

# 微机同步数据传输的实现

张 拓 淮北职业技术学院(235000)

## Abstract

This paper introduces a kind of circuit and software brief introduction with 8273 as core that realizes the at the same time data transmission of personal computer.

**Keywords:** 8273, HDLC, synchronism

## 摘 要

本文介绍一种以 8273 为核心, 实现微机同步数据传输的电路及软件简介。

**关键词:** 8273, HDLC, 同步

微机的数据传输方式有串行和并行两种形式, 在用于通讯时主要采用的是串行方式, 串行通讯由异步和同步两种基本工作方式, 在大多数场合使用的是串行工作方式, 而且绝大多数微机配置中提供的也是异步接口, 即 COM1 和 COM2。虽然异步方式的硬件造价和软件实现较同步方式要简单和容易, 但是其传输效率和速度却不如同步传输, 因此在有实时性要求的场合采用同步方式要比异步方式优越得多。本文介绍一种实用的同步数据传输方法。

## 1 硬件电路

本电路主要由 8255, 8253, 8273 构成, 电路的核心元件是 Intel 8273, 它是专为 HDLC 协议设计的通讯控制器, 其主要特点是: 支持 HDLC 的所有规定; 可用于全双工、半双工及 HDLC 环路操作; 可与 modem 接口; 操作上支持帧级操作。8255 用其 PA、PB、PC 口来控制读取信号, 主要是内/外时钟切换、DMA 和 INT 输出允许及 8273 的复位。8253 用于产生内发的同步收/发时钟, TxC 和 RxC 可有两种方式产生, 即由 8253 产生或者由外部 DCE(modem)送入, 两种方式可以通过 8255 控制 74ls51 的切换来实现。电路图见图 1。本电路可以工作在查询、DMA、中断三种方式。

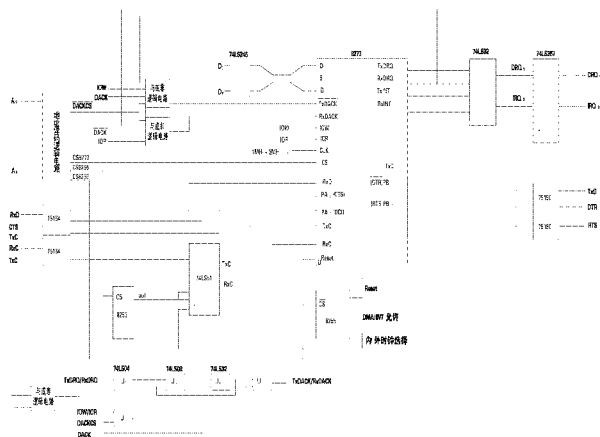


图 1 硬件电路图

## (1) 查询方式

用 8255 关闭 IRQ3 和 DRQ1 的输出, CPU 查询 8273 状态寄存器, 当有 TxINT=1 且 TxIRA=0 时, 说明有数据发送, 用 8255 是 IxDACK 有效并把数据写入发送寄存器。接收过程也一样。

## (2) DMA 方式

在写入命令和参数后, 要进行发送或者接收帧时, 8273 通过 TxDRQ 或者 RxDRQ 项系统发出 DRQ1, CPU 响应后 DMA 控制器发出 DACK=0, IOW 或者 IOR 经过一与或非逻辑产生 TxDACK 或者 RxDACK, 这样就实现了 8273 收发寄存器与 DMA 控制器件的读取。

## (3) 中断方式

在写入命令和参数后, 要进行发送或者接收帧时, 8273 通过 TxINT 或者 RxINT 向系统发出 IRQ3, CPU 响应中断, 执行中断服务子程序, 子程序利用译码电路产生片选(DACKCS)和 CPU 发出 IOW 或者 IOR 经过一与或非逻辑产生 TxDACK 或者 RxDACK, 这样就实现了 CPU 对 8273 收发寄存器进行读写。

## 2 软件编程

本电路可以编写三种方式的通讯程序, 下面是用是查询方式收发程序的实例。中断方式程序和 DMA 方式在本质上是相近的, 就不再列出。

### 发送程序

```
#include<dos.h>
#include<conio.h>
#include<stdio.h>
#define sbuffer***
#define sendresult***
#define statcomm***
#define para***
void init53-55(void)
/* 用于初始化 8255 和 8253, 在此省略/
void init8273(void)
int initpara[4]=0x7f, 0x01, 0xf8;
int initcomm[4]=0x64, 0x97, 0x91, 0x60;
int i=0
```

```

do while((inportb(statcomm)&0x80==0x80);
    outportb(statcomm,initcomm[i];
    while((inportb(statcomm)&0x20==0x20);
        outportb(para,initpara[i]);
        i++;
while (i!=4);
void sendcomm(void)
    int sendpara[2]=0x3c,0x00;
    int i=0;
    do while((inportb(statcomm)&0x80==0x80);
        outportb(statcomm,0xc8);
        while(inportb(statcomm)&0x20==0x20);
        outportb(para,sendpara[i]);
        while(i!=2);
void sending(void)
    int * p,sresult,stop=0;
    int data[60]=0xaa,0x60,.....;
    p=data;
    do while((inportb(statcomm)&0x04)!=0x04);
        if((inportb(statcomm)&0x01)!=0x01)
            outportb(sbuffer,*p++);
        else result=inportb(sendresult);
            stop=1;
            while (stop!=1);
main()
.....
.....
init5355()
init8273()
sendcomm()
sending()
.....
.....
接收程序
#include<dos.h>
#include<conio.h>
#include<stdio.h>
#define sbuffer * * *
#define sendresult * * *
#define statcomm * * *
#define para * * *
voidinit53-55(void)
    /* 用于初始化 8255 和 8253,与发送程序同在此省略/
voidinit8273(void)

```

(上接第 39 页)

在变频器的输出电压中,含有高频尖峰浪涌电压。这些高次谐波冲击电压将使电动机绕组的绝缘强度降低,尤其以 PWM 控制型变频器更为明显,应采取以下措施。

1) 尽量缩短变频器到电机的配线距离。

2) 采用阻断二极管的浪涌电压吸收装置,对变频器输出电压进行处理。

3) 对 PWM 型变频器应尽量在电机输入侧加装图 5a 所示的滤波器。图 5b 中无滤波器使输出电压上升

```

int initpara[4]=0x7f,0x01,0xf8;
int initcomm[4]=0x64,0x97,0x91,0x60;
int i=0;
do while((inportb(statcomm)&0x80==0x80);
    outportb(statcomm,initcomm[i];
    while((inportb(statcomm)&0x20==0x20);
        outportb(para,initpara[i]);
        i++;
while (i!=4);
void recvcomm(void)
    int recvpara[2]= 0xff,0xff;
do while((inportb(statcomm)&0x80==0x80);
    outportb(statcomm,0xc0);
    while((inportb(statcomm)&0x20==0x20);
        outportb(para,recvpara[i]);
        i++;
while (i!=2);
void recving(void)
    int * p,rresult,stop=0;
    int data[1000];
    p=data;
    do while((inportb(statcomm)&0x08)!=0x08);
        if((inportb(statcomm)&0x02)!=0x02)
            *p++ = inportb(rbuffer);
        else rresult=inportb(recvresult);
            stop=1;
            while (stop!=1);
main()
.....
.....
init5355()
init8273()
recvcomm()
recving()
.....
.....

```

无论哪种工作方式都是基于硬件向 8273 发送正确的命令序列从而完成三步工作过程。以上关于 HCLC 通讯的软硬件设计,运行正确可靠。

## 参考文献

- 1 马宏杰,等.微机通讯原理与实用技术.清华大学出版社

[收稿日期:2003.4.24]

沿有明显冲击电压,容易造成电机绝缘损伤。

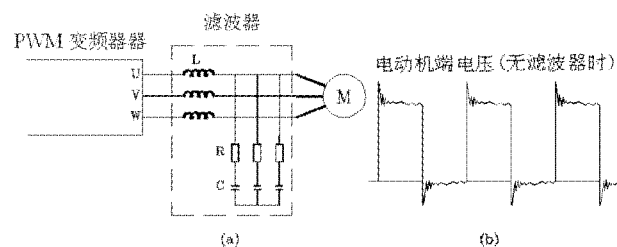


图 5 滤波器

[收稿日期:2003.3.26]