

# **CPCL 标签打印机 中文编程手册**

# 使用指南

## ● 文档字体规则

符号	描述
{ }	必填项
[ ]	可选项
( )	缩写命令
< >	文字项（字符串）
\r\n	回车和换行字符，用于每行结尾
空格字符用于分隔命令行中的各个字段	
文档中的默认单位为 dot，与毫米换算：1 mm = 8 dots	

## ● 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。

**! 0 200 200 210 1\r\n**

**TEXT 4 0 30 40 Hello World\r\n**

**PRINT\r\n**

一张标签通常包含三个部分，即标签规格设定（蓝色部分）、标签内容设定（绿色部分）和执行打印走纸指令（红色部分）。

标签规格设定包括标签尺寸、打印宽度

标签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，下文代码范例中不再赘述。

# 打印机命令

## ● 标签初始化指令

标签文件通常以“!”字符作为开头，后接“x”偏置参数、“x”和“y”轴分辨率、标签长度以及要打印的标签数量。包含这些参数的行称为命令起始行。

任何情况下，标签文件都是以命令起始行开头，以“PRINT”命令结尾。用于构建具体标签的命令置于这两项命令之间。

### 指令语法

`<!> {offset} <200> <200> {height} {qty}`

参数	说明
<!>	使用“!”作为控制会话的起始字符
{offset}	标签横向偏移量
<200>	横向 DPI
<200>	纵向 DPI
{height}	标签最大高度
{qty}	标签数量

标签最大高度的计算方法是，先测出从第 1 个黑条（或标签间隙）底部到下一个黑条（或标签间隙）顶部之间的距离。然后从中减去 1/16 英寸（1.5 毫米），所得结果即最大高度。（以点为单位时：对于 203 DPI 打印机，减去 12 点；对于 306 DPI 打印机，减去 18 点）

### 代码范例

```
! 0 200 200 210 1
TEXT 4 0 30 40 Hello World
PRINT
```

## ● PRINT 指令

PRINT 命令作为整个命令集的结束命令，将会启动文件打印。在任何情况下（行式打印模式除外），这项命令都必须是最后一条命令。执行 PRINT 命令时，打印机将从控制会话中退出。确保使用回车和换行字符结束此项及所有命令。

指令语法：

{command}

参数	说明
{command}	PRINT

## ● 使用注释

注释可以添加在命令会话第一行和“PRINT”命令之间。

在文件中添加注释时，需要将“;”字符置入第一列，以此作为注释行的起始部分。“;”字符与行末尾的所有其他文本都将被忽略。

指令语法

无

代码范例

```
! 0 200 200 55 1
; Center justify text
CENTER
; Print the words ' A COMMENT'
TEXT 5 1 0 5 A COMMENT
; Print the label and go to top of next form
PRINT
```

# 文本指令

## ● TEXT 命令

TEXT 命令用于在标签上添加文本。这项命令及其各衍生命令可以控制使用的具体字体号和大小、标签上文本的位置以及文本的方向。标准常驻字体能够以 90 度的增量旋转，如下例所示。

指令语法：

```
{command} {font} {size} {x} {y} {data}
```

{command}	指令效果
TEXT (或 T)	横向打印文本。
VTEXT (或 VT)	逆时针旋转 90 度，纵向打印文本。
TEXT90 (或 T90)	(同 VTEXT。)
TEXT180 (或 T180)	逆时针旋转 80 度，反转打印文本。
TEXT270 (或 T270)	逆时针旋转 270 度，纵向打印文本。

参数	说明
{font}	字体名称/编号
{size}	忽略该参数，请输入任意数字
{x}	横向起始位置
{y}	纵向起始位置
{data}	要打印的文本

{font}	英文字体	中文字体
0	12*24	24*24 简体中文 GB18030
1	9*17	24*24 简体中文 GB18030
2	12*24	24*24 简体中文 GB18030
3	10*20	20*20 简体中文 GBK
4	16*32	32*32 简体中文 GBK

5	9*17	24*24 简体中文 GB18030
6	12*24	
7	12*24	24*24 简体中文 GB18030
8	12*24	24*24 简体中文 GB18030
10	24*48	48*48 简体中文 GBK
11	8*16	24*24 简体中文 GB18030
13	12*24	24*24 繁体中文 BIG5
20	8*16	16*16 简体中文 GB18030
24	12*24	24*24 简体中文 GB18030
41	8*12	
42	12*20	
43	16*24	
44	24*32	
45	32*48	
46	14*19	
47	21*27	
48	14*25	
49	28*56	
55	8*16	16*16 简体中文 GB18030

## 代码范例

```
! 0 200 200 210 1
TEXT 0 0 200 100 TEXT
TEXT90 0 0 200 100 T90
TEXT180 0 0 200 100 T180
TEXT270 0 0 200 100 T270
PRINT
```

## ● TEXT 串联命令 (CONCAT)

使用文本串联，可以为字符串分配不同的字符样式，在同一文本行上使用统一间距进行打印。

指令语法：

```
{command} {x} {y}  
{font} {size} {offset} {data}  
.....  
{font} {size} {offset} {data}  
<ENDCONCAT>
```

{command}	指令效果
CONCAT	横向串联

参数	说明
{x}	横向起始位置
{y}	纵向起始位置
{font}	字体名称/编号
{size}	忽略该参数，请输入任意数字
{offset}	文本相对起始位置的偏置单位值
{data}	要打印的文本
<ENDCONCAT>	终止文本串联

代码范例

```
! 0 200 200 210 1  
CONCAT 75 75  
2 2 5 $  
3 3 0 12  
4 2 5 34  
ENDCONCAT  
PRINT
```

## ● SETMAG 命令

SETMAG 命令可将常驻字体放大指定的放大倍数

指令语法：

{command} {w} {h}

参数	说明
{command}	SETMAG
{w}	宽度放大倍数，有效放大倍数为 1 到 16
{h}	高度放大倍数，有效放大倍数为 1 到 16

SETMAG 命令在标签打印后仍保持有效。这意味着要打印的下一标签将使用最近设置的 SETMAG 值。要取消 SETMAG 值并使打印机可以使用默认字体大小，请使用 SETMAG 命令，且放大倍数为 0,0。

#### 代码范例

```
! 0 200 200 210 1
CENTER
SETMAG 1 1
TEXT 0 0 0 10 Font 0-0 at SETMAG 1 1
SETMAG 1 2
TEXT 0 0 0 40 Font 0-0 at SETMAG 1 2
SETMAG 2 1
TEXT 0 0 0 80 Font 0-0 at SETMAG 2 1
SETMAG 2 2
TEXT 0 0 0 110 Font 0-0 at SETMAG 2 2
SETMAG 2 4
TEXT 0 0 0 145 Font 0-0 at SETMAG 2 4
; Restore default font sizes
SETMAG 0 0
PRINT
```

## ● SETBLOD 命令

SETBLOD 命令可将常驻字体加粗

指令语法：

{command} {value}



参数	说明
{command}	SETBLOD
{value}	0, 不加粗 1, 加粗

SETBLOD 命令在设定后保持有效。这意味着要打印的下一部分标签内容将使用最近设置的 SETBLOD 值。要取消加粗请发送 SETBLOD 0

代码范例

```
! 0 200 200 210 1
CENTER
SETBOLD 1
TEXT 1 0 0 10 BOLD FONT
SETBOLD 0
TEXT 1 0 0 40 NORMAL FONT
PRINT
```

# 条码指令

## ● BARCODE 命令

BARCODE 命令能够以指定的宽度和高度纵向和横向打印条码。建议以理想宽窄比和窄条宽度进行打印。

指令语法：

`{command} {type} {width} {ratio} {height} {x} {y} {data}`

{command}	指令效果
BARCODE (或 B)	横向打印条码
VBARCODE (或 VB)	纵向打印条码

{tpe}	条码种类	理想宽窄比	理想窄点宽度
UPCA	UPC-A	2:1	2
UPCE	UPC-E	2:1	2
EAN13	EAN/JAN-13	2:1	2
EAN8	EAN/JAN-8	2:1	2
39	Code39	2.5:1	2
93	Code93/Ext. 93	1.5:1	1
128	Code128	N/A	2
CODABAR	Codabar	2.5:1	2

{ratio}	宽窄比
0	1.5:1
1	2.0:1
2	2.5:1
3	3.0:1
4	3.5:1
20	2.0:1

21	2. 1:1
22	2. 2:1
23	2. 3:1
24	2. 4:1
25	2. 5:1
26	2. 6:1
27	2. 7:1
28	2. 8:1
29	2. 9:1
30	3. 0:1

参数	说明
{width}	窄条的单位宽度
{ratio}	宽条与窄条的比率
{height}	条码的单位高度
{x}	横向起始位置
{y}	纵向起始位置
{data}	条码数据

#### 代码范例

```
! 0 200 200 210 1
  BARCODE 128 1 1 50 150 10 HORIZ.
  TEXT 7 0 210 70 HORIZ.
  VBARCODE 128 1 1 50 10 200 VERT.
  VTEXT 7 0 70 140 VERT.
PRINT
```

## ● **BARCODE-TEXT 命令**

BARCODE-TEXT 命令用于通过创建条码时所用的相同数据来标记条码。这项命令避免了使用单独文本命令注释条码的必要。文本位于条码下方的中间位置。

使用 BARCODE-TEXT OFF（或 BT OFF）可以禁用它。

指令语法：

{command} {font number} {font size} {offset}

参数	说明
{command}	BARCODE-TEXT（或 BT）
{font number}	注释条码时要使用的字体号
{font size}	忽略该参数，请输入任意数字
{offset}	文本距离条码的单位偏移量

代码范例

! 0 200 200 400 1

CENTER

BARCODE-TEXT 7 0 5

BARCODE 128 1 1 50 0 20 123456789

VBARCODE 128 1 1 50 40 400 112233445

BARCODE-TEXT OFF

PRINT

## ● QR Code

指令语法：

{command} {type} {x} {y} [M n] [U n]

{data}

<ENDQR>

{command}	指令效果
BARCODE（或 B）	横向打印条码
VBARCODE（或 VB）	纵向打印条码

参数	说明
{type}	QR

{x}	横向起始位置
{y}	纵向起始位置
[M n]	QR Code 规范编号,1 或 2, 推荐为 2
[U n]	模块的单位宽度/单位高度
	1-32, 默认为 6
<ENDQR>	终止 QR Code

{data}除了包含实际的输入数据字符串外,还包含一些模式选择符号。输入数据类型可以由打印机软件自动识别,也可以通过手动方式设置。模式选择符号和实际数据之间有一个分隔符(逗号)。

{date}格式如下:

<纠错等级><掩码><输入模式>,<所需生成二维码的数据>

{data}	含义及参数范围
<纠错等级>	H - 极高可靠性级别 (H 级) Q - 高可靠性级别 (Q 级) M - 标准级别 (M 级) L - 高密度级别 (L 级)
<掩码>	省略 - 软件自动选择掩码 0 至 7 - 使用带有相应编号 (0 至 7) 的掩码 8 - 无掩码
<输入模式>	A - 自动 M - 手动 (不支持)
<所需生成二维码的数据>	

代码范例

! 0 200 200 500 1

CENTER

B QR 10 100 M 2 U 10

MA,QR code ABC123

ENDQR

T 4 0 10 400 QR code ABC123

PRINT

# 图形指令

## ● BOX 命令

用户可以使用 BOX 命令生成具有指定线条宽度的矩形。

指令语法：

`{command} {x0} {y0} {x1} {y1} {width}`

参数	说明
{command}	BOX
{x0}	左上角的 X 坐标
{y0}	左上角的 Y 坐标
{x1}	右下角的 X 坐标
{y1}	右下角的 Y 坐标
{width}	形成矩形框的线条的单位宽度

代码范例

```
! 0 200 200 210 1
BOX 0 0 200 200 1
PRINT
```

## ● LINE 命令

使用 LINE 命令可以绘制任何长度、宽度和角度方向的线条。

指令语法：

`{command} {x0} {y0} {x1} {y1} {width}`

参数	说明
{command}	LINE （或 L）
{x0}	起始点的 X 坐标

{y0}	起始点的 Y 坐标
{x1}	终止点的 X 坐标
{y1}	终止点的 Y 坐标
{width}	线条的单位宽度

#### 代码范例

```
! 0 200 200 210 1
LINE 0 0 200 0 1
LINE 0 0 200 200 2
LINE 0 0 0 200 3
PRINT
```

## ● INVERSE-LINE 命令

INVERSE-LINE 命令的语法与 LINE 命令相同。位于 INVERSE-LINE 命令所定义区域内的以前创建的对象黑色区域将重绘为白色，白色区域将重绘为黑色。这些对象可以包括文本、条码和/或图形。INVERSE-LINE 对在其之后创建的对象不起作用，即使这些对象位于该命令的覆盖区域内也是如此。

指令语法：

```
{command} {x0} {y0} {x1} {y1} {width}
```

参数	说明
{command}	INVERSE-LINE （或 IL）
{x0}	起始点的 X 坐标
{y0}	起始点的 Y 坐标
{x1}	终止点的 X 坐标
{y1}	同 y0
{width}	反色区域高度

#### 代码范例

```
! 0 200 200 210 1
T 4 2 30 20 $123.45
```



```
T 4 2 30 70 $678.90
INVERSE-LINE 25 40 350 40 90
T 4 2 30 120 $432.10
PRINT
```

## ● GRAPHICS 命令

可以使用图形命令打印位映射图形。扩展图形数据使用 ASCII 十六进制字符来表示（参见示例）。通过对十六进制数据的等效二进制字符使用 COMPRESSED-GRAPHICS 命令，可以将数据大小减半。

如果使用 CG，对于每 8 位图形数据，将会发送一个 8 位字符。如果使用 EG，将使用两个字符（16 位）来传输 8 位图形数据，因此 EG 的效率会减半。但是由于该数据是字符数据，因此比二进制数据更容易处理和传输。

如果传输一个 byte (0xFF), CG 指令下直接发送 0xFF，而 EG 下发送字符串 FF，即 16 进制 0x46 0x46

指令语法：

```
{command} {width} {height} {x} {y} {data}
```

{command}	说明
EXPANDED-GRAPHICS（或 EG）	横向打印扩展图形
COMPRESSED-GRAPHICS（或 CG）	横向打印压缩图形

参数	说明
{width}	图像的宽度（以字节为单位）
{height}	图像的高度（以点为单位）
{x}	横向起始位置
{y}	纵向起始位置
{data}	图形数据，由上至下，由左至右

代码范例

```
! 0 200 200 210 1
```

```
EG 2 16 90 45 F0F0F0F0F0F0F00F0F0F0F0F0F0F
F0F0F0F0F0F0F0F00F0F0F0F0F0F0F0F
PRINT
```

范例中位图数据结构如下图所示，宽度为 2 个 byte，高度为 16 个点

	图像的宽度 （以字节为单位）												对应16进制数据					
图像的高度	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	F0	F0
	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	F0	F0
	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	F0	F0
	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	F0	F0
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0F	0F
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0F	0F
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0F	0F
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0F	0F
	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	F0	F0
	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	F0	F0
	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	F0	F0
	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	F0	F0
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0F	0F
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0F	0F
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0F	0F
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0F	0F
	1 byte								1 byte									

# 高级指令

## ● JUSTIFICATION 命令

使用对齐命令可以控制字段的对齐方式。默认情况下,打印机将左对齐所有字段。  
对齐命令将对所有后续字段保持有效,直至指定了其他对齐命令。

指令语法 :

`{command} [end]`

参数	说明
<code>{command}</code>	从下面选择一项: CENTER: 居中对齐所有后续字段。 LEFT: 左对齐所有后续字段。 RIGHT: 右对齐所有后续字段。
<code>[end]</code>	对齐的结束点。如果未输入参数,则对于横向打印,对齐命令将使用打印头的宽度;而对于纵向打印,对齐命令将使用零(页头)

代码范例

```
! 0 200 200 210 1
CENTER 383
TEXT 4 0 0 75 C
LEFT
TEXT 4 0 0 75 L
RIGHT 383
TEXT 4 0 0 75 R
PRINT
```

## ● PAGE-WIDTH 命令

打印机假定页面宽度为打印机的完整宽度。打印会话的最大高度由页面宽度和可用打印内存决定。如果页面宽度小于打印机的完整宽度，则用户可以通过指定页面宽度来增加最大页面高度。

指令语法：

`{command} {width}`

参数	说明
<code>{command}</code>	PAGE-WIDTH（或 PW）
<code>{width}</code>	页面的单位宽度

代码范例

```
! 0 200 200 210 1
LEFT
PAGE-WIDTH 374
TEXT 4 0 0 0 测试文本 123abc
PRINT
```

```
! 0 200 200 210 1
LEFT
PAGE-WIDTH 574
TEXT 4 0 0 0 测试文本 123abc
PRINT
```

## ● SPEED 命令

此命令用于设置电机的最高速度级别。每一款打印机型号都设置了最低和最高极限速度。SPEED 命令可以在 0 到 5 的范围内选择速度级别，0 表示最低速度。为每一款打印机型号设置的最高速度仅可在理想条件下达到。电池或供电电压、材料厚度、打印黑度、是否使用贴标机、是否使用剥离器以及标签长度等诸多因素均会影响最大极限打印速度。

指令语法：

{command} {speed level}

参数	说明
{command}	SPEED
{speed level}	0-5

代码范例

```
! 0 200 200 450 1
LEFT
SPEED 4
TEXT 5 0 0 20 PRINTS AT SPEED 4
PRINT
```

## ● BEEP 命令

此命令用于指示打印机让蜂鸣器发出给定时间长度的声音。未配备蜂鸣器的打印机将忽略此命令。

指令语法：

{command} {beep\_length}

参数	说明
{command}	BEEP
{beep_length}	蜂鸣持续时间，以 1/8 秒为单位递增

代码范例

```
! 0 200 200 210 1
CENTER
TEXT 5 0 0 10 beeps for two seconds
BEEP 16
PRINT
```

# 查询指令

●     <ESC>1B 68

打印机返回一个 byte 的数据，根据返回值，判断打印机当前状态。

指令语法

16 进制：1B 68

返回值（16 进制）	打印机状态
00	正常
01	走纸或正在打印
02	缺纸
04	开盖有纸
06	开盖缺纸

返回值可转换为二进制，按位（bit）进行观察判定，0 位假，1 为真

bit	打印机状态（值为 0 或 1）
0	正在打印
1	缺纸
2	开盖
3	电量低
4	未定义
5	未定义
6	未定义
7	未定义