**PC手柄与单片机之间的通讯**

[ps2](http://blog.163.com/zgbn2002@126/blog/#m=0&t=1&c=fks_094069084095084064085086074065081083081067085087086) 2010-05-09 17:00:44 阅读118 评论0   字号：大中小 订阅

本科毕设参加了亚太大学生机器人比赛国内选拔赛，接触了机器人的机械和控制系统的设计。我主要负责手动机器人控制系统的设计。由于控制任务不是很复杂，对数据处理能力和控制实时性上要求不是很高，所以主控芯片选择了ATMEL的51系列，控制器选的是平时玩拳皇和实况时用的PS手柄。下面是单片机与手柄的通讯的设计过程：

**PC手柄的硬件剖析**

输入设备在控制系统中是十分重要的,是人机交互的界面。手柄输入在一些特殊的场合比如机器人控制中十分方便,直观。比赛中采用的是PC接口(25针)的数字手柄（Psx  Digital）。这种接口的手柄通讯协议简单，且与单片机通讯时不需要转换接口。在与单片机通讯时只需要四根信号线，手柄与单片机是通过串行方式通讯的市场上可以买到的普通PC手柄大都由PS手柄改装而成.图1是PS手柄到PC手柄的改装线路.由图可以看出,普通PS手柄插头中第3针和第8针没有用,剩余的7根针所接的线从左到右的颜色依次为:棕,桔,黑,红,黄,蓝,绿.每根线都有固定的作用.手柄与主机之间是通过串行方式通讯的.针脚具体含义如图2:

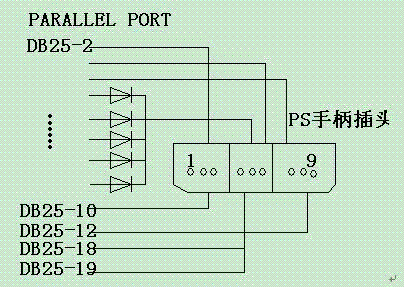


图1  PC手柄的改装接线图

|  |  |  |
| --- | --- | --- |
| 针脚 | 定 义 | 用    途 |
| 1 | DATA | 信号流方向:从手柄到主机.此信号是一个8bit的串行数据.同步传送于时钟的下降沿(输入输出信号在时钟信号由高到低时变化,所有信号的读取在时钟前沿到电平变化之前完成) |
| 2 | COMMAND | 信号流方向:从主机到手柄.此信号和DATA相对,是一个8bit的串行数据,同步传送于时钟下降沿 |
| 3 | N/C | 未用 |
| 4 | GND | 电源地和信号地 |
| 5 | Vcc | 电源电压,有效工作电压3V-5V |
| 6 | ATT | ATT用于提供手柄触发信号.信号在通讯期间处于低电平 |
| 7 | CLOCK | 信号流方向:从主机到手柄.用于保持数据同步 |
| 8 | N/C | 未用 |
| 9 | ACK | 从手柄到主机的应答信号.此信号在每个8bit数据发送之后的最后一个时钟周期变低,并且ATT一直保持低电平.如果ACK信号保持60μs 不变低，PS主机会试另一个外设 |

 图2  PC手柄针脚含义

**PC手柄的信号通讯协议**

手柄通信都是8 bit串行数据最低有效位先行。PC手柄总线的所有时码在时钟下降沿都是同步的。传送一个字节的情况如图3：

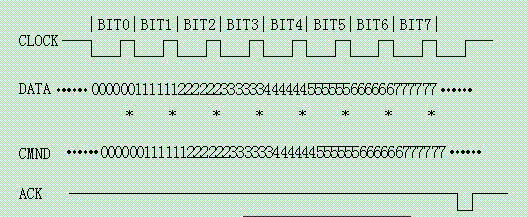


    图 3  PC手柄传输数据时序图

数据线的逻辑电平在时钟下降沿驱动下触发改变。数据的接收读取在时钟的前沿（在记号\*处）到电平变化之前完成。在被选手柄接收每个COMMAND信号之后，手柄需拉低ACK电平在最后一个时钟。如果被选手柄没有ACK应答，主机将假定没手柄接入。

      当主机想读一个手柄的数据时，将会拉低ATT线电平并发出一个开始命令（0x01）.手柄将会回复它的ID(0x41=数字,0x23=Negcon,0x73=模拟红灯,0x53=模拟绿灯)。在手柄发送ID字节的同时，主机将传送0x42 请求数据。随后命令线将空闲，手柄送出0x5A,表明数据准备好了。图4是一个数字手柄的时钟信号：

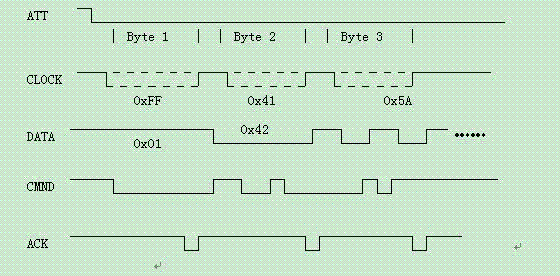


                                                 图 4  数字手柄时钟信号

在手柄执行初始化命令之后将发送它的所有的数据字节（数据手柄只有两个字节）。在最后字节发送之后使ATT高电平，手柄无需ACK应答。数字手柄的数据传送如图5：

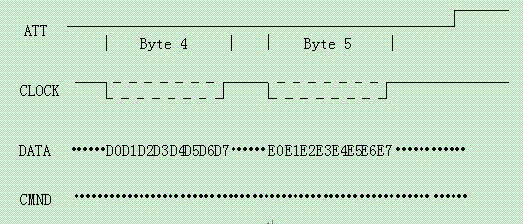


图 5  数字手柄的数据传送

标准数字手柄的实际发送字节（所有按键按下均有效）如图6：

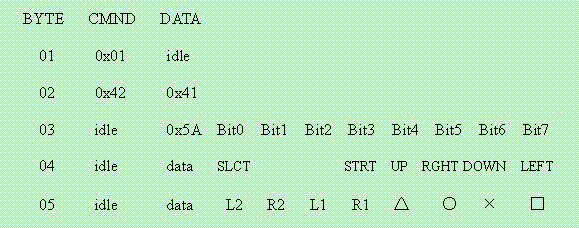


                      图 6  数字手柄按键码位数据图

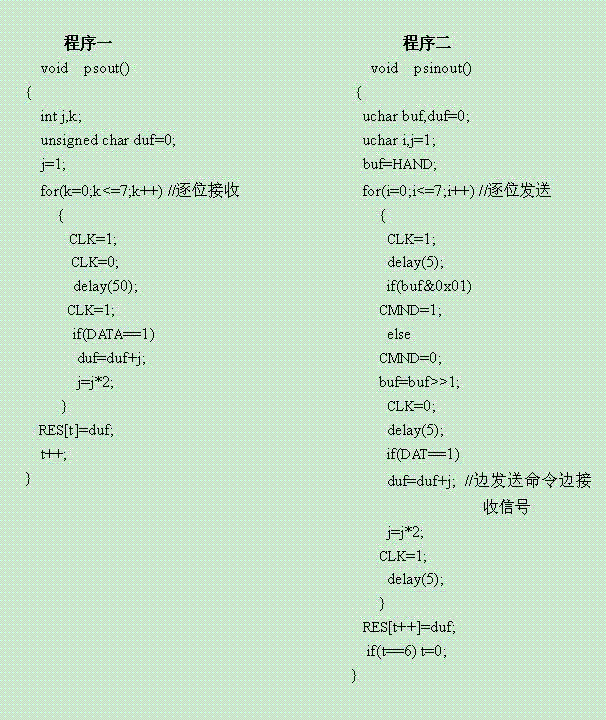
    手柄是以串行方式和主机进行通讯，通讯时钟由CLOCK提供，在实时性要求不高的场合握手信号可以省去。

**PC手柄与单片机的通讯调试**

根据通讯协议，我们设计了手柄与单片机的硬件通讯电路，编写了通讯程序。为了调试方便，我们通过LED的亮灭来测试手柄与单片机的通讯。加电后发现发光二极管没有任何反应。测量单片机的输出信号发现信号比较稳定但太微弱，我们怀疑是单片机输出的驱动能力不够，致使手柄接收不到单片机的控制信号。于是我们拆开手柄，分析了手柄的线路，查清各个信号线，把从单片机到手柄的信号线即COMMEND,ATT和CLOCK三根线断开，在中间增加一级驱动，重新试验，LED有了反应，说明我们的判断是正确的。

但是LED的亮灭并不是象预想的那样随按键的变化而变化，所以还存在问题。利用仿真软件我们检查了程序执行过程中通讯参数的变化情况。通过程序中存储器的变化可以看出单片机可以正确的送给手柄信号，但是单片机不能正确地接收手柄的信号。仔细分析程序，研究手柄的通讯时序图，修改了单片机的接收子程序。

修改程序之后LED的显示有的时候与按键符合，有的时候不符合。怀疑是通讯时钟信号的问题。手柄与单片机的通讯实时性要求不是很高，但要正确识别按键信号须准确控制时钟同步信号。通过不断调整时钟参数，单片机终于能够准确的接收到手柄的按键信号了。最终将程序一改为程序二



**手柄与单片机控制源程序**

#include<reg52.h>

#define  uchar unsigned char

#define  uint  unsigned int

#define  time  100

uchar  HAND;

uchar  keybuf0;  //手柄按键编码存储单元

uchar  keybuf1;

uchar  RES[6];

sbit  ATT=P3^0;   //手柄控制信号

sbit  CLK=P3^1;

sbit  CMND=P3^2;

sbit  DAT=P3^3;

void main()

 {

    motor(1,0);   //初始化各个功能电机

    motor(2,0);

    motor(3,0);

    motor(4,0);

    motor(5,0);

    WatchDog\_state=0;   //清看门狗

    while(1)

    {

     WDO=0;

     delay\_1ms(1);

     WDO=1;

     key\_scan();

     WDO=0;

     delay\_1ms(1);

     WDO=1;

     if(auto\_reset\_flag)

        keybuf1=0xFE;

     if(auto\_catch\_flag)

        keybuf1=0xFB;

     key\_function();    //各个按键对应的功能

    }

}

    void key\_scan()//键扫描

{

    t=0;

    ATT=0;         //主机读手柄先拉低ATT

    HAND=0x01;     //主机发送开始命令

    psinout();     //0xff

    delay(5);

    HAND=0x42;     //主机发送请求数据命令

    psinout();     //0x41：手柄返回请求应答信号

    delay(25);

    psout();       //0x5A

    delay(25);

    psout(); //keybuf0(空0xff)  //手柄返回按键编码第一字节

    delay(25);

    psout(); //keybuf1(空0xff)  //手柄返回按键编码第二字节

    keybuf0=RES[3];

    keybuf1=RES[4];

    delay(10);

    ATT=1;

}

    void psout()//主机接收子程序

{

    int j,k;

    unsigned char duf=0;

    j=1;

    for(k=0;k<=7;k++)       //逐位发送

    {

       CLK=1;

       delay(5);

       CLK=0;

       delay(5);

       if(DAT==1)

          duf=duf+j;

       j=j\*2;

       CLK=1;

       delay(5);

     }

    RES[t++]=duf;

}

  void psinout()//手柄发送子程序

{

    uchar buf,duf=0;

    uchar i,j=1;

    buf=HAND;

    for(i=0;i<=7;i++)     //逐位接收

    {

     CLK=1;

     delay(5);

     if(buf&0x01)

        CMND=1;

     else

        CMND=0;

     buf=buf>>1;

     CLK=0;

     delay(5);

     if(DAT==1)

        duf=duf+j;

     j=j\*2;

     CLK=1;

     delay(5);

    }

    RES[t++]=duf;

}

    void delay(uchar k)

{

   uchar i;

   for(i=0;i<k;i++);

}

void key\_function()

{

……

……

}

void delay\_1ms(uchar k)

{

   do

   {

    TMOD=0x01;

    TH0=-(300/256);

    TL0=-(300%256);

    TR0=1;

    while(!TF0);

    TF0=0;

   }while (k--);

}

**附** **手动机器人部分图片**

