable	变量名称最多可达 8 个字符
start	可为整数或浮点数
end	可为整数或浮点数
increment	可为整数或浮点数，正数或负数

注：不支持嵌套循环

范例

```

DOWNLOAD "TEST.BAS"
SIZE 4,2.5
GAP 0,0
CLS
FOR I=1 TO 10 STEP 1
TEXT 100,10+30*(I-1),"3",0,1,1,STR$(I)
NEXT

```

```

FOR I=1 TO 1000 STEP 100
TEXT 200,10+((I-1)/10)*3,"3",0,1,1,STR$(I)
NEXT
FOR I=110 TO 10 STEP -10
TEXT 300,10+(ABS(I-110))*3,"3",0,1,1,STR$(I)
NEXT
FOR I=1 TO 5 STEP 0.5
IF I-INT(I)=0 THEN
Y=10+60*(I-1)
ELSE
Y=Y+30
ENDIF
TEXT 400,Y,"3",0,1,1,STR$(I)
NEXT
PRINT 1
EOP
TEST

```

● IF…THEN…ELSE…ENDIF

条件判断式指令

程序语法

```

IF condition THEN
Statements
ENDIF

```

```

IF condition THEN
Statements
ELSE
Statements
ENDIF

```

参数	说明
----	----

condition	可使用的运算符有<, >, =
statement	程序仅能用一行

范例

```

DOWNLOAD "TEST.BAS"
SIZE 3,4
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 1
REFERENCE 0,0
SET PEEL OFF
CLS
A=90
B=5
C$=""
D$=""
:L1
IF A>100 THEN
GOTO L2
ELSE
A=A+10
ENDIF
CLS
C$=STR$(A)+" IS SMALLER THAN 100"
TEXT 10,10,"2",0,1,1,C$
PRINT 1
GOTO L1
:L2
A=A+B
D$=STR$(A)+" IS LARGER THAN 100"
CLS
TEXT 10,100,"2",0,1,1,D$

```

```
PRINT 1  
END  
EOP
```

● GOSUB...RETURN

该指令可由主程序跳到子程序执行再跳回主程序

程序语法

```
GOSUB LABEL  
statement  
END
```

```
:LABEL  
statement  
RETURN
```

参数	说明
LABEL	子程序的起始点，LABEL 名称不可超过 8 个字符

范例

```
DOWNLOAD "GOSUB1.BAS"  
SIZE 4,3  
GAP 0,0  
DIRECTION 1  
CLS  
TEXT 10,10,"3",0,1,1,"GOSUB & RETURN COMMAND TEST"  
GOSUB DR_BOX  
PRINT 1  
END  
:DR_BOX  
FOR I=21 TO 81 STEP 10  
BOX 80+I,80+I,80+300-I,80+300-I,5  
NEXT
```

RETURN

EOP

GOSUB1

● GOTO

此命令是用来分行指定的标签。标签不能超过 8 个字符。

程序语法

GOTO LABEL

:LABEL

范例

DOWNLOAD "GOTO1.BAS"

SIZE 4,3

GAP 0,0

DIRECTION 1

CLS

A=0

TOTAL=0

:START

IF A<100 THEN

GOTO SUM

ELSE

GOTO PRTOUT

ENDIF

:SUM

A=A+1

TOTAL=TOTAL+A

GOTO START

:PRTOUT

B\$="THE SUMMATION OF 1..100 IS "+STR\$(TOTAL)

TEXT 10,100,"2",0,1,1,B\$

PRINT 1

END

EOP

● REM

该指令用于程序批注

程序语法

REM

范例

```
REM *****  
REM This is a demonstration program*  
REM *****  
DOWNLOAD "REMARK.BAS"  
SIZE 4,3  
GAP 0,0  
DIRECTION 1  
CLS  
TEXT 50,50,"3",0,1,1,"REMARK DEMO PROGRAM"  
REM TEXT 50,100,"3",0,1,1,"REMARK DEMO PROGRAM2"  
PRINT 1,1  
EOP  
REMARK
```

● INT()

该函式将传回浮点数的整数部份

程序语法

INT(n)

参数	说明
n	n 可以是正数或负数、浮点数或数学表达式

范例

```
DOWNLOAD "DEMO.BAS"
SIZE 3,4
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 1
REFERENCE 0,0
SET PEEL OFF
CLS
A=INT(99.99)
B=INT(-199.89)
C=INT(80)
TEXT 50,100,"3",0,1,1,"INT(99.99) = "+STR$(A)
TEXT 50,150,"3",0,1,1,"INT(-199.89) = "+STR$(B)
TEXT 50,200,"3",0,1,1,"INT(80) = "+STR$(C)
PRINT 1
EOP
DEMO
```

● LEFT\$()

该函式传回字符串中最左边指定的字符

程序语法

```
LEFT$(X$,n)
```

参数	说明
X\$	欲处理的字符串
n	欲截取回传的字符数

范例

```
DOWNLOAD "TEST.BAS"
SIZE 4,1
```



```

GAP 0,0
DIRECTION 1
A$="BARCODE PRINTER DEMO PRINTING"
C$=LEFT$(A$,10)
CLS
TEXT 10,10,"2",0,1,1,A$
TEXT 10,100,"2",0,1,1,"10 LEFT 10 CHARS: "+C$
PRINT 1
EOP
TEST

```

● LEN()

该函数返回字符串的长度

程序语法

```
LEN(string)
```

参数	说明
string	欲量测的字符串

范例

```

DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
B=LEN(A$)
CLS
TEXT 10,10,"3",0,1,1,A$
TEXT 10,50,"3",0,1,1,"STRING LENGTH="+STR$(B)
PRINT 1
EOP
DEMO

```

● MID\$()

该函式用来传回字符串中某几个字符

程序语法

MID\$(string,m,n)

参数	说明
string	欲处理的字符串
m	字符串中第 m 个位置起始位置 $1 \leq m \leq \text{字符串长度}$
n	欲传回的字符数

范例

```
DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
E$=MID$(A$,11,10)
CLS
TEXT 10,10,"3",0,1,1,A$
TEXT 10,40,"3",0,1,1,"10 MIDDLE CHARS: "+E$
PRINT 1
EOP
DEMO
```

● RIGHT\$()

该函式将从字符串的最右边传回 n 个字符

程序语法

RIGHT\$(X\$,n)

参数	说明
X\$	欲处理的字符串
n	从字符串的最右边传回 n 个字符

范例

```

DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
D$=RIGHT$(A$,10)
CLS
TEXT 10,10,"3",0,1,1,A$
TEXT 10,80,"3",0,1,1,"10 RIGHT CHARS: "+D$
PRINT 1
EOP
DEMO

```

● STR\$()

该函数将数字转换为字符串

程序语法

STR\$(n)

参数	说明
n	欲处理的数字

范例

```

DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1

```

```

A$="ABCDEFGHJKLMNOPQRSTUVWXYZ"
F=100
G=500
H$=STR$(F+G)
CLS
TEXT 10,10,"3",0,1,1,A$
TEXT 10,60,"3",0,1,1,"F="+STR$(F)
TEXT 10,110,"3",0,1,1,"G="+STR$(G)
TEXT 10,160,"3",0,1,1,"F+G="+H$
PRINT 1
EOP
DEMO

```

● VAL ()

该函数将数字字符串转为数字型态

程序语法

```
VAL("numeric character")
```

参数	说明
numeric character	"0~9", ". "

范例

```

DOWNLOAD "DEMO.BAS"
SIZE 4,1
GAP 0,0
DIRECTION 1
A$="ABCDEFGHJKLMNOPQRSTUVWXYZ"
F$="100"
G$="500"
CLS
H=VAL(F$)+VAL(G$)
I$=STR$(H)

```

```

TEXT 10,10,"3",0,1,1,A$
TEXT 10,60,"3",0,1,1,"F="+F$
TEXT 10,110,"3",0,1,1,"G="+G$
TEXT 10,160,"3",0,1,1,"F+G="+I$
PRINT 1
EOP
DEMO

```

● BEEP

该指令为控制蜂鸣器收到该指令时会发出一声响

程序语法

```
BEEP
```

范例

```

DOWNLOAD "DEMO.BAS"
SIZE 4,4
GAP 0,0
DIRECTION 1
BEEP
INPUT "Text1 =",TEXT1$
CLS
TEXT 100,100,"3",0,1,1,TEXT1$
PRINT 1
EOP

```

● INPUTFILE

该指令用于接收串口文件数据。主要用来接收电子称数据

程序语法

```
INPUTFILE file handle
```

参数	说明
----	----

范例

```
DOWNLOAD F,"INPUTFILE.BAS"
SIZE 60 mm,40 mm
GAP 2 mm,0 mm
:START
CLS
INPUTFILE 0
TEXT 10,20,"3",0,1,1,"20"+STR$(YEAR)+"-"+STR$(MONTH)+"
"+STR$(DATE)+"
"+STR$(HOUR)+":"+STR$(MINUTE)+":"+STR$(SECOND)
READ 0,T1$
A$=RIGHT$(T1$,7)
TEXT 20,60,"3",0,1,1,"T1= "+A$
READ 0,G$
B$=RIGHT$(G$,7)
TEXT 20,100,"3",0,1,1,"G= "+B$
READ 0,T$
C$=RIGHT$(T$,7)
TEXT 20,140,"3",0,1,1,"T= "+C$
READ 0,N$
D$=RIGHT$(N$,7)
TEXT 20,180,"3",0,1,1,"N= "+D$
PRINT 1,1
ADJUST
GOTO START
EOP
```

● ADJUST

该指令用于循环打印命令时，使标签与撕纸口位置对齐，方便撕纸

程序语法

ADJUST

范例

DOWNLOAD F,"ADJUST.BAS"

SIZE 60 mm,40 mm

GAP 2 mm,0 mm

:START

CLS

INPUTFILE 0

TEXT 20,20,"3",0,1,1,"Hello"

PRINT 1,1

ADJUST

GOTO START

EOP

打印机外围功能设定指令

● SET COUNTER

设定计数器及增量，该指令不支持数学表达式

程序语法

SET COUNTER @n step

参数	说明
@	n: 计数器号码，打印机可使用 50 组计数 (@0-@49)
step	计数器跳号的增量，可为正或负数 -999999999<=step<=999999999

范例

```
DOWNLOAD "DEMO13.BAS"
SIZE 3,4
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET PEEL OFF
SET COUNTER @1 1
SET COUNTER @2 5
CLS
@1="00001 "
@2="TSC00001"
TEXT 50,50, "3",0,1,1,@1
BARCODE 50,500,"39",48,1,0,2,4,@2
```


PRINT 1

EOP

● SET KEY1, SET KEY2

该指令用来起动/关闭 KEY1, KEY2 的预设功能。

程序语法

SET KYE1 ON /OFF

SET KEY2 ON /OFF

参数	说明
ON	开启按键
OFF	关闭按键
KEY1	预设进纸功能
KEY2	预设暂停功能

注：关闭电源时该设定值仍会储存在打印机内。

范例

DOWNLOAD "DEMO17.BAS"

SIZE 3,4

GAP 0,0

DENSITY 8

SPEED 3

DIRECTION 0

REFERENCE 0,0

SET PEEL OFF

SET KEY1 OFF

CLS

:START

A=GETKEY()

IF A=0 THEN GOTO PAUSEB

IF A=1 THEN GOTO FEEDB

:PAUSEB

```

CLS
TEXT 50,10,"4",0,1,1,"PAUSE key is pressed!"
PRINT 1
GOTO START
:FEEDB
CLS
TEXT 50,10,"4",0,1,1,"FEED key is pressed!"
PRINT 1
GOTO START
EOP

```

● SET PEEL

该指令用来启动/关闭剥离模式，默认值为关闭

程序语法

```
SET PEEL ON/OFF
```

参数	说明
ON	启动剥离模式
OFF	关闭剥离模式

范例

```

REM SELF-PEELING FUNCTION ON
SIZE 3,4
GAP 0,0
DENSITY 8
SPEED 3
DIRECTION 0
REFERENCE 0,0
SET PEEL ON
SET KEY1 OFF
CLS
TEXT 50,100,"3",0,1,1,"SELF-PEELING FUNCTION TEST"

```

PRINT

● SET TEAR & SET STRIPPER

此命令是用来启用/禁用撕纸位置走到撕纸处，此设置关掉电源后将保存在打印机内

程序语法

SET TEAR ON/OFF

SET STRIPPER ON/OFF

参数	说明
ON	启用撕纸位置走到撕纸处
OFF	禁用撕纸位置走到撕纸处，命令在起始位置有效

范例

```
REM ***TEAE FUNCTION ON***  
SIZE 3,3  
GAP 0.08,0  
DENSITY 8  
SPEED 4  
DIRECTION 0  
REFERENCE 0,0  
SET PEEL OFF  
SET TEAR ON  
CLS  
TEXT 50,100,"3",0,1,1,"TEAR FUNCTION TEST"  
PRINT 1
```

● SET HEAD

此设置用于启用/禁用打印头合盖传感器。如果禁用合盖传感器，打印机头被打开时，将不会传回错误信息。此设置将保存在打印机内存。

程序语法

SET HEAD ON/OFF

参数	说明
ON	启用打印头合盖传感器
OFF	禁用打印头合盖传感器

范例

SET HEAD ON

SET HEAD OFF

● SET PRINTKEY

此命令将打印一个标签并走到下一个标签的间隙到撕纸位置处,按下 FEED 按键,打印下一个标签或多份的标签。如果标签内容包括串行文字或条形码,它将改变序号,此设置将保存在打印机内存

程序语法

SET PRINTKEY OFF/ON/AUTO/<num>

参数	说明
OFF	关闭此功能
ON	开启此功能
AUTO	自动开启此功能
<num>	按 FEED 键来按多少下

范例

SIZE 3,2,5

GAP 0.12,0

SET PRINTKEY ON

SET COUNTER @0 1

@0="0001"

CLS

```
TEXT 10,10,"5",0,1,1,@0  
PRINT 1
```

● SET REPRINT

此命令将禁用/启用标签机在无纸或开盖错误发生后，上纸或合盖后重新打印一次标签内容

程序语法

```
SET REPRINT OFF/ON
```

参数	说明
OFF	禁止此功能
ON	启用此功能

范例

```
SET REPRINT ON
```

● PEEL

此命令是用来获取纸剥离传感器。其属性是只读

程序语法

```
PEEL
```

参数	说明
0	当没有纸在纸剥离传感器上方时返回值
1	当有纸在纸剥离传感器上方时返回值

范例

```
DOWNLOAD "DEMO.BAS"  
SIZE 4,4  
GAP 0,0  
DENSITY 8
```

```

SPEED 3
DIRECTION 0
REFERENCE 0,0
SET PEEL OFF
SET LED1 OFF
CLS
IF PEEL=1 THEN LED1=1
EOP

```

● KEY1, KEY2

此命令用来读取打印机按键的状态

程序语法

```
KEYm = n
```

参数	返回值
KEY1 (PAUSE)	0:released 1:pressed
KEY2 (FEED)	0:released 1:pressed

范例

```

DOWNLOAD "DEMO.BAS"
SIZE 3,1
GAP 0,0
SPEED 4
DENSITY 8
DIRECTION 1
REFERENCE 0,0
SET KEY1 OFF
:START
IF KEY1=1 THEN
CLS

```

```
TEXT 100,10,"3",0,1,1,"KEY FUNCTION TEST"
PRINT 1,1
GOTO START
EOP
DEMO
```

● SET RIBBON

设定开启/关闭碳带感应器，即切换热转式/热感印式打印。通常打印机于开启电源时，碳带感应器即会自动检测打印机是否已装上碳带，并藉此决定使用热感式或热转式打印。此项设定并不会存于打印机中。此方法仅适用于热转式机器。

程序语法

SET RIBBON ON/OFF

参数	说明
ON	热转式打印
OFF	热感式打印

● SET CUTTER

此命令用于设置切刀状态，关闭打印机电源后，该设置将会被存储在打印机内存中。

程序语法

SET CUTTER OFF/BATCH/pieces

参数	说明
OFF	关闭切刀功能
BATCH	在 PRINT 命令结束后切纸
pieces	0-65535，用于设置每几个标签进行切纸

范例

输出结果：关闭切刀功能

```
SIZE 3,3
GAP 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 50,50,"3",0,1,1,"SET CUTTER OFF"
PRINT 3
```

输出结果：在 6 个标签打印完毕后切纸

```
SET CUTTER BATCH
CLS
TEXT 50,50,"3",0,1,1,"SET CUTTER BATCH"
PRINT 3,2
```

输出结果：在每 1 个标签打印完毕后切纸

```
SET CUTTER 1
CLS
TEXT 50,50,"3",0,1,1,"SET CUTTER 1"
PRINT 3,2
```

● SET RESPONSE

此指令用于设置打印机自动返回状态

程序语法

```
SET RESPONSE ["Job ID",] ON/OFF/BATCH
```

参数	说明
["Job ID"]	可选项，默认为空
ON	打开自动返回状态功能，每打印一张返回一次
OFF	关闭自动返回状态功能
BATCH	打开自动返回状态功能，打印完毕后返回一次

返回值

{Status,#####,ID}

参数	返回值
Status	参考<ESC>!/?指令（16 进制）
#####	00001-99999

范例

SET RESPONSE ON
SIZE 4,2
GAP 0,0
PRINT 3

输出结果：（返回值第一个参数为 16 进制 0x00，其余部分均为字符串类型）

{00,00001} {00,00002} {00,00003}

SET RESPONSE "ID1",ON
SIZE 4,2
GAP 0,0
PRINT 3,2

输出结果：

{00,00001, ID1} {00,00002, ID1} {00,00003, ID1} {00,00004, ID1} {00,00005, ID1}
{00,00006, ID1}

SET RESPONSE "CCCC",BATCH
SIZE 4,2
GAP 0,0
PRINT 3,2

输出结果：

{00,00006,CCCC}

打印机全局变量

● @LABEL

该变量记录打印机已打印标签张数，当打印机断电后重启，计数重置。

程序语法

Write attribute: @LABEL=n

Read attribute: A=@LABEL

参数	说明
n	打印标签张数 $0 < n < 65535$

范例

```
DOWNLOAD "DEMO20.BAS"  
SIZE 3,4  
GAP 0,0  
DENSITY 8  
SPEED 3  
DIRECTION 0  
REFERENCE 0,0  
SET PEEL ON  
SET KEY1 OFF  
SET DEBUG LABEL  
SET RIBBON OFF  
SET COM1 96,N,8,1  
CLS  
IF @LABEL=100 THEN @LABEL=0 ELSE  
TEXT 100,100,"3",0,1,1,STR$(@LABEL)  
PRINT 1
```

EOP

● YEAR

这个变量通过 RTC (Real Time Clock) 读写年份数据。四位数年份格式由 RTC 支持

程序语法

Write attribute: YEAR=02

Read attribute: A=YEAR

Range: 00-50=2000-2050;51-99=1951-1999

范例

```
DOWNLOAD "Set Year.BAS"
REM *****SetYear Parameter to RTC*****
YEAR=05
EOP
SetYear
DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
REM *****Read YEAR parameter form RTC*****
YEAR$=STR$(YEAR)
Y=YEAR
REM *****Print*****
TEXT 10,10,"5",0,1,1,"YEAR1="+YEAR$
TEXT 10,110,"5",0,1,1,"YEAR2="+STR$(Y)
```

```
TEXT 10,210,"5",0,1,1,"YEAR3="+STR$(YEAR)
PRINT 1
EOP
DEMO
```

● MONTH

这个变量通过 RTC（Real Time Clock）读写月份数据。两位数（01-12）月份格式由 RTC 支持

程序语法

```
Write attribute: MONTH=01
Read attribute: A=MONTH
Range: 01-12
```

范例

```
DOWNLOAD "SetMonth.BAS"
MONTH=05
EOP
SetMonth
DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
MONTH$=STR$(MONTH)
M=MONTH
REM *****Print*****
TEXT 10,10,"5",0,1,1,"MONTH1="+MONTH$
```

```
TEXT 10,110,"5",0,1,1,"MONTH2="+STR$(M)
TEXT 10,210,"5",0,1,1,"MONTH3="+STR$(MONTH)
PRINT 1
EOP
DEMO
```

● DATE

这个变量通过 RTC（Real Time Clock）读写日期数据。两位数（01-31）日期格式由 RTC 支持

程序语法

```
Write attribute: DATE=12
Read attribute: A=DATE
Range: 01-31
```

范例

```
DOWNLOAD "SetDate.BAS"
REM *****Set Date Parameter to RTC*****
DATE=30
EOP
SetDate
DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
REM *****Read Date parameter form RTC*****
DATE$=STR$(DATE)
```

```

D=DATE
REM *****Print*****
TEXT 10,10,"5",0,1,1,"DATE1="+DATE$
TEXT 10,110,"5",0,1,1,"DATE2="+STR$(D)
TEXT 10,210,"5",0,1,1,"DATE3="+STR$(DATE)
PRINT 1
EOP
DEMO

```

● WEEK

这个变量通过 RTC (Real Time Clock) 读写星期数据，由一位数 (1-7) 表示
程序语法

```

Write attribute: WEEK=3
Read attribute: A=WEEK
Range: 1(Sunday)-7(Saturday)

```

范例

```

DOWNLOAD "SetWeek.BAS"
REM *****Set Week Parameter to RTC*****
WEEK=6
EOP
SetWeek
DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS

```

```

REM *****Read Week parameter form RTC*****
WEEK$=STR$(WEEK)
W=WEEK
REM *****Print*****
TEXT 10,10,"5",0,1,1,"WEEK1="+WEEK$
TEXT 10,110,"5",0,1,1,"WEEK2="+STR$(W)
TEXT 10,210,"5",0,1,1,"WEEK3="+STR$(WEEK)
PRINT 1
EOP
DEMO

```

● HOUR

这个变量通过 RTC（Real Time Clock）读写时钟数据。24 小时计时法（00-23）由 RTC 支持

程序语法

```

Write attribute: HOUR=12
Read attribute: A=HOUR
Range: 00-23

```

范例

```

DOWNLOAD "SetHour.BAS"
REM *****Set Hour Parameter to RTC*****
HOUR=11
EOP
SetHour
DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0

```

```

SET CUTTER OFF
SET PEEL OFF
CLS
REM *****Read Hour parameter form RTC*****
HOUR$=STR$(HOUR)
H=HOUR
REM *****Print*****
TEXT 10,10,"5",0,1,1,"HOUR1="+HOUR$
TEXT 10,110,"5",0,1,1,"HOUR2="+STR$(H)
TEXT 10,210,"5",0,1,1,"HOUR3="+STR$(HOUR)
PRINT 1
EOP
DEMO

```

● MINUTE

这个变量通过 RTC（Real Time Clock）读写分钟数据。双位数（00-59）分钟格式由 RTC 支持

程序语法

```

Write attribute: MINUTE=12
Read attribute: A=MINUTE
Range: 00-59

```

范例

```

DOWNLOAD "SetMinute.BAS"
REM *****Set Minute Parameter to RTC*****
MINUTE=59
EOP
SetMinute
DOWNLOAD "DEMO.BAS"
SIZE 3,3
GAP 0.08,0
DENSITY 8

```



```

SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
REM *****Read Minute parameter form RTC*****
MINUTE$=STR$(MINUTE)
MIN=MINUTE
REM *****Print*****
TEXT 10,10,"5",0,1,1,"MINUTE1="+MINUTE$
TEXT 10,110,"5",0,1,1,"MINUTE2="+STR$(MIN)
TEXT 10,210,"5",0,1,1,"MINUTE3="+STR$(MINUTE)
PRINT 1
EOP
DEMO

```

● SECOND

这个变量通过 RTC（Real Time Clock）读写秒钟数据。双位数（00-59）秒钟格式由 RTC 支持

程序语法

Write attribute: SECOND=12

Read attribute: A=SECOND

Range: 00-59

范例

```

DOWNLOAD "SetSecond.BAS"
REM *****Set Second Parameter to RTC*****
SECOND=59
EOP
SetSecond
DOWNLOAD "DEMO.BAS"

```

```

SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 4
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
REM *****Read Second parameter form RTC*****
SECOND$=STR$(SECOND)
SEC=SECOND
REM *****Print*****
TEXT 10,10,"5",0,1,1,"SECOND1="+SECOND$
TEXT 10,110,"5",0,1,1,"SECOND2="+STR$(SEC)
TEXT 10,210,"5",0,1,1,"SECOND3="+STR$(SECOND)
PRINT 1
EOP
DEMO

```

● @YEAR

这个变量通过 RTC（Real Time Clock）读写年份数据。两位数年份的格式由 RTC 支持

@YEAR 全局变量可以直接访问，而无需使用 BASIC 语言功能

程序语法

Write attribute: @YEAR="01"

Read attribute: @YEAR

Range: 00-99

范例

```

REM *****Set @YEAR*****
@YEAR="05"

```

```
REM *****Print*****  
SIZE 3,3  
GAP 0.08,0  
DENSITY 8  
SPEED 6  
DIRECTION 0  
REFERENCE 0,0  
SET CUTTER OFF  
SET PEEL OFF  
CLS  
TEXT 10,10, "5",0,1,1, "@YEAR"  
TEXT 310,10, "5",0,1,1,@YEAR  
PRINT 1
```

● @MONTH

这个变量通过 RTC (Real Time Clock) 读写月份数据。双位数 (01-12) 月份格式由 RTC 支持。

@MONTH 全局变量可以直接访问，而无需使用 BASIC 语言功能。

程序语法

```
Write attribute: @MONTH="01"  
Read attribute: @MONTH  
Range: 01-12
```

范例

```
REM *****Set @MONTH*****  
@MONTH="12"  
REM *****Print*****  
SIZE 3,3  
GAP 0.08,0  
DENSITY 8  
SPEED 6  
DIRECTION 0
```

```
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,"5",0,1,1,"@MONTH"
TEXT 310,10,"5",0,1,1,@MONTH
PRINT 1
```

● @DATE

这个变量通过 RTC (Real Time Clock) 读写日期数据。双位数 (01-31) 日期格式是由 RTC 支持。

@DATE 全局变量可以直接访问，而无需使用 BASIC 语言功能。

程序语法

```
Write attribute: @DATE="12"
Read attribute: @DATE
Range: 01-31
```

范例

```
REM *****Set @DATE*****
@DATE="31"
REM *****Print*****
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,"5",0,1,1,"@DATE"
TEXT 310,10,"5",0,1,1,@DATE
```

PRINT 1

● @WEEK & @DAY

这个变量通过 RTC (Real Time Clock) 读写星期数据，由一位数 (1-7) 表示 @WEEK 全局变量可以直接访问，而无需使用 BASIC 语言功能。

程序语法

Write attribute: @WEEK="3"

Read attribute: @WEEK

Range: 1(Sunday)-7(Saturday)

范例

```
REM *****Set @WEEK*****
@WEEK="5"
REM *****Print*****
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,"5",0,1,1,"@WEEK"
TEXT 310,10,"5",0,1,1,@WEEK
PRINT 1
```

● @HOUR

这个变量通过 RTC (Real Time Clock) 读写时钟数据。24 小时计时法 (00-23) 由 RTC 支持。

@HOUR 全局变量可以直接访问，而无需使用 BASIC 语言功能。

程序语法

Write attribute: @HOUR ="12"

Read attribute: @HOUR

Range: 00-23

范例

```
REM *****Set @HOUR*****
@HOUR="23 "
REM *****Print*****
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10, "5",0,1,1,"@HOUR"
TEXT 310,10, "5",0,1,1,@HOUR
PRINT 1
```

● @MINUTE

这个变量通过 RTC (Real Time Clock) 读写分钟数据。双位数 (00-59) 分钟格式由 RTC 支持

@MINUTE 全局变量可以直接访问，而无需使用 BASIC 语言功能。

程序语法

Write attribute: @MINUTE ="12"

Read attribute: @MINUTE

Range: 00-59

范例

```
REM *****Set @MINUTE*****
```

```

@MINUTE="59"
REM *****Print*****
SIZE 3,3
GAP 0.08,0
DENSITY 8
SPEED 6
DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,"5",0,1,1,"@MINUTE"
TEXT 310,10,"5",0,1,1,@MINUTE
PRINT 1

```

● @SECOND

这个变量通过 RTC（Real Time Clock）读写秒钟数据。双位数（00-59）秒钟格式由 RTC 支持

@SECOND 全局变量可以直接访问，而无需使用 BASIC 语言功能。

程序语法

```

Write attribute: @SECOND="12"
Read attribute: @SECOND
Range: 00-59

```

范例

```

REM *****Set @SECOND*****
@SECOND="59"
REM *****Print*****
SIZE 3,3
GAP 0,0
DENSITY 8
SPEED 6

```

DIRECTION 0
REFERENCE 0,0
SET CUTTER OFF
SET PEEL OFF
CLS
TEXT 10,10,"5",0,1,1,"@SECOND"
TEXT 310,10,"5",0,1,1,@SECOND
PRINT 1