

当前嵌入式系统技术已得到了广泛应用,但传统嵌入式系统的人机接口多采用小键盘操作的文本菜单方式,用户操作较为不便。本文介绍了一种利用 PS/2 接口鼠标,在点阵 LCD 的单片机系统上实现图形化用户界面的方案。用窗口菜单和图形按钮取代了传统的键盘操作,具有成本低、效果好等特点,具有很强的实用性。

1 PS/2 接口和协议

1.1 接口的物理特性

PS/2 接口用于许多现代的鼠标和键盘,由 IBM 最初开发和使用。物理上的 PS/2 接口有两种类型的连接器:5 脚的 DIN 和 6 脚的 mini-DIN。图 1 就是两种连接器的引脚定义。使用中,主机提供+5V 电源给鼠标,鼠标的地连接到主机电源地上。

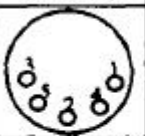
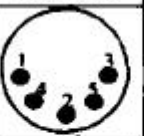


	插座(孔)	插头(针)		5脚的DIN	6脚的mini-DIN
5脚的DIN			1	时钟 (CLOCK)	数据 (DATA)
			2	数据 (DATA)	未实现、保留
			3	未实现、保留	电源地 (GND)
6脚的mini-DIN			4	电源地 (GND)	电源+5V (VCC)
			5	电源+5V (VCC)	时钟 (CLOCK)
			6		未实现、保留

图 1 PS/2 接口连接器引脚定义

1.2 接口协议原理

PS/2 鼠标接口采用一种双向同步串行协议。即每在时钟线上发一个脉冲,就在数据线上发送一位数据。在相互传输中,主机拥有总线控制权,即它可以在任何时候抑制鼠标的发送。方法是把时钟线一直拉低,鼠标就不能产生时钟信号和发送数据。在两个方向的传输中,时钟信号都是由鼠标产生,即主机不产生通信时钟信号。

如果主机要发送数据,它必须控制鼠标产生时钟信号。方法如下:主机首先下拉时钟线至少 100μs 抑制通信,然后再下拉数据线,最后释放时钟线。通过这一时序控制鼠标产生时钟信号。当鼠标检测到这个时序状态,会在 10ms 内产生时钟信号。如图 3 中 A 时序段。主机和鼠标之间,传输数据帧的时序如图 2、图 3 所示。2.2 数据包结构在主机程序中,利用每个数据位的时钟脉冲触发中断,在中断例程中实现数据位的判断和接收。在实验过程中,通过合适的编程,能够正确控制并接收鼠标数据。但该方案有一点不足,由于每个 CLOCK 都要产生一次中断,中断频繁,需要耗用大量的主机资源。

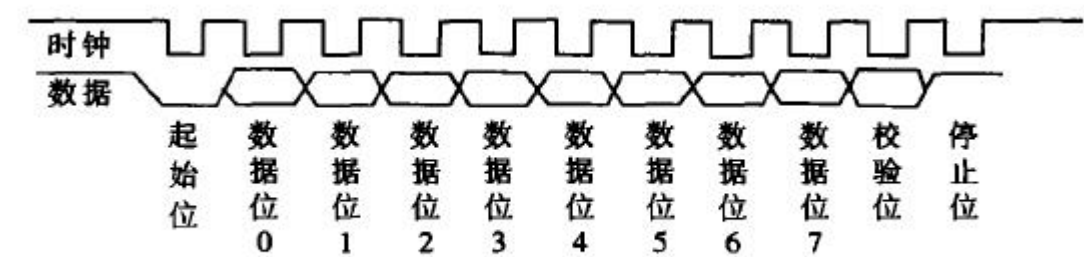


图 2 鼠标到主机的传输时序

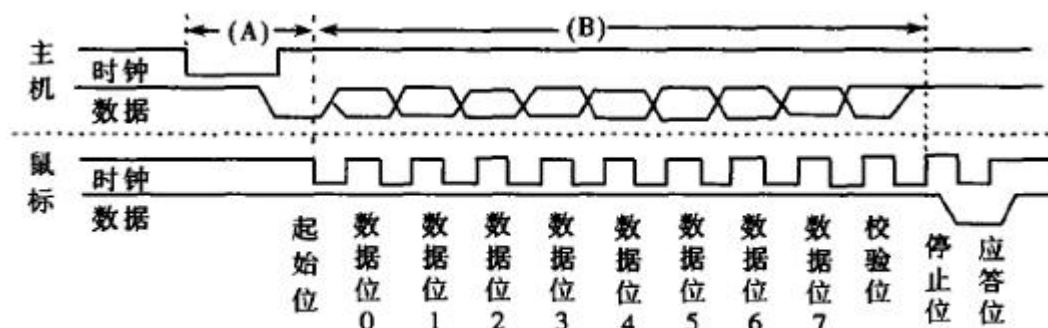


图 3 主机到鼠标的传输时序

2 PS/2 鼠标的工作模式和协议数据包格式

2.1 PS/2 鼠标的四种工作模式

PS/2 鼠标的四种工作模式是:Reset 模式,当鼠标上电或主机发复位命令 0xFF 给它时进入这种模式;Stream 模式鼠标的默认模式,当鼠标上电或复位完成后,自动进入此模式,鼠标基本上以此模式工作;Remote 模式,只有在主机发送了模式设置命令 0xF0 后,鼠标才进入这种模式;Wrap 模式,这种模式只用于测试鼠标与主机连接是否正确。

PS/2 鼠标在工作过程中,会及时把它的状态数据发送给主机。发送的数据包格式如表 1 所示。

表 1 鼠标发送的数据包格式

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 1	Y overflow	X overflow	Y sign bit	X sign bit	Always 1	Middle Btn	Right Btn	Left Btn
Byte 2	X Movement							
Byte 3	Y Movement							
Byte 4	Z Movement							

Byte1 中的 Bit0、Bit1、Bit2 分别表示左、右、中键的状态,状态值 0 表示释放,1 表示按下。Byte2 和 Byte3 分别表示 X 轴和 Y 轴方向的移动计量值,是二进制补码值。Byte4 的低四位表示滚轮的移动计量值,也是二进制补码值,高四位作为扩展符号位。这种数据包由带滚轮的三键三维鼠标产生。若是不带滚轮的三键鼠标,产生的数据包没有 Byte4 其余的相同。

3 设计与实现

3.1 接口设计

因为 PS/2 鼠标接口采用双向同步串行协议,时钟脉冲信号 以下皆称 CLOCK 总是由鼠标产生。因此,可以考虑这种方案:鼠标的 CLOCK 接主机的一外中断线,数据线 以下皆称 DATA 接主机的某一 I/O 口线,如图 4 所示。

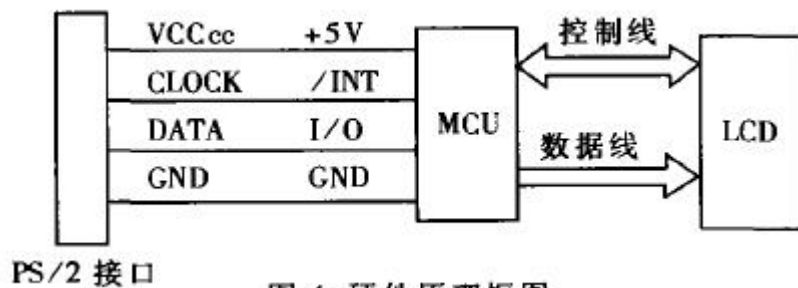


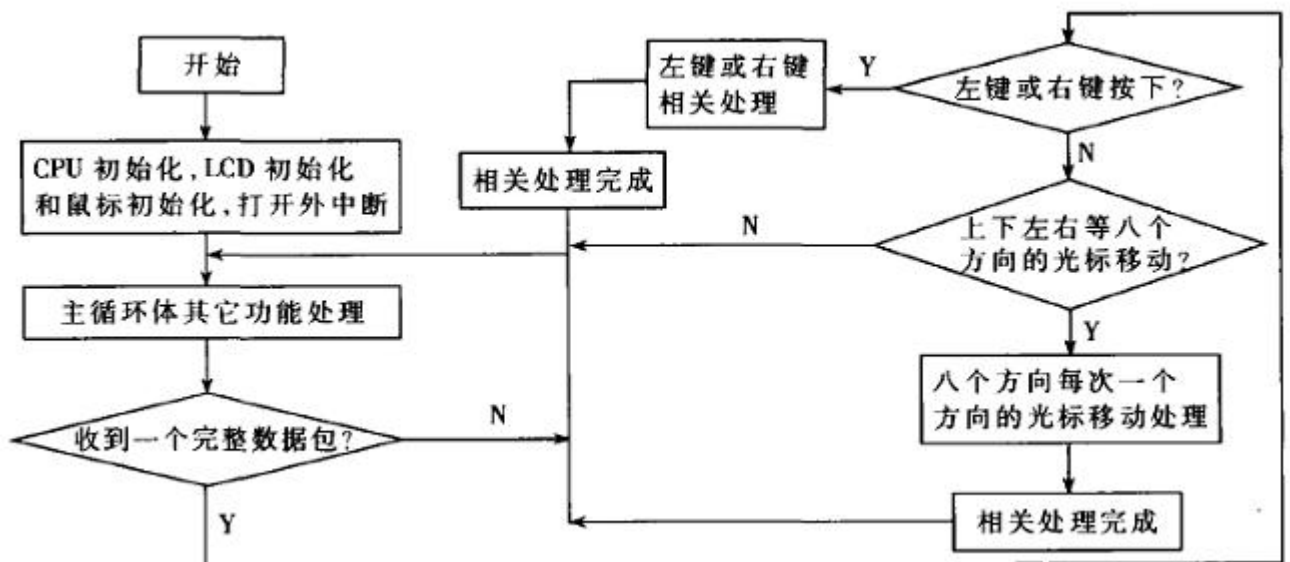
图 4 硬件原理框图

由于鼠标与主机之间以双向同步串行协议传送数据,若不考虑 **CLOCK**,仅考虑 **DATA**,则其数据帧的时序与单片机的 **UART** 异步串行时序类似。所以,采用了另一种方案:鼠标的 **CLOCK** 仍旧接主机的外中断,但鼠标的 **DATA** 接 **UART** 的接收脚 **RxD**。参照图 4 **DATA** 改接 **RxD**。在初始化过程中,主机利用 **CLOCK** 的外中断和 **RxD** 脚的 **I/O** 口线功能实现数据的传输。初始化完成后,切换到 **RxD** 功能即 **UART** 的接收引脚功能。因为鼠标已处于 **Stream** 模式的工作状态,这时鼠标能主动发送数据。这样,主机可以在每收到一帧数据时才中断一次。中断次数大大降低,减少了主机资源的耗用。

不过,在此方案中,必须实现另一个功能:主机波特率的自适应。因为 **PS/2** 接口的鼠标一般工作在 **10kHz~20kHz** 时钟频率。不同厂家制造的鼠标工作的时钟频率不同。嵌入设备主机要做到与不同鼠标的波特率同步和自适应,才能够正确接收鼠标传送来的数据。波特率的自适应是这样实现:鼠标上电自检时会产生一串时钟脉冲,利用鼠标时钟脉冲产生的中断,结合主机的定时器测量时钟脉冲周期,可以得出所用鼠标的时钟频率,进而求出波特率。通过设置相应的波特率寄存器,实现了波特率的自适应。

3.2 软件实现

软件实现原理框图如图 5 所示。



(1) 鼠标初始化

最简单的初始化就是当鼠标上电自检完成后,主机给鼠标发送一个使能鼠标数据传送命令字节 (**0xf4**),鼠

标就会在默认设置状态下工作。主机也可实现自定义初始化,如:复位三次(Snd_CMD(0xff),Snd_CMD(0xff),Snd_CMD(0xff); 设置采样率:Snd_CMD(0xf3),Snd_CMD(0x0a); 设置解析度(2 点/毫米):Snd_CMD(0xe8),Snd_CMD(0x01);设置缩放比例(1:1):Snd_CMD(0xe6);使能鼠标数据传送:Snd_CMD(0xf4)。鼠标每收到一个命令字节都会给出一个应答字节(0xfa)。

(2)两种方案的实现过程

两种方案的软件实现过程基本相同。只是后一种方案中,初始化时还要实现主机波特率的自适应,关闭时钟脉冲中断和打开串口中断。此后主机利用 UART 的接收功能接收鼠标数据。

(3)图形化人机接口(GUI)的实现

在点阵式 LCD 显示屏上实现图形化的人机接口界面,主要有两个方面:一个是菜单图标的实现;另一个是鼠标光标的实现。实现菜单图标,显示屏一般工作在图形显示模式。菜单图标有正常显示状态和反显状态,它们都用函数实现:voidDraw_ICON(signed int xICON, signed int yICON,unsigned char *pDatlCON)。xICONyICON 是图标所在位置的左上角坐标值,pDatlCON 是各个图标及其不同显示状态的点阵码值。反显状态是当图标被光标滑到或点取时才显现的。实现鼠标光标,又分两种情况。一种是单层显示的 LCD,只能由程序画出鼠标光标。但是,当光标移动较快时,画出光标的点阵图形需要耗用较多的主机资源。另一种是有双层显示和光标功能的 LCD,只需程序控制它的光标移动位置,无需程序画出光标点阵图形,因而耗用主机资源较少,实现起来效果较好。

两种方案简单、明了,容易实现,都已在实验中得到验证。并且,后一种方案已在某一仪表系统中得到成功应用。总体来说,随着嵌入式处理器性能的不提高,在嵌入设备中接入鼠标,既可灵活使用,也可减少因接入许多按键而占用的口线数,还能使 LCD 的图形化显示界面更美观、更人性化。