通过 CH365 芯片实现 PC 机与单片机之间的数据互传

1. 实现过程概述

PC 机通过 CH365 与单片机之间进行数据交换，按数据交换速度和硬件实现成本可以分为三类情况。一是以数据块为单位进行数据交换的高速传输，可以在 CH365 与单片机之间增加双口 RAM 或者双缓冲接口芯片 CH421 等实现；二是以字节为单位进行数据交换的中速传输，可以在 CH365 与单片机之间增加 I/O 扩展芯片 8255 或者双缓冲接口芯片 CH421（在两个方向分别提供 64 字节的缓冲）等实现；三是以位数据交换为主的低速传输，可以用 CH365 提供的两线串行接口或者以软件模拟的三线串行接口实现，而不需要增加额外的硬件成本。

本例介于二类和三类之间，主要针对低速的数据交换提供低成本的应用参考，基本原理是将一个字节分成两个 4 位数据进行传输，而不需要增加任何额外的硬件成本。CH365 在本地端提供了通用的 8 位数据总线和可以独立控制输出的 6 位地址信号线 A15-A10，本例使用 CH365 的 A15-A10 作为输出，由 PC 机控制；D7-D0 作为输入，由单片机控制，其中 D6-D4 未定义。

CH365 的 A15 作为 PC 机对单片机的中断控制信号。将 A15 置为低电平，则触发单片机的中断，等待单片机响应。在单片机响应中断后，将 A14 作为判断数据上传下传的标志信号。如果 A14 为低电平，则为数据上传；如果 A14 为高电平，则为数据下传。

在数据下传过程中，A14 作为下传数据高 4 位或低 4 位的指示信号，A13-A10 作为数据位，D7 作为单片机对 PC 机的应答信号。如果 A14 为 1，则 A13-A10 上的数据作为下传数据的低 4 位；如果 A14 为 0，则 A13-A10 上的数据作为下传数据的高 4 位。

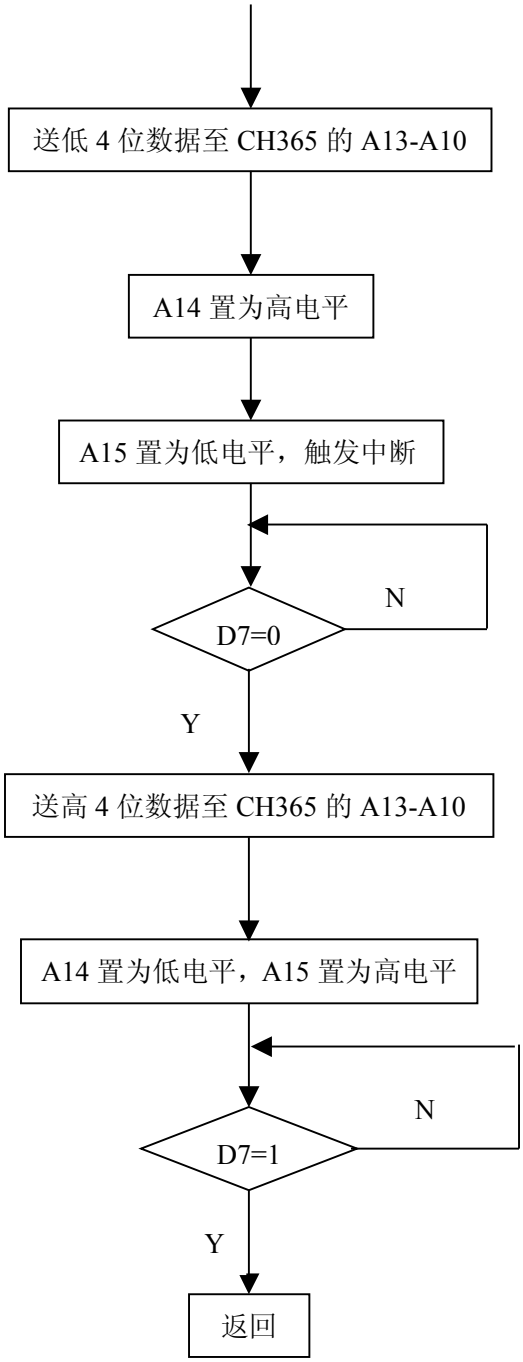
在数据上传过程中，D7 作为上传数据高 4 位或低 4 位的指示信号，D3-D0 作为数据位，A14 作为 PC 机对单片机的应答信号。如果 D7 为 0，则 D3-D0 上的数据作为上传数据的低 4 位；如果 D7 为 1，则 D3-D0 上的数据作为上传数据的高 4 位。

注意：设计 WINDOWS 下的应用程序,CH365 提供了 CH365DLL.H 和 CH365DLL.LIB 两个文件;对于一般的 VC 编译环境,将上述两个文件放置在应用程序的同一目录下,然后将 CH365DLL.LIB 添加到应用程序的链接库中,当应用程序被编译时就会自动链接。如果程序只在 DOS 下使用，请使用 DOS 库程序 CH365DOS.H 代替 WINDOWS 下的库程序。

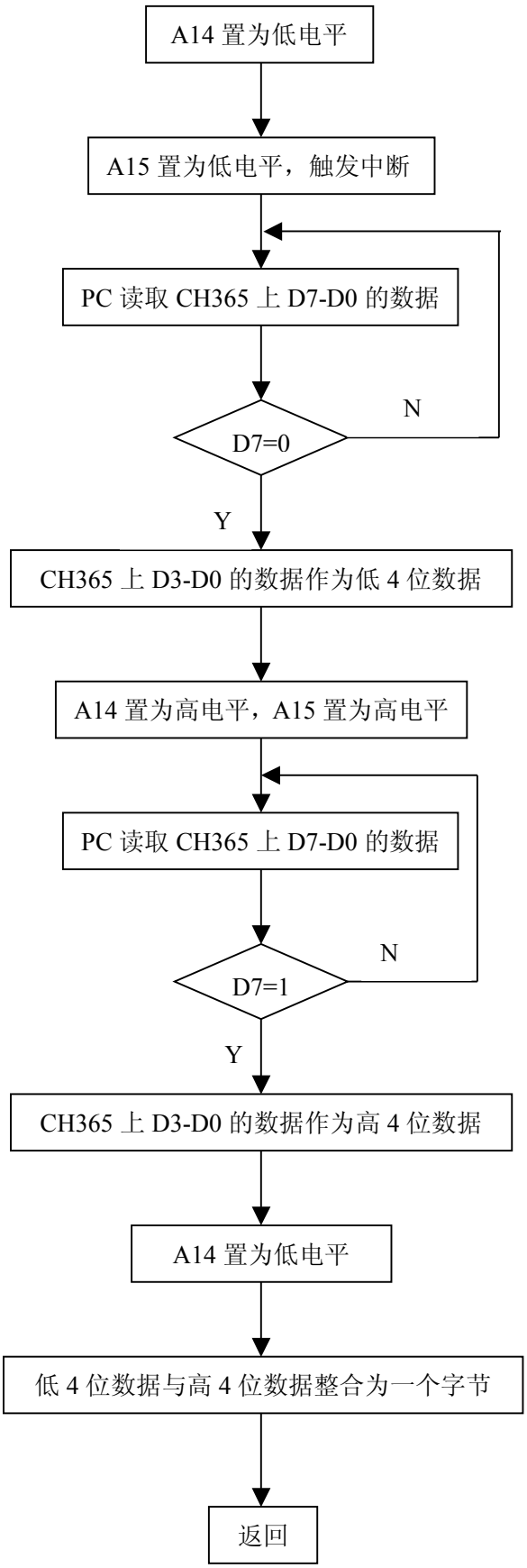
2. 流程图

流程图Ⅰ（PC 机程序）

下传子程序流程图（PC 机程序）

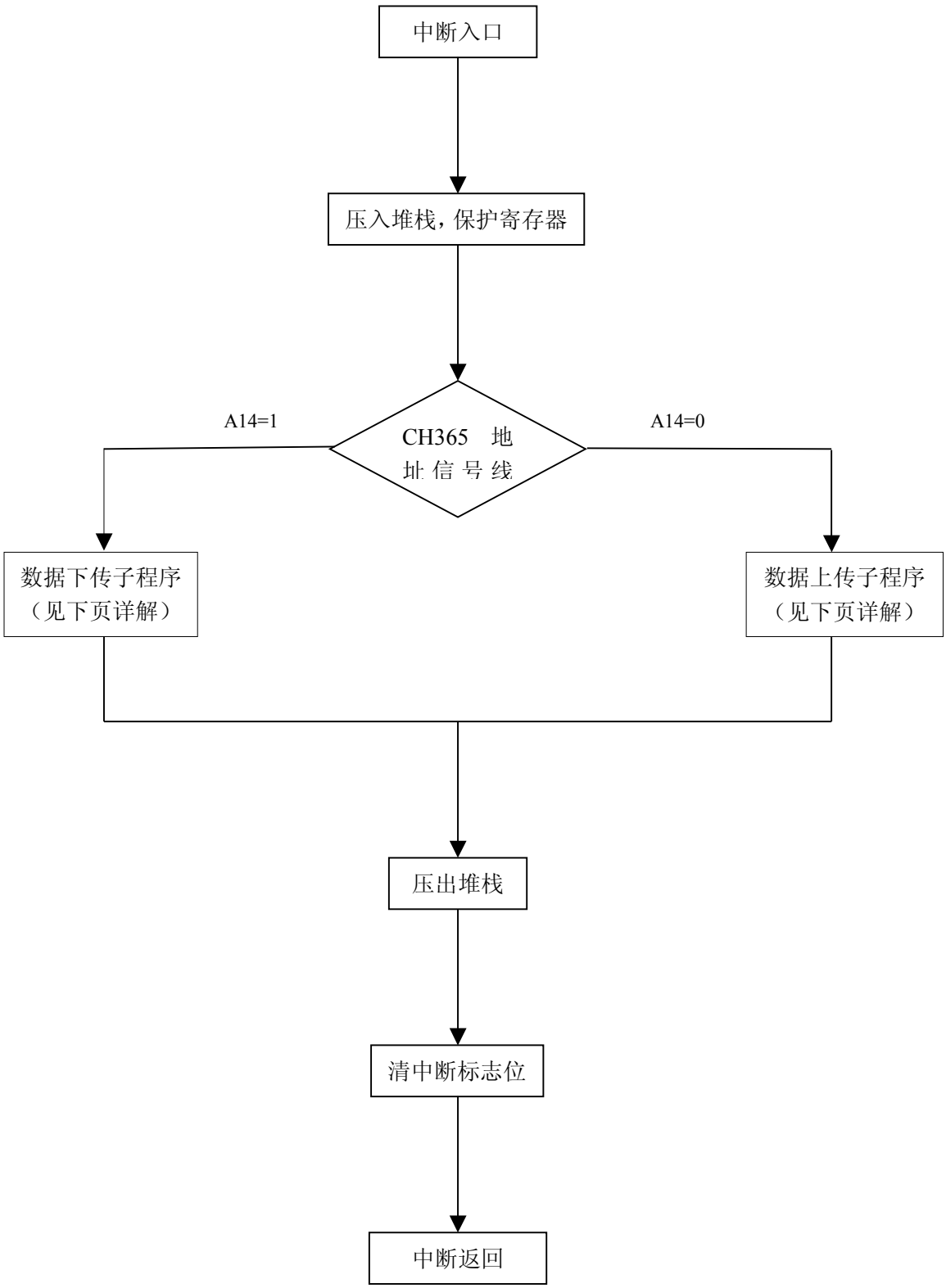


上传子程序流程图（PC 机程序）

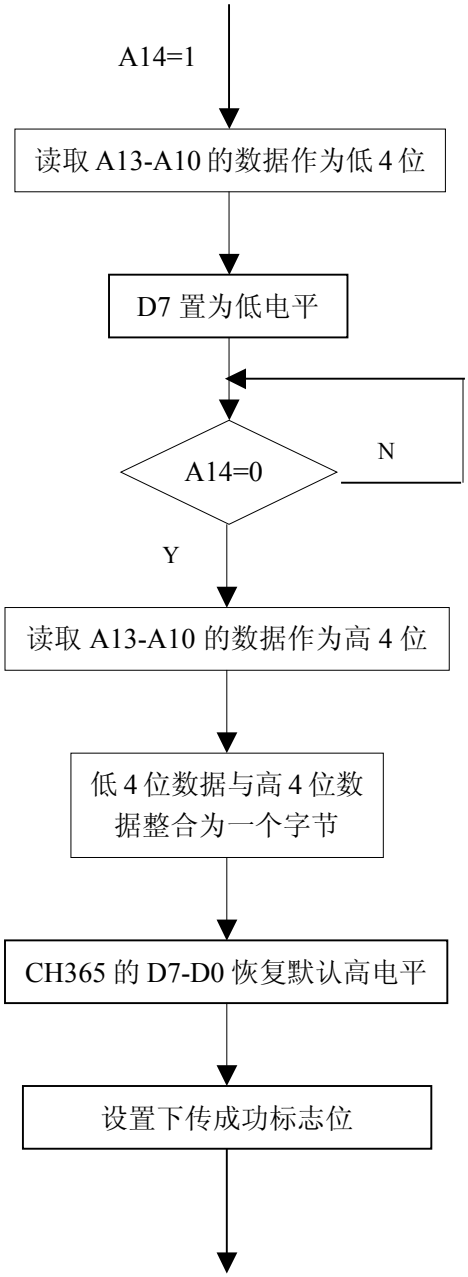


流程图Ⅱ（单片机程序）

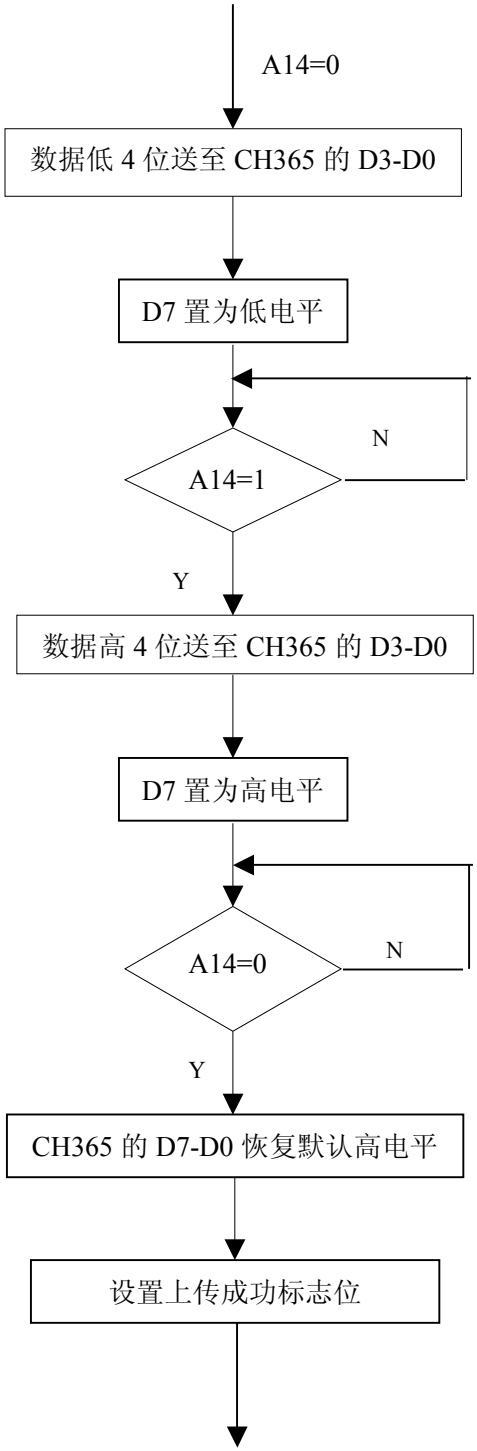
中断程序流程图（单片机程序）



下传子程序流程图（单片机程序）

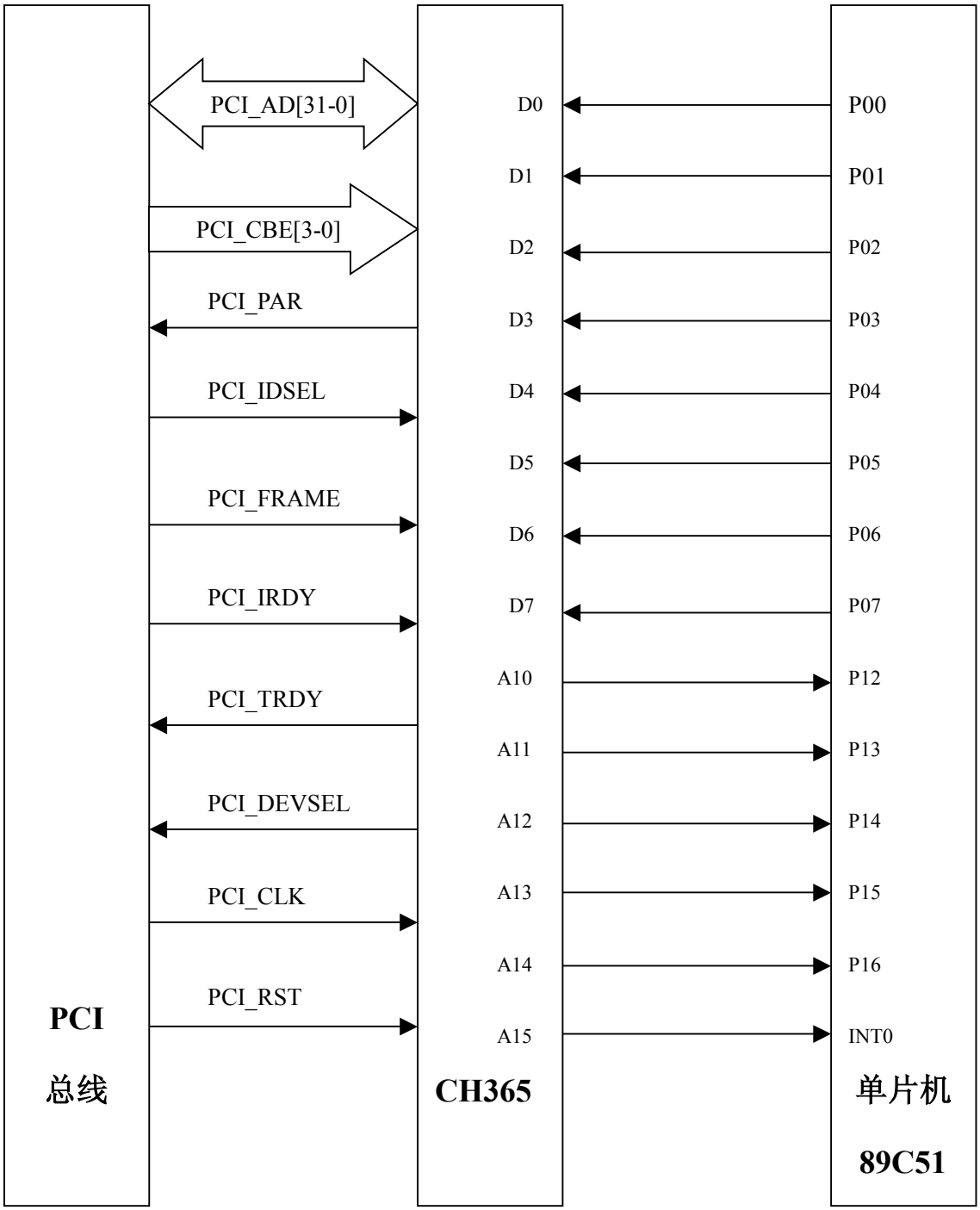


上传子程序流程图（单片机程序）



3. 源程序参考

PC 机通过 CH365 与 MCS-51 单片机的接线图



① PC 机程序源代码

```
#include    <windows.h>
#include    <stdlib.h>
#include    <stdio.h>
#include    <conio.h>
#include    <winioctl.h>
#include    "CH365DLL.H"    // 如果在 DOS 下使用，请使用 DOS 库
#define   DIN_PORT   0xef   // 数据输入端口，如果没有其它 I/O 应用则可以为 0 至 0xEF

mPCH365_IO_REG  mIoBase;           // I/O 基址
UCHAR  Read_data( );               // 读（上传）一个字节数据的子程序
void   Write_data( UCHAR data );   // 写（下传）一个字节数据的子程序

void main() {    // 演示用主程序
    UCHAR data,Input_data;
    int i;
    // 需要使用 DLL 则需要先加载
    if ( LoadLibrary( "CH365DLL.DLL" ) == NULL ) return;  // 加载 DLL 失败
    //打开 CH365 设备
    if ( CH365OpenDevice( FALSE, FALSE ) == INVALID_HANDLE_VALUE ) return;
    CH365GetIoBaseAddr( &mIoBase );
    // 使用系统为 CH365 自动分配的 I/O 基址
    for ( i=0; i<=100; i++ )  {   // 测试
          Input_data=rand()%0x0100;  // 随机数 0-255
          printf("*****WRITE DATA=%2x   ",Input_data);
          Write_data(Input_data);
// 因为 MCS-51 单片机速度较慢，最好做少量延时，等待单片机处理一个字节的数据
          data=Read_data();
// 因为 MCS-51 单片机速度较慢，最好做少量延时，等待单片机处理一个字节的数据
          printf("*****READ DATA=%2x   ",data);
          if(data==Input_data) printf("*****test correct!\n");
          else printf("*****test wrong!\n");
    }
    CH365CloseDevice( );
}

// 从单片机读取数据（上传）
UCHAR  Read_data( )
{
   UCHAR  low, mByte;
   CH365SetA15_A8( 0x00 ) ;  // A15=0 触发单片机中断，A14=0 表示要上传
   while (1) {
       CH365ReadIoByte(&mIoBase->mCH365IoPort[DIN_PORT], &mByte);  // 得到 D0-D7 的数据
       if ( (mByte & 0x80) == 0 ) break;  // 判断 D7
```

```
    }  // 如果 D7=0 则单片机已经中断并且送出低 4 位数据，否则继续等待单片机
   low = mByte & 0x0f;   // 先收到低 4 位数据
   CH365SetA15_A8( 0xc0 ); // 置 A15 为高电平，置 A14 为高电平通知单片机
   while (1) {
       CH365ReadIoByte(&mIoBase->mCH365IoPort[DIN_PORT], &mByte);  // 得到 D0-D7 的数据
       if ( (mByte & 0x80) == 0x80 ) break;  // 判断 D7
   }  // 如果 D7=1 则单片机已经送出高 4 位数据，否则继续等待单片机
   CH365SetA15_A8( 0x80 );           //置 A14 为低电平，通知单片机结束
   mByte=( mByte & 0x0f ) << 4;      //保存 4 位数据,左移 4 位作为高 4 位数据
   return( mByte | low ) ;           //高 4 位和低 4 位组合成一个字节
}

// 向单片机写入数据（下传）
void  Write_data( UCHAR data )
{
    UCHAR mByte;
    CH365SetA15_A8( ( (data<<2) & 0x3c ) | 0x40 );
// 输出低 4 位数据，置 A14 为高电平表示下传，置 A15 为低电平触发单片机中断
    while (1) {
       CH365ReadIoByte(&mIoBase->mCH365IoPort[DIN_PORT], &mByte);  // 得到 D0-D7 的数据
       if ( (mByte & 0x80) == 0 ) break;  // 判断 D7
    }  // 如果 D7=0 则单片机已经中断并且收取了 4 位数据，否则继续等待单片机
    CH365SetA15_A8( ( (data>>2) & 0x3c ) | 0x80 );
// 输出高 4 位数据给单片机，A15 为高电平，A14 为低电平通知单片机收取另 4 位数据
    return;
}
```

②单片机程序源代码(MCS-51)

```
;单片机与 CH365 接口演示程序.
;功能：通过 CH365 芯片实现了计算机对单片机的读和写操作.
;用到的寄存器：ACC
CH365_UP       DATA P0                ;定义 P0 口为数据上传
CH365_DOWN     DATA P1                ;定义 P1 口为数据下传
CH365_A14      BIT  P1.6              ;定义为判断位
CH365_UP_OK    BIT  20H.0             ;表示上传成功的位变量
CH365_DOWN_OK  BIT  20H.1             ;表示下传成功的位变量
UP_BYTE        DATA 30H               ;定义上传数据保存的地址
DOWN_BYTE      DATA 31H               ;定义下传数据保存的地址

    org 0000H                         ;主程序入口
    sjmp MAIN
    org 0003H                         ;INT0 入口
    sjmp JUDGE                        ;中断读、写判断
```

```
MAIN:
        CLR CH365_UP_OK
        CLR CH365_DOWN_OK
        SETB EX0
        SETB IT0                        ;边沿触发或者电平触发
        SETB EA                         ;中断允许
SEEK:
        JBC CH365_UP_OK ,NEXT_UP        ;上传成功后将该位清 0 并且转移
        JBC CH365_DOWN_OK ,NEXT_DOWN    ;下传成功后将该位清 0 并且转移
        AJMP SEEK                       ;循环执行此过程
NEXT_UP:
        AJMP SEEK                       ;用户可以在此定义下一次要上传的数据
NEXT_DOWN:
        MOV A,DOWN_BYTE
        MOV UP_BYTE,A                   ;下传数据送给上传数据单元
        AJMP SEEK                       ;用户可以在此定义下一次要下传的数据

JUDGE: 单片机中断程序入口
        PUSH PSW
        PUSH ACC                        ;保存寄存器
        JB CH365_A14,CH365_DOWNDATA     ;判断 A14=1 跳到写入数据子程
        JMP  CH365_UPDATA               ;A14=0 跳到读数据子程
;中断子程序(读)
CH365_UPDATA:
        MOV A,UP_BYTE                   ;选定上传数据
        ANL A,#0FH                      ;保留低四位,置 D7 位为 0
        MOV CH365_UP,A                  ;把 A 上的低四位数据送到 P0 口
        JNB CH365_A14,$                 ;等待 A14=1
;正式发布的程序请在等待中加入超时计数,避免一直等待,
;因为如果中途 PC 机及 CH365 被复位,则单片机有可能永远等待
        MOV A,UP_BYTE                   ;选定上传数据入口
        RR  A
        RR  A
        RR  A
        RR  A                       ;循环右移 4 次
        ANL A,#0FH                  ;这是高 4 位数据
        ORL A,#80H                  ;D7 位为 1
        MOV CH365_UP,A              ;把高四位数据送到 P0 口
        JB CH365_A14,$              ;等待 A14=0
;正式发布的程序请在等待中加入超时计数,避免一直等待,
;因为如果中途 PC 机及 CH365 被复位,则单片机有可能永远等待
        MOV CH365_UP,#0FFH          ;P0 口全置为 1,D7=1
        SETB CH365_UP_OK            ;上传成功
        CLR IE0
        POP ACC
```

```
        POP PSW                     ;恢复寄存器
        RETI
;中断子程序(写)
CH365_DOWNDATA:
        MOV A,CH365_DOWN        ;读取 P1 口上的数据送到 A
        RR A
        RR A                        ;A13-A10 是数据
        ANL A,#0FH
        MOV DOWN_BYTE,A         ;保存收到的低 4 位数据
        MOV CH365_UP,#7FH       ;P0 口输出,置 D7 位为 0
        JB CH365_A14,$          ;等待 A14=0
;正式发布的程序请在等待中加入超时计数,避免一直等待,
;因为如果中途 PC 机及 CH365 被复位,则单片机有可能永远等待
        MOV A,CH365_DOWN        ;读取 P1 口上的数据送到 A
        RL A
        RL A
        ANL A,#0F0H             ;保留高四位
        ORL DOWN_BYTE,A         ;低 4 位和高 4 位组合为一个字节的数据
        MOV CH365_UP,#0FFH      ;P0 口输出全为 1,置 D7 位为 1
        SETB CH365_DOWN_OK      ;下传成功
        CLR IE0
        POP ACC
        POP PSW                     ;恢复寄存器
        RETI
```