基于 USB 传输的针式打印机驱动程序开发

樊星男
大连理工大学自动化系，辽宁 大连 (116024)
E-mail: fanxingnan@163.com

摘 要: 本文从针式打印机的整体设计出发，介绍了 USB 针式打印机驱动程序的设计过程，包括使用 MDT 开发打印机小驱动程序、使用 At91sam7s32 芯片开发打印机固件程序，以及使用查表法和 CPLD 设计打印机主控程序，实现了打印机与 Windows 操作系统的交互通信以及打印任务的创建与执行。从前期实验结果和产品进入市场后的用户反馈情况来看，本设计方案的可行性和应用性较强。

关键词: 针式打印机；打印机驱动程序；打印机固件；打印机主控程序；CPLD
中图分类号: TP334.8

1. 引言

针式打印机曾经在相当长的一段时间占据打印机市场的主导地位，但是近年来由于喷墨、激光等非击打式打印机的冲击，针式打印机的市场份额逐年下降。即便如此，由于针式打印机在票据打印领域的不可取代性，同时鉴于当前国内打印机市场一直被国外品牌充斥的现状，我们设计并开发了这款基于 USB 传输的针式打印机。一个完整的 USB 针式打印机系统包括打印机软件和打印机硬件两部分，其中打印机软件负责打印任务的创建，而打印机硬件负责打印任务的执行。如果仔细划分，打印机软件又分为运行于主机系统的应用程序、打印机驱动程序和端口驱动程序，以及运行于打印机的打印机固件程序、打印机命令解析程序和打印机控制程序，打印机命令解析程序和打印机控制程序接受打印机主控程序的调度。打印机硬件也可分为电路组成和执行机构两部分。

2. 针式打印机系统设计

一个打印任务的完成必须要经过任务创建和任务执行两个步骤。在 Windows 操作系统下，打印任务起始于应用程序，如图 1。当用户使用应用程序的打印功能后，应用程序将用户的打印任务提交给打印机驱动程序，打印机驱动程序将该任务解析为一些字符数据，包括打印机控制命令数据和打印图形数据。这些数据通过端口驱动程序发送到打印机。当打印机接收到打印数据后，首先解析这些数据为打印机的实际执行指令，然后由控制程序操作执行机构将位图信息打印到媒介上。整个打印数据的传输都建立在主机与打印机正常通信的基础上，这项工作由打印机固件程序完成。

了解了打印机系统的工作流程和系统内各个组件的作用，我们来对系统各个组件进行设计。

(1) Windows 打印机驱动程序设计

开发打印机驱动程序通常有两种途径：一是开发一个完整的驱动程序，二是利用微软提供的通用打印机驱动程序(Unidrv) 开发小型驱动程序(Minidriver) 。开发一个完整的驱动程序工作量大且过程复杂，因此我们采用通用打印机驱动程序+小型驱动程序来开发打印机驱动程序。

(2) 打印机固件程序设计

在 USB 协议中，不同的设备类具有着不同描述符和设备请求，USB 打印机作为一种特殊的 USB 设备也有其特殊性，必须严格按照协议规定进行设计。在本文中，我们使用集成

有 USB 通信口的 At91sam7s32 芯片完成打印机固件程序开发。

(3) 打印机主控程序设计

打印机主控程序由打印机解析程序模块和打印机控制程序模块组成。解析模块通常使用顺序解析法解析数据，但是这种方法结构死板，解析效率低，为此本文提出了一种基于表格的解析法；对打印机执行机构进行控制是整个打印机系统的核心。一般采用的方法是使用单片机作为系统控制器，但是控制打印机需要同时控制两个步进电机和 24 个打印针头，对单片机处理速度要求较高，同时还要使用大量的单片机接口资源， 一般单片机都无法满足要求，为此我们提出了一种单片机+CPLD 的打印机控制方法，有效的解决了这个问题。单片机使用上面提到的 At91sam7s32，CPLD 使用 Altera 生产的 EPM1270 芯片，该芯片具有 1270 个逻辑单元和 212 个用户引脚，满足打印机控制的速度要求和接口数量要求。

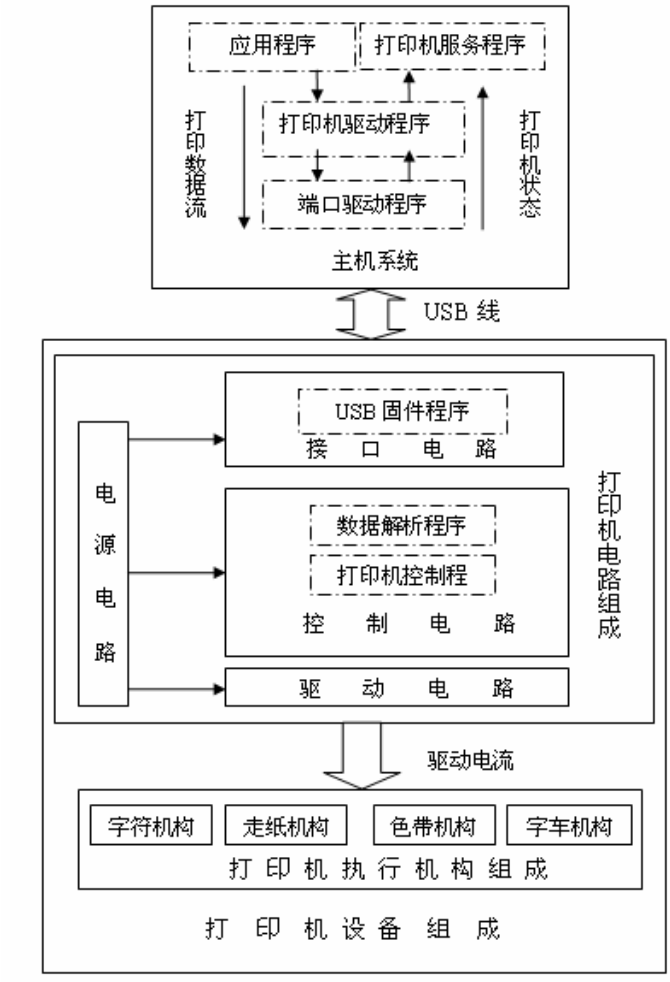


图 1 针式打印机系统总体设计图

Fig.1 The Globle Design of Stylus Printer System

3. Windows 打印机驱动程序设计

Windows 打印体系结构由打印假脱机系统（SPOOL）和一系列的打印机驱动程序（DRIVER）组成。 SPOOL 是 Windows 打印系统的核心，它有一个关键的作用就是对打印任务进行路由，将打印任务传递到正确的打印机驱动程序进行处理。打印驱动程序由打印图形动态链接库和打印接口动态链接库组成，打印图形动态链接库负责将打印任务解析为打印

命令和位图数据；打印接口动态链接库则负责用户与打印驱动程序的交互，比如通过此接口选择进纸器、纸张等。

3.1 小型驱动程序的开发

小型驱动程序是建立在通用驱动程序(Unidrv)基础之上的。通用驱动程序由打印图形动态链接库 UNIDRV.DLL、打印接口动态链接库 UNIDRVUI.DLL 以及一些文本文件和二进制结构文件组成，这些文本文件和二进制文件文件构成了我们的小驱动程序。小型驱动程序的文本文件称为 GPD（General Printer Description)文件，一个小型驱动程序可以由一个或多个 GPD 文件构成。小型驱动程序的二进制文件包括 GTT(Glyph Translation Table)文件和 UFM（Unidrv Font Metric）文件，GTT 文件用来描述打印机所能支持的字体，UFM 文件用来描述字体的尺寸大小。由于 GTT 文件和 UFM 文件不是小型驱动程序所必须的文件，所以这里只介绍 GPD 文件的设计方法。

3.2 GPD 文件

为了能够方便的开发小型驱动程序，微软在 DDK 中为我们提供了一个工具 Print MDT。通过使用 MDT 可以方便的管理和开发小型驱动程序，该工具可以作为 GPD 文件的编辑器和语法检查器。

(1)：GPD 文件的头部分

GPD 文件的头部分包含有多项内容，其中最重要的是驱动名称和资源文件名称，GPD 规定用关键字 *ModelName 表示驱动程序的名称，用*ResourceDLL 表示资源文件名称，示例代码如下：

```
*GPDSpecVersion: "1.0"
*Include: "StdNames.gpd"
*GPDFileVersion: "1.00"
*GPDFileName: "NX500.GPD"
*ModelName: "NX-500"
*MasterUnits: PAIR(360, 360)
*ResourceDLL: "NX-500.Dll"
```

NX-500 是我们设计的打印机名称，NX-500.Dll 是编译后得到的资源文件。

(2) 设定打印清晰度

打印清晰度是打印机非常重要的一个性能指标，用关键字 Resolution 表示，通常有 180x180 和 360x360 两种设定模式，其部分代码如下：

```
 *Feature: Resolution
{*rcNameID: =RESOLUTION_DISPLAY
*DefaultOption: Option3
*Option: Option1
{*Name: "360 x 360 " =DOTS_PER_INCH
*DPI: PAIR(360, 360)
*TextDPI: PAIR(360, 360)
*PinsPerLogPass: 48
*PinsPerPhysPass: 24
*Command: CmdSendBlockData { *Cmd : "<1B>*(" %l{NumOfDataBytes / 3}          }
*Option: Option2{……}
```

*Option: Option3{……}}

DPI 表示横竖方向上每英寸的针数；PinsPerLogPass 表示逻辑距离，PinsPerPhysPass 表示物理距离，*Command 项表示的是在打印机描述语言中对设定打印清晰度的命令码。

(3) 设定打印机纸张

设定纸张大小就是要告诉打印机当前的打印作业选取的纸张类别，用关键字 PaperSize 表示。部分代码如下：

*Feature: PaperSize
{*rcNameID: =PAPER_SIZE_DISPLAY
*DefaultOption: A4
*Option: A6_ROTATED
{*rcNameID: =RCID_DMPAPER_SYSTEM_NAME
*Command: CmdSelect
{*Order: DOC_SETUP.4
*Cmd: "<1B>2<1B>C<19>" }
……
}}

*Order 表示命令码生成顺序，*Cmd: "<1B>2<1B>C<19>"表示打印机语言中 A6 纸的命令码。

(4) 打印机位置控制

前面介绍的都是打印属性，这里来介绍打印机位置控制命令。打印机在进行打印的时候，不仅需要知道打印的内容，还必须要知道在何处打印，以下指令控制打印机的移动位置。

① *Command: CmdXMoveAbsolute { *Cmd : " ….."}，表示打印机针头横向移动的绝对位置，这个“绝对”表示这个距离值是相对于纸张左边沿的移动距离，而不是相对于当前针头的位置。

② *Command: CmdYMoveRelDown { *Cmd : "…… " }，表示纵向移动的距离，这个距离是相对于当前针头位置。

③ *Command: CmdCR { *Cmd : "<0D>" }表示换行。

④ *Command: CmdLF { *Cmd : "<0A>" }表示回到行首。

⑤ *Command: CmdFF { *Cmd : "<0C>" }表示换页。

4. 打印机固件程序设计

4.1 打印机描述符设计

USB 设备通过描述符来定义自身的特性。一个基本的 USB 设备需要实现也必须实现四个描述符，分别是：设备描述符、配置描述符、接口描述符和端点描述符。

(1) 设备描述符：USB 设备描述符用于表示 USB 设备的总体信息，一个 USB 设备只能有一个设备描述符。为方便描述符的管理，在 At91sam7s32 芯片内可以将描述符设计为结构体，示例代码如下：

```
typedef struct {
    unsigned char bLength;
    unsigned char bDescriptorType;
    unsigned short bcdUSB;
```

```
    unsigned char bDeviceClass;
    unsigned char bDeviceSubClass;
    unsigned char bDeviceProtocol;
    unsigned char bMaxPacketSize0;
    unsigned short idVendor;
    unsigned short idProduct;
    unsigned short bcdDevice;
    unsigned char iManufacturer;
    unsigned char iProduct;
    unsigned char iSerialNumber;
    unsigned char bNumConfigurations;
} USBDeviceDescriptor;
```

在打印机设备类的开发中，DeviceClassb, DeviceSubClass，DeviceProtocol 三项的值均为 0，表示 USB 设备的接口互相独立，这三项的值是按照 USB 协议中对打印机类的描述设定的；idVendor 由 USB 芯片生产商提供，这个值是由 USB 官方组织统一发放的。我们采用的是 ATMEL 公司生产的 At91sam7s32 芯片，该公司的 USB 供应商号码为 03EB；idProduct 由 USB 设备生产商自行定义。

(2) 配置描述符：在 USB 设备中可以有一个或多个配置，而每一个配置对应一个配置描述符。

在 USB 设备的开发中，TotalLength 表示的是一个完整配置的长度，即配置描述符长度 + 接口描述符长度 + IN 端口描述符长度 + OUT 端口描述长度；NumInterfaces 表示当前配置接口数；Attributes 表示 USB 设备不采用总线供电，不支持远程唤。

(3) 接口描述符：USB 设备的接口是一个端点的集合，负责完成该 USB 的特定功能。每一个 USB 接口都必须有一个接口描述符，用来表示接口的特性，包括接口的端点个数、所属的设备类和子类等。

InterfaceClass 取值 0x07, InterfaceSubClass 取值 0x01，这两个值分别表示的是打印机设备类的基类和子类，这两个值也是由 USB 协议定义的；InterfaceProtocol 取值 0x02，表示的是打印机接口协议。USB 官方组织为打印机定义了 4 种接口协议，分别是 0x01 单向接口协议、0x02 双向接口协议、 0x03 与 1284.4 接口协议兼容的双向接口协议，以及 0xff 用户自定义接口协议。单向接口协议的 USB 打印机只有一个 OUT 端点；双向接口协议有 IN 和 OUT 端点各一个；与 1284.4 接口协议兼容的双向接口协议也具有 IN 和 OUT 端点各一个。我们采用的是双向接口协议，即打印机具有 IN 和 OUT 端口各 1 个，并且 USB 协议还规定，这两个端口的传输类型必须为块传输。

(4) 端点描述符：端点描述符用于指出 USB 设备端点的特性，包括其所属的传输类型，传输方向等信息。

EndpointAddress 值分别设置为 0x01 和 0x82，表示将 1 号端口设置为 OUT 口，将 2 号端口设置为 IN 口，注意 IN 和 OUT 都是相对 USB 主机来说的，所以这里的 IN 口是打印机向 USB 主机传输数据，OUT 口是 USB 主机向打印机传输数据；bmAttributes 取值 0x02 表示采用块传输；MaxPacketSize 代表端点数据包最大的字节数，在全速设备中，一般是 64 个字节，在高速设备中是 512 个字节，我们的打印机是一个全速设备，所以取值 64；Interval

仅在中断传输中使用，所以必须为 0。

配置描述符、接口描述符和端点描述符的结构体设计方法与设备描述符相同。

4.2 打印机设备请求机制设计

主机通过设备请求机制读取 USB 设备描述符，达到对设备的识别。USB 协议规定了 11 种标准的 USB 设备请求，包括分配 USB 设备地址、读取设备状态等，同时还为每一种设备类规定了其特有的设备请求。打印机设备类有 3 个自己的特殊请求。

(1) 获取设备 ID。它的返回值是一个 IEEE-1284 字符串。IEEE-1284 字符串是由 IEEE-1284 协议所定义，用来确认所连接打印机的属性和能力。IEEE-1284 有很多关键字，有四个关键字 MANUFACTURER、MODEL、CO

MMEND SET 和 CLASS 必须包括。 MANUFACTURER 和 MODEL 必须与打印机驱动安装文件 INF 文件中对这两项的描述一致，否则 USB 主机将无法把打印机驱动程序和 USB 打印机连接起来。以下是一个示例字符串：

char str1284[ ] = “XXMANUFACTURER：ZhongYing；MODEL：NX-500；COMMEND SET：EPSON；CLASS：PRINTER；DESCRIPTION：ZY Printer ”；

str1284[0] = 0； str1284[1] = 92；

IEEE-1284 字符串的前两位用来表示字串长度。

(2) 获得当前打印机的状态。它的返回值只有一个字节，并且这个字节是按位定义的。每一位表示了打印机的一种状态，包括是否缺纸，是否连接等。

(3) 软件复位。软件复位没有返回值，它的作用是清除打印机缓冲，重置打印机端口状态。

在 At91sam7s32 中，每个设备请求都会触发芯片内的 USB 中断并发送一个数据包，通过读取数据包中的设备请求码来区分不同请求。比如当收到请求码为 0xA1 时，固件程序必须通过控制端口向主机返回 1284 字符串，如果超时则主机不能正确识别打印机。

5．打印机主控程序设计

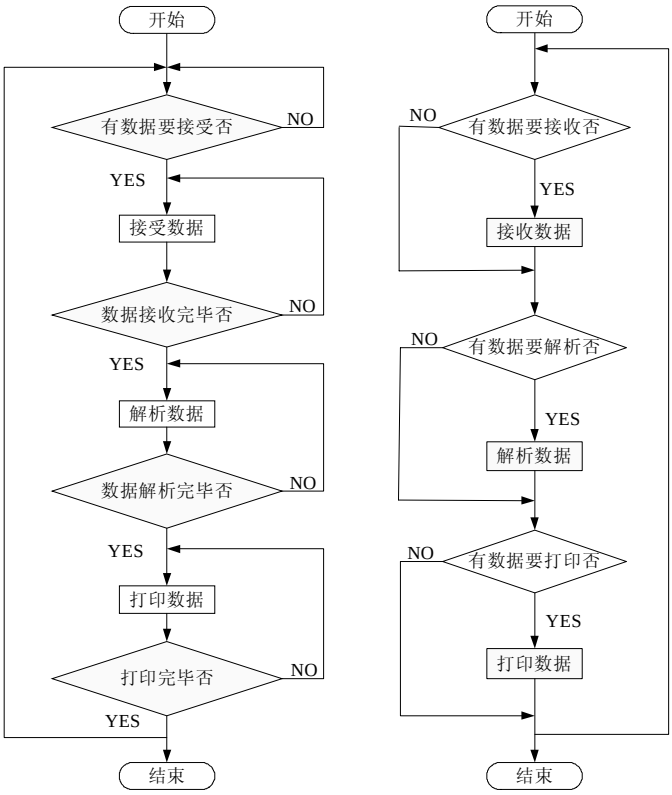
5.1 主控程序工作方式设计

打印机有两种工作方式.第一种是顺序工作方式，如图 2（a），当一个打印任务发起后，数据接收程序首先工作，将所有数据接收完毕，然后数据解析程序开始解析，直至将所有数据解析完，最后是打印机控制程序工作，负责将所有解析得到的命令打印完毕。这种工作方式逻辑简单明确，容易设计，但是在整个打印过程中，任何一个环节发生异常都会导致整个系统奔溃，稳定性较差。另一种是并行工作方式，如图 2（b）。并行工作方式引入了操作系统中任务调度机制。在整个打印机主控程序中有三个任务，数据接收、数据解析和打印控制，主控程序在工作过程中，根据每个任务所处状态依次给予 CPU 时间，每部分程序都只完成自己所有工作中的小部分内容，然后就让出 CPU 时间给下一个任务。这种工作方式稳定性较好，因此我们采用并行工作方式作为打印机主程序的工作方式。

5.2 打印机解析程序设计

打印机语言中对各个打印命令都进行了定义，每个命令都是由一个字符串组成，长度不定。比如 “0x1B 0x28 0x63 XXXXXX”表示设置打印纸张页长，“0x0A”表示换行等。

通常采用的解析方式是顺序逐个解析每个字符，比如在进行页长设置时要连续判断三次，如果发现有三个连续字符依次是“0x1B 0x28 0x63”，那么才能知道这是设定页长命令，然后再继续读取后面 6 个参数计算页长。这种解析方式结构死板，不利于后期升级，并且解析效率低。为此，我们设计了一种基于表格的打印机命令解析方式，这种方式结构灵活，方便于解析程序的升级，同时也提高了解析效率。查表解析法原理如图 3。当解析程序接收到一段字符串后，依次提取每个字符，将每个字符看过一维表的下标查找该表格的内容，表格的内容指明了进行下一步解析所要完成的工作。当前解析的命令不同，下一步解析所要完成的工作也不同，可能是发送一条打印机动作命令，比如换行，因为换行只有“0x0A”一个字符，也可能是查找二级表或者三级表。



（a）顺序工作方式　　（b）并行工作方式

图2 打印机工作方式

Fig.2 The Pattern of Printer Work

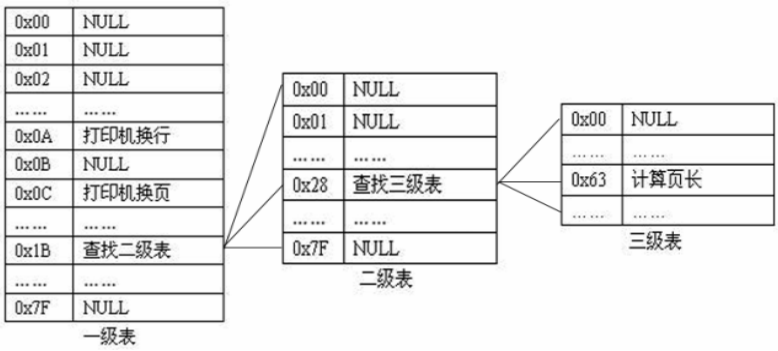


图3.查表法原理图

Fig.3 The Schematic of Look-up Table Method

查表法的关键是建表。

首先定义结构体，该结构体中需要包括一个通用的函数指针，如下示例：

struct strOperate{ void (*myOpefun) (); }

然后，创建该结构体的数组。strOperate firstTable[256],那么这个数组就是我们的一级表。至于二级表和三级表的创建方法和一级表相同，我们可以根据实际需要进行创建。

最后，给指针赋值。firstTable[下标].myOpefun = Myfun；Myfun()是我们编写的实际解析函数，它具有指明下一步解析任务的功能。

创建了表后，在解析过程程序中，依次提取字符调用相应函数。例如我们收到的字符是0x0A，那么通过语句 firstTable[0x0A]. myOpefun()就能很快的调用到换行函数。

由此可以看出，查表法解析速度快，效率高，只需一步就能找到相对应的解析命令；此外，查表法结构简单灵活，在后期解析程序升级过程中，只要通过 firstTable[下标].myOpefun = Mynewfun 语句就能完成升级而不需要改变整体结构。

5.3 打印机控制程序设计

为了实现对打印机的控制，较多采用的方案是以单片机作为控制系统的微处理器，通过一些大规模的集成电路来控制。但是这种方案中微处理器所需的周边器件较多，对整个系统的稳定性、可靠性有较大影响，同时由于打印机控制器件较多，包括两个步进电机和 24 个打印针头，所以单片机处理速度也成为制约提高系统实时控制性的一个瓶颈。此外，针式打印机具有 24 个针头，每个针头都需要单独控制，这样就要占用单片机大量的接口资源，而一般单片机并没有这么多的接口。为此，我们采用单片机 + CPLD 的方法来实现对打印机的控制。在本文设计中，单片机采用的是 Atmel 公司生产的 AT91sam7s32 芯片，该芯片是一款集成有 USB 通信口的 ARM7 内核控制器；CPLD 使用 Altera 生产的 MAXIIEPM1270 芯片，该芯片具有 1270 个逻辑单元和 212 个用户引脚，可以满足打印机控制的速度要求和接口要求。

在单片机 + CPLD 的打印机控制系统中，CPLD 内运行打印机运动控制算法程序，负责打印机执行机构的精确控制，单片机内运行任务调度程序，负责打印指令的调度。任务调度程序与运动控制算法程序之间，通过 CPLD 为单片机提供的寄存器接口进行通信。由于篇幅有限，在此我们仅对打印机控制程序的总体工作流程进行介绍，对于 CPLD 的设计暂不叙述。

打印机控制程序如图 4，当打印机控制程序开始执行时，首先检查 CPLD 状态寄存器，查看当前 CPLD 是否有任务正在执行，如果有就交出 CPU 时间给其他程序模块；如果当前没有任务，检查指令栈中是否有指令需要执行，如果有指令，就将该栈顶指令放入 CPLD 的指令寄存器中，然后设置状态寄存器通知 CPLD 执行。如果没有指令则继续让出 CPU 时间给其他程序模块。当 CPLD 将指令执行完毕后，修改状态寄存器为无任务装态，为接受下条指令准备。

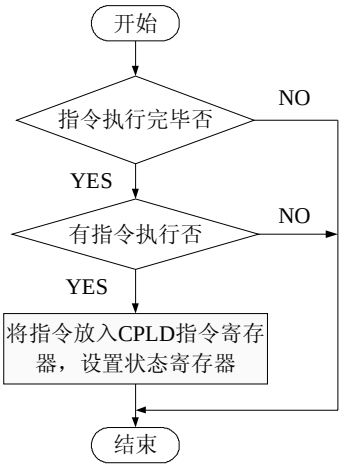


图4 打印机控制程序流程图
Fig.4 The Flow Chart of Printer Control Program

5 结论

从打印机的实验结果以及市场客户反馈，我们可以得出以下结论:

(1) 基于通用驱动程序的打印机小驱动程序开发时间短，并且工作效果稳定可行。

(2) 用查表法设计的打印机解析程序，机构灵活，易于升级，并且工作效率高。

(3) 用单片机+CPLD 设计打印机控制系统工作稳定，打印速度快。

参考文献

[1] 陆世爵.针式打印机[M].北京: 电子工业出版社，1989: 23-81
[2] 金山编辑室.EPSON 打印机使用大全[M].北京:清华大学出版社，1996
[3] 薛园园.USB 应用开发技术大全[M].北京：人民邮电出版社，2007
[4] 何斌，黄进，陈其昌.Windows 环境下打印机驱动程序的设计[J].电子计算机与外部设备, 2000,24(3):19-21
[5] 田玉敏，燕红锁.Windows2000 下打印机驱动程序的开发[J].计算机工程，2002,28(3):214-216

The Driver Development of Stylus Printer based on USB Transmission

Fan Xingnan
Department of Automation, Dalian University of Technolgy. Liaoning Dalian (106024)

Abstract

A driver design scheme of a USB stylus printer is presented in this paper. The scheme includes the printer mini-driver designed by MDT, the printer firmware by using at91sam7s32 and the printer master program based on look-up method and CPLD. The interactive communication between printer and Windows operation system is realized by using the design scheme, and printing tasks are created and executed successfully. From the results of early experiments and users' response, the design scheme is feasible and applicable.

Keywords: stylus printer; printer driver ; printer firmwear; printer master control program; CPLD

作者简介：
樊星男，男，1984 年生，硕士研究生，主要研究方向：控制理论与控制工程。