

打印机控制语言 编程实例集

(PCL 5)



朱 聆 编
熊可宜 审校

北京希望电脑公司

打印机控制语言编程实例集 (PCL5)

朱 聆 编
熊可宜 审校

北京希望电脑公司

前 言

PCL5(PRINTER CONTROL LANGUAGE VERSION 5)是 HP 系列打印机专用语言的杰出代表,不但适用于国内越来越流行的 HP LASERJET III 激光打印机和 HPDESKJET 500 喷墨打印机,而且成为了一种事实上的工业标准,与 EPSON 公司的 ESC PK 打印控制语言(点阵打印机), ADOBE 公司的 POSTSCRIPT 页描述语言(高档激光打印输出设备)并称三大打印控制语言。

本书为读者提供了三十多个 PCL5 程序实例,覆盖了 PCL5 中最精华的部分:字模和图形,并给出了汉字驱动程序的编制方法和外部文件的接口,旨在确实提供一些有指导意义和实用价值的知识。

作为本书作者,我真诚地希望这本书能够给广大从事软件开发的同行们以帮助,并欢迎各界朋友就本书内容进行切磋。

编 者

1993 · 北京

目 录

第一章 PCL5 语言概述	1
1.1 PCL5 命令语法	1
1.2 打印分辨率	2
1.3 基本打印命令	3
1.4 页面格式控制	4
1.5 编程实例源代码	7
第二章 字模和字模变换	65
2.1 简介	65
2.2 字模基础知识	65
2.3 PCL5 字模文件	67
2.4 下载字模集 (DOWNLOADING FONT)	77
2.5 应用程序字模设计	82
2.6 字模变换	88
2.7 编程实例源代码	95
第三章 光栅图形	210
3.1 打印矩形区域	210
3.2 光栅图形	215
3.3 扩充存储空间	220
3.4 光栅图形文件	222
3.5 光栅图形设计实例	224
3.6 编程实例源代码	226
第四章 向量图形	313
4.1 在 PCL5 语言中使用 HP-GL	313
4.2 HP-GL/2 编程要点	313
4.3 HP-GL 外部文件	320
4.4 编程实例源代码清单	320
第五章 文件格式	324
5.1 PCX 文件	324
5.2 WORDPERFECT 文件	328
5.3 编程实例源代码清单	329
第六章 汉字打印驱动程序	364
6.1 点阵字库打印程序设计	364
6.2 汉字变换	376
6.3 轮廓字库	379
附录 A PCL5 语言控制命令	384
附录 B HP-GL 语言	392
附录 C PCL5 头文件 LJMAIN.H	395
附录 D 参考文献	397

第一章 PCL5 语言概述

近年来,随着计算机技术的飞速发展,以打印机为代表的输出技术也是日新月异,先后经历了行式、点阵针打式、激光式等几个阶段。

行式打印机只能根据打印纸的宽度打印一行固定字模的字符,图象处理能力差,但文本输出速度很快,目前 IBM 的高速行式打印机仍然广泛应用于大中型机房;点阵针式打印机采用点阵字模,用打印针的点阵描绘字符和图形,通用性好,目前应用范围最为广泛,如 LQ1600K、AR3240 等几种常用打印机占国内打印机市场的 75% 以上;激光打印机采用高科技手段,以“页”为基本输出单位,代替了以往以“行”为单位的打印模式,既便于对整页内容做统一变换和修改,又具有较高的精度和速度,其中以 HP LASERJET 系列为代表的激光打印机已经在当今高档字处理和轻印刷系统中独占鳌头。

PCL5 (PRINTER CONTROL LANGUAGE VERSION 5) 是 HP 公司首先推出的一种打印控制语言,是一种解释型的命令语言,与 BASIC 语言类似,打印机对每条接收到的命令马上解释执行,而不用等所有命令全部接收。与此相对的是编译型的语言(如 C, ADOBE 公司的 POSTSCRIPT 打印页描述语言等),强调语句的模块化和完整性,只有全部命令都接收到后进行编译连接,才能完成相应的功能。

1.1 PCL5 命令语法

PCL5 语言有三种类型的命令格式:即控制码、PCL 码、HP-GL 码。

1.1.1 控制码

由 ASCII 字符表示,执行一项基本打印功能。

举例:回车(CR),
换行(LF),
送纸(FF)等。

1.1.2 PCL 码

由主计算机发出的代码序列,打印机内置的 PCL 命令解释器解释执行,一般由两个或更多的字符组成,第一个字符是 ASCII 字符“ESCAPE”(十进制数 27),其后续字符串构成“ESC 序列”表示一条打印命令。

语法结构如下:

[1] 双字符 ESC 序列:

格式: ESC X

说明: X 是 ASCII 表中十进制为 48—126 之间的任意字符。

举例:

ESC 9:

[2] ESC 参数序列:

格式: ESC X Y #Z1 #Z2 ... #Zn [data]

说明:

- X 标识符(十进制 33—47 之间的 ASCII 符号)表示此序列设置了参数。
- Y 控制属性符号,表示此序列是用来控制状态,页面,光标等的某一类(组)。
- # 数值(-32767—32767),任选项。
- Zi 参数字符(十进制 96—126)。该字符表示接收赋值的参数,任选项。
- Zn 结束符
- [data] 二进制数据,任选项。

举例:

ESC & a 4 L: 多字节PCL码,表示打印头定位于左边空为 4 列的地方。

其中 & 表示有关控制打印页格式的一系列命令。

a 表示具体命令标识。

4 表示数据值,这里单位为列。

L 表示“左边空”。

ESC & a 72 M: 表示打印头定位于右边空为 72 列的地方。

其中 &, a 各参数含义与前例相同, M 表示“右边空”。

注: ESC & a 又称为命令前缀,表示功能相近的一组命令集。

PCL5 中可以把功能相近,又连续执行的两条命令或多条命令加以组合,只写一个命令前缀,并把前几条命令的最后一个字符由大写改为小写。

例如: ESC & a 4 l 72 M 表示打印头定位于左边空为第 4 列,右边空为第 72 列的地方。

1.1.3 HP-GL 码

HP 打印机和绘图仪特有的矢量图形命令。由两个字母的助记符表示,其后的一个或多个参数表示执行这一命令的方式。

举例:

IN: 初始化

PD: 设置落笔状态

PU: 设置抬笔状态

1.2 打印分辨率

1.2.1 点(DOT, 相当于 1/300 英寸)

在图象打印时,经常需要取点为单位,其中 PCL5 基本打印分辨率为 300 DPI(又称 300 线,即每英寸 300 点);可选择的分辨率有 3 种: 150 DPI, 100 DPI, 75DPI。

这些低分辨率的每个点(DOT)是用一个 300 DPI 点组成的 $N \times N$ 点阵组成的,对于

150 DPI (N=2, 即 2 * 2 点阵), 100 DPI (N=3, 即 3 * 3 点阵), 75 DPI (N=4, 即 4 * 4 点阵)。

1.2.2 磅 (POINT, 相当于 1/72 英寸)

在字模 (FONT) 操作中, 经常用到以磅为单位, 磅是指字模的高度, 即字模的字形大小, 这样 36 磅的字符高度为半英寸。

1.2.3 十分之一磅 (DECIPOINT, 相当于 1/720 英寸)

打印机 ROM 中固化了 PCL5 命令解释程序, 在计算内部计算机打印距离时采用十分之一磅 (DECIPOINT) 为单位, 但在应用程序设计中一般不会用到它, 因为硬件会自动舍入到最近的点 (DOT) 上。

1.2.4 行 (ROW), 列 (COLUMN)

垂直方向字符行间隔, 水平方向单个字符间距

行 (ROW), 列 (COLUMN) 是两个宏观度量单位, 换行命令 (LF) 实现行位置加一, 空格符 (16 进制 20) 实现列位置加一。

1.3 基本打印命令

1.3.1 纵向定位

行距 (LINE SPACE), 即换行量, 标识两行正文之间距离, 其度量单位为 LPI (每英寸行数)。有两条命令可供使用。

[1] 标准行距

命令格式:

ESC & I # D

其中 # 代表 LPI 数值, 取值范围为 1, 2, 4, 6, 8, 12, 16, 24, 48, 缺省值为 6。

[2] 随机行距 (VMI 命令)

命令格式:

ESC & I # C

其中 # 代表以 1/48 英寸为单位的数值, 取值范围为有理数。

例如: 设置 10 DPI 换行量 (非标准 LPI 值), 可使用下列命令:

ESC & I 4.8 C 即行距为 $4.8 / 48 = 1/10$ 英寸。

注意: 行距设置要适度, 太密会引起两行字符重叠。

1.3.2 字符宽度

控制字符宽度采用 HMI 命令。

命令格式

ESC & I # H

其中 # 代表以 1/120 英寸为单位的数值, 取值范围为有理数。

注意: PCL5 中有两种字符集状态: 定宽字模和比例字模。

定宽字模集中所有组成字符宽度相同, 均为 # 值。

比例字模集中所有组成字符宽度不同, (如 "W" 要比 "I" 宽些), 这时 # 值为单个字符宽度。

PCL5 缺省值为 10 PITCH (每英寸 10 个字符)。

1.3.3 光标定位

光标 (CURSOR) 是指当前打印位置。

PCL5 坐标系取右为 X 轴正方向, 取下为 Y 轴正方向, 坐标原点为打印纸的最左端和最上端, 其右下方为打印区域。

在 PCL5 坐标系中, 有相对位置定位和绝对位置定位两组命令。

下面以应用较为广泛的相对位置定位为例:

度量单位	水平方向	竖直方向
点	ESC * p # X	ESC * p # Y
磅	ESC & a # H	ESC & a # V
行(列)	ESC & a # C	ESC & a # R

在这三种光标设置命令中, 以点为单位的最为常用。

例如:

设当前光标位置为打印坐标原点, 移动光标去(300, 900)处, 即在下方 3 英寸, 右方 1 英寸定位光标, (以 300 DPI 为单位)。

命令格式: ESC * p 300 x 400 Y

1.4 页面格式控制

1.4.1 物理页 (PHYSICAL PAGE)

真实打印纸的规格和大小。

命令格式: ESC & I # A

其中 # 的取值含义如下:

取值	规格	尺寸
1	公文纸	7 1/4" × 10 1/2"
2	信纸	8 1/2" × 11"
3	法律公文纸	8 1/2" × 14"
26	A4 纸	210 mm × 297 mm
80	公函信封	3 7/8" × 7 1/2"
81	COM10 信封	4 1/8" × 9 1/2"
90	DL 信封	110 mm × 220 mm
91	C5 信封	162 mm × 229 mm

PCL5 在执行此命令后自动检查当前纸规格并确认。

1.4.2 可打印区域(PRINTABLE AREA)

由于机械上的限制, 打印机并不能在物理纸的所有位置上打印, 其中在四个页边有部分区域(约 50 个点, 即 1/6 英寸)不可打印, 除此之外的区域称为可打印区域。

1.4.3 逻辑页(LOGICAL PAGE)

可打印区域加上页顶, 页底的不可打印区域, 它是光标定位的依据。

(1) 页顶边空 (TOP MARGIN)

缺省值为 1/2 英寸, 可用 PCL5 命令自由设置。

命令格式: ESC & I # E

其中 # 为行数, 注意这个参数与行距有关。

例如: PCL5 行距设置为 8 LPI, 则 # = 8 时代表 1 英寸的边空;

若行距设置为 6 LPI, 则 # = 8 时代表 1 1/3 英寸的边空;

一旦边空设置完毕, 再修改 LPI 值将不影响打印。

页顶边空的缺省值为 72% 到 75% VMI 长度。

(2) 页底边空 (BOTTOM MARGIN)

根据页顶边空和每页打印行数(TEXT LENGTH)而定。

命令格式: ESC & I # F

其中 # 为正文区域所占的行数。

(3) 页缝区域

本页页底边空和下一页页顶边空之间的区域。对于连续打印纸而言, 这一区域通常用于打印页号和标题。

命令格式: ESC & I # L

其中 # = 0 表示可以使用页缝区域,

= 1 表示不能使用页缝区域, 这时将走纸到下一页正文打印区域。在实际应用时, 有时需要突破边空的限制, 在正文区域之外移动光标打印。

例如:

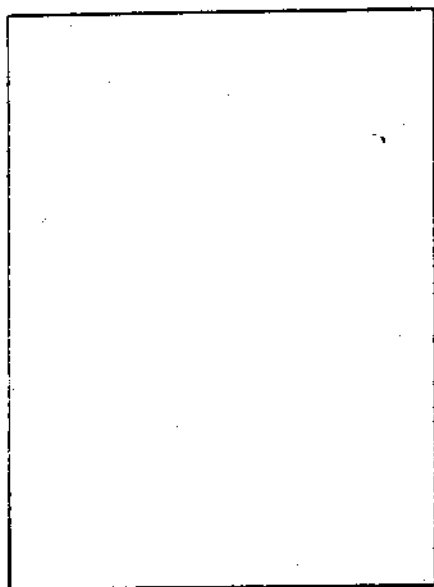
设置正文区域为最大取值, 即等于整个逻辑页大小 (设置页顶边空为 0, 允许使用页缝区域并初始化左右边空为最小取值:

ESC & I 0 I 0 L ESC 9

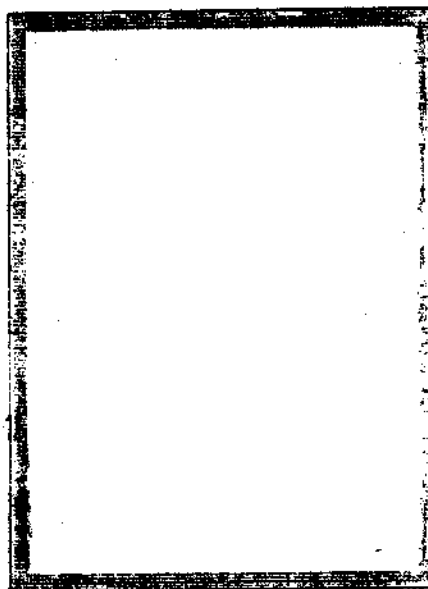
1.4.4 正文打印区域 (TEXT AREA)

可打印区域的子集, 即除去了用户自定义页边空(MARGIN)后的可打印区域。

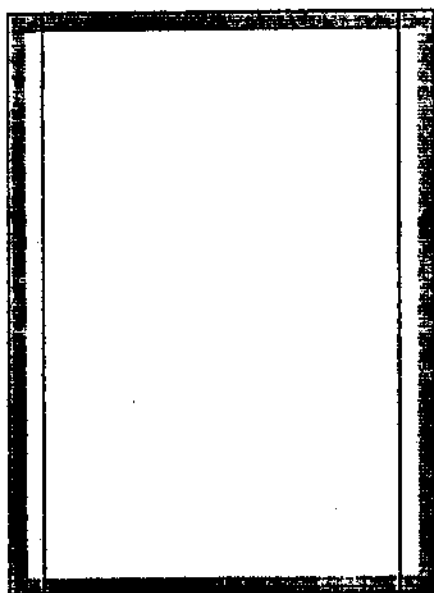
图 1.1 给出了这四种区域概念的示意图:



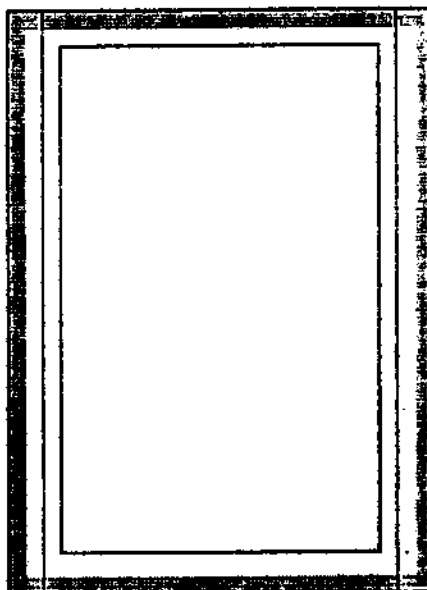
Physical Page



Unprintable Areas



Logical Page



Text Area (depending on user margins)

图 1.1 PCL5 四种页面区域示意图

1.5 编程实例源代码

PCL5 语言把打印机作为一个标准输出文件：先打开(open)之后才能向它写数据，操作完毕后要向其它文件一样关闭(close)，并冲去存储在缓冲器中的未打印的数据信息。这一段过程在具体实现上有些问题，文件打开时会受到 DOS 系统中断和数据传输错误的影响。

初始化打印机是一种良好的编程习惯，它可以清除以前工作遗留下来的打印数据，结束上一命令状态，恢复用户打印的缺省配置，等待处理下一打印命令。

命令格式：ESC E

接下来是传送打印命令和数据，并需要在页末发送一个换页字符 (FF) 结束打印页。

编程实例一：PCL5 程序模板

本例程给出了 PCL5 基本程序框架。

```
#include <stdio.h>
#include "ljmain.h"
FILE * lj;
main(int argc, char * argv[])
{
    if((lj=fopen__laserjet("PRN"))==NULL)
    {
        fputs("ERROR OPENING PRINTER", stderr);
        exit(2);
    }
    fprintf(lj, "%cE", ESC);
    /* 应用程序主体模块 */
    .....
    .....
    .....
    .....
    fputs(FormFeed, lj);
    fprintf(lj, "%cE", ESC);
    fclose(lj);
    return(1);
}
```

编程实例二：PCL5 文件打开例程

本例程用于屏蔽文件打开操作的同时可能出现中断调用。

```
#include <stdio.h>
```

```
#include "lmain.h"
```

```
#if MSDOS
```

```
/* ***** */
/*          设置二进制码到标准设备          */
/* ***** */
```

```
void set__binary__mode(FILE * fptr)
```

```
{
```

```
    int handle;
```

```
    union REGS r;
```

```
    handle = fileno(fptr);
```

```
    r.h.ah = 0x44;
```

```
    r.h.al = 0x00;
```

```
    r.x.bx = handle;
```

```
    r.x.dx = 0x00;
```

```
    int86(0x21, &r, &r);
```

```
/* 检查是否为字符设备 */
```

```
if(r.h.dl & 0x20)
```

```
{
```

```
    if(!(r.h.dl & 0x20))
```

```
    {
```

```
        r.h.ah = 0x44;
```

```
        r.h.al = 0x01;
```

```
        r.x.bx = handle;
```

```
        r.h.dh = 0x00;
```

```
        r.h.dl = 0x20;
```

```
        int86(0x21, &r, &r);
```

```
    }
```

```
}
```

```
}
```

```
#endif /* MSDOS */
```

```
/* ***** */
/*          打开标准设备文件流          */
/* ***** */
```

```

FILE * fopen__laserjet(char * filename)
{
    FILE * fp;
    fp = fopen(filename, "wb");

    #if MSDOS
    if(fp != NULL)
        set__binary__mode(fp);
    #endif
    return(fp);
}

```

编程实例三：PCL5 光标定位例程

本例程用于检验光标在横，纵两个方向的定位功能。

功能：输入为 60 行的正文，程序自动调整位置，在打印页中央输出。

```

#include <ctype.h>
#include <stdio.h>
#include <string.h>

#include "lmain. h"

FILE * fin, * fout;

#define MAXLINES 61; /* 打印行数 */
#define MAXCHARS 80; /* 每行打印字符数 */

void center__string(char *);
FILE * fopen__laserjet(char *);

/*****
/*
/*                               主程序，列中光标定位
/*
/*
*****/

void main(int argc, char * argv[])
{

```

```

char buffer[MAXCHARS+1];
int i, lines;

if(argc < 2)
{
    puts("Input file must be specified");
    exit(2);
}

if((fin = fopen(argv[1], "rt")) == NULL)
{
    fprintf(stderr, "Unable to open %s", argv[1]);
    exit(3);
}

fout = fopen__laserjet("PRN");
fprintf(fout, "%cE", ESC); /* 打印机初始化 */

for(lines = 0; lines < MAXLINES-6; lines++) /* 计算行数 */
{
    if((fgets(buffer, MAXCHARS+1, fin)) == NULL)
        break;
}

i = MAXLINES-lines;
fprintf(fout, "%c&a%dR", ESC, i/2);

/* 若 I 为奇数, 半行走纸 (ESC =) */

if(i%2 != 0)
    fprintf(fout, "%c=", ESC);

rewind(fin); /* 从文件头开始 */

for(lines = 0; lines < MAXLINES; lines++)
{

    if((fgets(buffer, MAXCHARS, fin)) == NULL)
        break;
}

```

```

        center__string(buffer); /* 行中定位 */
        fputs(buffer, fout);
        fputc('\n', fout);
    }
    fputc(FormFeed, fout);
}

/ * * * * * /
/ * * * * * /
/ * * * * * 行中光标定位 * * * * * /
/ * * * * * /
/ * * * * * /

void center__string(char str[])
{
    int i, j, k;

    k = strlen(str);

    / * 找到行尾最后一个非空格字符，并作标记 * /

    for(i = k-1; i > 0; i--)
        if (isgraph(str[i]))
            break;

    str[i+k] = '\0';

    / * 找到行首最后一个非空格字符，并作标记 * /

    for(i = 0;; i++)
        (isgraph(str[i]) || str[i] == '\0')
            break;

    / * 对于有空格的行，用缓冲区存字符串 * /

    if(i != 0)
        emmove(str, str+i, k-i+1);

```

```

        trlen(str);
        i = (MAXCHARS-k) / 2;  /* 处理缩进格式 */
        fprintf(fout, "%c&a%dC", ESC, i);

```

/* 若字符串长度为奇数，右移半个字符位置 (1/20 英寸) */

```

        if(MAXCHARS-k) %2 != 0)
            fprintf(fout, "%c * p+%dX", ESC, 15);
    }

```

编程实例四：PCL5 命令查询例程

本例程用于检验 PCL5 命令的语法上的正确，并有联机 HELP 功能。

功能：输入为 PCL5 命令文件，程序自动检查语法，给出调试信息。

组成：

PCLPRINT.H：PCL5 命令集头文件，与源程序 PCLPARSE.C 相独立，读者只要为点阵打印机编写相应的头文件，如根据 EPSON 的 ESC/P 命令集写一个 MATRIX.H，直接使用 PCLPARSE.C 可完成相同的功能。

PCLPARSE.C：语法分析程序，采用编译技术识别特征字符串。

命令行调用方法：

PCLPARSE FILENAME.IN FILENAME.OUT

其中 FILENAME.IN 为 PCL 输入文件，FILENAME.OUT 为输出文件。

```

/*
 *
 *          PCL5 头文件: PCLPRINT.H
 *
 */

```

```

void pcl__announce__next__char(int);
void pcl__assign__font__id(int);
void pcl__assign__font__temp(void);
void pcl__default__font(int);
void pcl__delete__fonts(int);
void pcl__delete__last__char(void);
void pcl__delete__last__font(void);
void pcl__delete__macro__id(void);
void pcl__delete__macros(void);
void pcl__delete__temp__macros(void);
void pcl__disable__overlay(void);

```

```

void pcl__diabile__underline(void);
void pcl__display__codes(int);
void pcl__dot__position(void);
void pcl__download__font__char(int);
void pcl__download__font__header(void);
void pcl__enable__fixed__underline(void);
void pcl__enable__float__underline(void);
void pcl__end__raster__graphics(void);
void pcl__enable__overlay(void);
void pcl__enter__hppl__mode(int);
void pcl__enter__pcl__mode(int);
void pcl__col__wrap(int);
void pcl__execute__macro(void);
void pcl__fill__rectangle(int);
void pcl__finish(void);
void pcl__font__status(int);
void pcl__graphic__presentation(int);
void pcl__half__line__feed(void);
void pcl__horz__move(int);
void pcl__horz__position(int, long int);
void pcl__identify__macro(int);
void pcl__imaging__pattern(int);
void pcl__init(int, char * []);
void pcl__line__spacing(int);
void pcl__line__termination(char []);
void pcl__make__macro__perm(void);
void pcl__make__macro__temp(void);
void pcl__make__copies(int);
void pcl__orientation(int);
void pcl__page__length(int);
void pcl__page__size(int);
void pcl__paper__source(int);
void pcl__pattern__transparency(int);
void pcl__perforation__skip(int);
void pcl__push__pop__position(int);
void pcl__raster__compress(int);
void pcl__raster__graphics(void);
void pcl__raster__height(int);
void pcl__raster__width(int);

```