**红外感应计时器**



图 1:实物图

**1、简要说明：**

一、 尺寸：43.7cm \*27.0cm \*39.5cm 长\*宽\*高。

二、 主要芯片：博光3-MNK对射式红外感应装置 \*1，CD4543 \*4, 89C52。

三、 工作电压：数码管8.2v-9.6v 51单片机：5v。

四、 特点：

1、可接电脑同步显示时钟计时器。

2、可屏蔽极短时间内红外多次接收到信号

五、应用：

适用于模型车比赛计时等。

**2、传感器说明：**

**2.1博光3-MNK对射式红外感应装置模块连接说明：**

**如图所示：**



图2

红线接单片机上正极

黑线接单片机上负极

白线接p2.2 作为中断控制，当感应装置中间没有障碍时恒从白线返回一个高电平，否则返回低电平

**不可在通电状态下将红（黑）线与白线短接，否则感应器将毁坏**

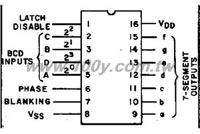
**2.2 4英寸共阴LED数码管**

Abdskaf\\\\

每一段的电压在8v到11v，小数点的1.5v到3v，小数点极易烧坏需要注意

**2.3CD4543BE**

详细建议参考百度文库，在这里只介绍其在电路中的搭接，芯片引脚图如下



这里的a-g 对应于数码管的a-g段。芯片连接见原理图。

**2.2原理图**

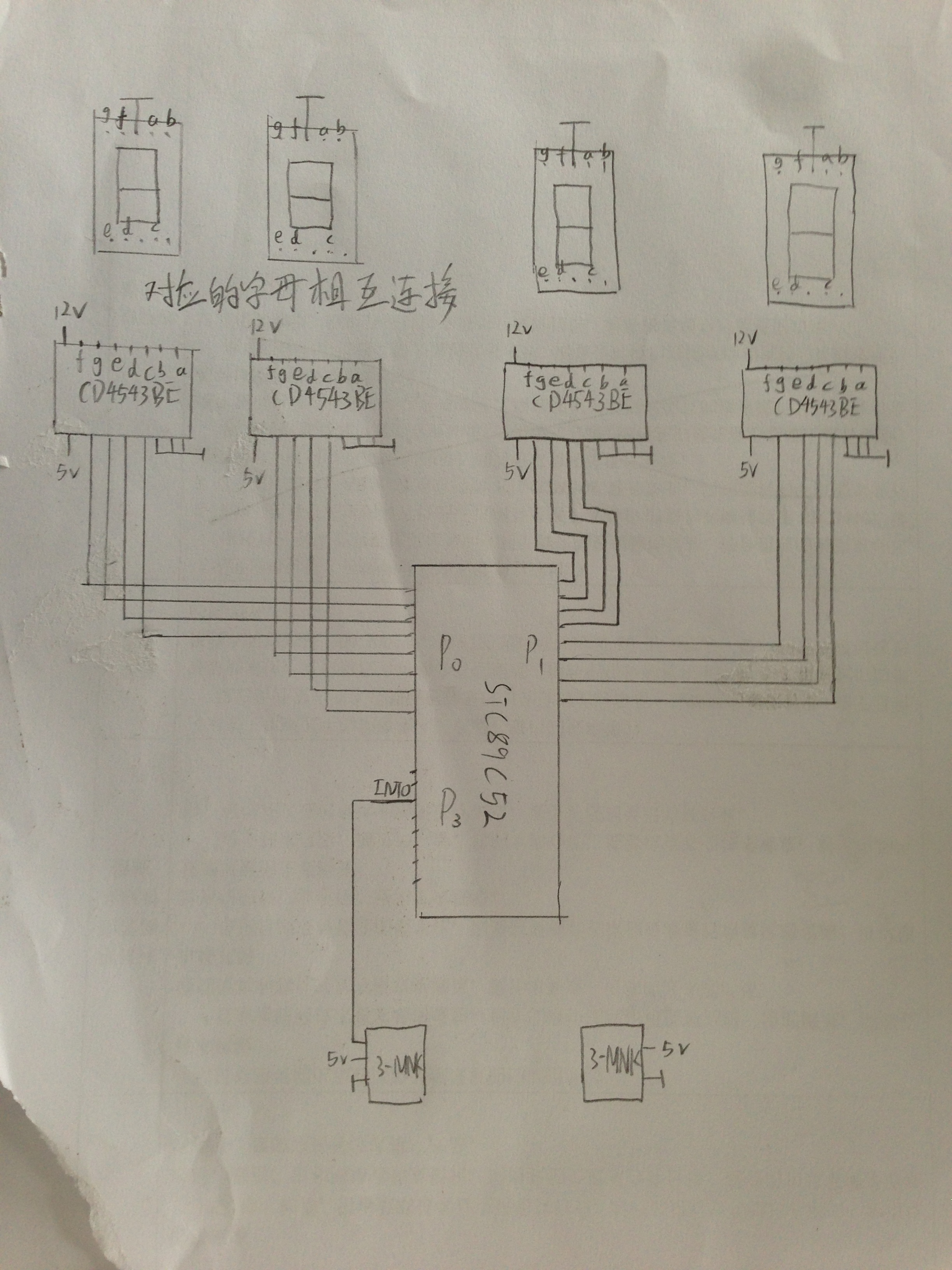
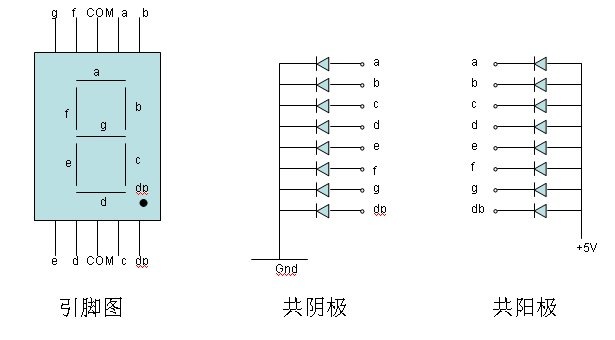


图 5：原理图

**2.3测试程序**

**#include <reg52.h>**

**#include <intrins.h>**

**#include <stdlib.h>**

**//unsigned char code dula[] = {0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f,0x80};**

**sbit a = P2^7;**

**sbit b = P2^6;**

**unsigned int numBit[4], iCount;**

**unsigned int num1 = 0;**

**unsigned char ch;**

**unsigned int sig = 0;**

**void delay(unsigned int n)**

**{**

**unsigned int a, b;**

**for(a = n; a > 0; a--)**

**for(b = n; b > 0; b--);**

**}**

**void init()**

**{**

**//interrupt enable**

**EA = 1;**

**ET0 = 1;**

**ES = 1;**

**//timer 0 for counting, timer 1 for Serial Port**

**TMOD = 0x21;**

**TH0 = (65535 - 10000) / 256;**

**TL0 = (65535 - 10000) % 256;**

**TH1 = 0xfd;**

**TL1 = 0xfd;**

**TR1 = 1;**

**// TR0 = 1;**

**//interrupt0**

**EX0 = 1;**

**IT0 = 1;**

**//reset control**

**a = 1;**

**b = 0;**

**//SCON**

**SM0 = 0;**

**SM1 = 1;**

**REN = 1;**

**}**

**void main()**

**{**

**init();**

**while(1)**

**{**

**if(sig)**

**{**

**sig++;**

**if(sig == 200)**

**{**

**sig = 0;**

**}**

**}**

**if(num1 > 9999)**

**{**

**num1 = 0;**

**// return ;**

**}**

**if(a == b || ch == 'r')//串口发过来’r’使计时器归零**

**{**

**num1 = 0;**

**TR0 = 0;**

**delay(2);**

**SBUF = 2;**

**ch = 0;**

**}**

**if(ch == 's')// 串口发过来’s’使计时器终止/运行**

**{**

**TR0 = ~TR0;**

**ch = 0;**

**}**

**numBit[0] = num1 / 1000;**

**numBit[1] = (num1 % 1000) / 100;**

**numBit[2] = (num1 % 100) / 10;**

**numBit[3] = (num1 % 10);**

**P0 = ((numBit[0] << 4)| numBit[1]) & 0x00ff;**

**P1 = ((numBit[2] << 4)| numBit[3]) & 0x00ff;**

**}**

**}**

**void AnswerTIME0() interrupt 1// using 1**

**{**

**TR0 = 0;**

**TH0 = (65535 - 9216) / 256;**

**TL0 = (65535 - 9216) % 256;**

**num1++;**

**SBUF = 1;//给串口发1给电脑，数字加以1**

**// delay(1);**

**TR0 = 1;**

**}**

**void AnswerINT0() interrupt 0// using 1**

**{**

**EX0 = 0;**

**if(P3^2 == 0)**

**{**

**delay(3);**

**if(P3^2 == 0 && sig == 0)**

**{**

**sig = 1;**

**TR0 = ~TR0;**

**}**

**}**

**EX0 = 1;**

**}**

**void ser() interrupt 4**

**{**

**if(RI == 1)**

**{**

**RI = 0;**

**ch = SBUF;**

**}**

**}**

**另有电脑运行程序：**

**Main.cpp:**

#include <windows.h>

#include <stdio.h>

#include"serailAPI.h"

#include <conio.h>

#define ID\_TIMER 1

LRESULT CALLBACK WndProc (HWND, UINT, WPARAM, LPARAM) ;

char ikey, okey;//ikey: input from 51; okey date(8 bit) to 51

int time = 0;

int WINAPI WinMain (HINSTANCE hInstance, HINSTANCE hPrevInstance,

PSTR szCmdLine, int iCmdShow)

{

static TCHAR szAppName[] = TEXT ("Clock") ;

HWND hwnd;

MSG msg;

WNDCLASS wndclass ;

//

wndclass.style = CS\_HREDRAW | CS\_VREDRAW ;

wndclass.lpfnWndProc = WndProc ;

wndclass.cbClsExtra = 0 ;

wndclass.cbWndExtra = 0 ;

wndclass.hInstance = hInstance ;

wndclass.hIcon = NULL ;

wndclass.hCursor = LoadCursor (NULL, IDC\_ARROW) ;

wndclass.hbrBackground = (HBRUSH) GetStockObject (WHITE\_BRUSH) ;

wndclass.lpszMenuName = NULL ;

wndclass.lpszClassName = szAppName ;

if (!RegisterClass (&wndclass))

{

MessageBox (NULL, TEXT ("Program requires Windows NT!"),

szAppName, MB\_ICONERROR) ;

return 0 ;

}

hwnd = CreateWindow (szAppName, TEXT ("MCU Clock"),

WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, CW\_USEDEFAULT,

CW\_USEDEFAULT, CW\_USEDEFAULT,

NULL, NULL, hInstance, NULL) ;

ShowWindow (hwnd, iCmdShow) ;

UpdateWindow (hwnd) ;

while(TRUE){

if(PeekMessage(&msg,NULL,0,0,PM\_REMOVE))//检测是否有消息 有的话处理 否则进行else的内容

{

if(msg.message == WM\_QUIT)

break;

TranslateMessage(&msg);

DispatchMessage(&msg);

}

else

{

InvalidateRect(hwnd, NULL, FALSE); //////

SendMessage(hwnd, WM\_PAINT,0, 0);

// Sleep(3);

}

}

return msg.wParam ;

}

LRESULT CALLBACK WndProc (HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam)

{

HDC hdc ;

static HFONT font;

static int cxClient, cyClient;

char numBit[4];

static HDC hdcBack;

static HBITMAP hBack;

static char sbuf[6];

PAINTSTRUCT ps;

switch (message)

{

case WM\_CREATE:

hdc = GetDC(hwnd);

// SetTimer(hwnd, ID\_TIMER, 2, NULL);

hdcBack = CreateCompatibleDC(hdc);

SerailInit();

return 0 ;

case WM\_SIZE:

cxClient = LOWORD(lParam);

cyClient = HIWORD(lParam);

font=CreateFont(cyClient - 10,cxClient/5,0,0,0,0,0,0,GB2312\_CHARSET,0,0,0,0,TEXT("方正舒体"));

hBack = CreateCompatibleBitmap(hdcBack, cxClient, cyClient);

SelectObject(hdcBack, hBack);

return 0;

case WM\_LBUTTONDOWN:

okey = 's';

return 0;

case WM\_RBUTTONDOWN:

okey = 'r';

return 0;

case WM\_PAINT:

ikey = ScanSerail();

if(ikey == 1)

time++;

else if(ikey == 2)

time = 0;

numBit[0] = time / 1000;

numBit[1] = (time % 1000) / 100;

numBit[2] = (time % 100) / 10;

numBit[3] = (time % 10);

sprintf(sbuf, "%d%d:%d%d", numBit[0], numBit[1], numBit[2], numBit[3]);

if(okey == 's')

{

WriteChar((BYTE\*)"s", 1);

okey = 0;

}

else if(okey == 'r')

{

WriteChar((BYTE\*)"r", 1);

okey = 0;

}

hdc = BeginPaint (hwnd, &ps) ;

SelectObject(hdcBack, font);

TextOut(hdcBack, 0, 0, sbuf, strlen(sbuf));

BitBlt(hdc,5,20,cxClient, cyClient, hdcBack, 0, 0, SRCCOPY);

EndPaint (hwnd, &ps) ;

return 0 ;

case WM\_DESTROY:

// KillTimer (hwnd, ID\_TIMER) ;

PostQuitMessage (0) ;

return 0 ;

}

return DefWindowProc (hwnd, message, wParam, lParam) ;

}

**SeriousAPI.h:**

#include <stdio.h>

#include <stdlib.h>

#include <windows.h>

HANDLE hComm;

OVERLAPPED m\_ov;

COMSTAT comstat;

DWORD m\_dwCommEvents;

bool openport(char \*portname)//打开一个串口

{

hComm = CreateFile(portname,

GENERIC\_READ | GENERIC\_WRITE,

0,

0,

OPEN\_EXISTING,

FILE\_FLAG\_OVERLAPPED,

0);

if (hComm == INVALID\_HANDLE\_VALUE)

return FALSE;

else

return true;

}

bool setupdcb(int rate\_arg)

{

DCB dcb;

int rate= rate\_arg;

memset(&dcb,0,sizeof(dcb));

if(!GetCommState(hComm,&dcb))//获取当前DCB配置

{

return FALSE;

}

/\* -------------------------------------------------------------------- \*/

// set DCB to configure the serial port

dcb.DCBlength = sizeof(dcb);

/\* ---------- Serial Port Config ------- \*/

dcb.BaudRate = rate;

dcb.Parity = NOPARITY;

dcb.fParity = 0;

dcb.StopBits = ONESTOPBIT;

dcb.ByteSize = 8;

dcb.fOutxCtsFlow = 0;

dcb.fOutxDsrFlow = 0;

dcb.fDtrControl = DTR\_CONTROL\_DISABLE;

dcb.fDsrSensitivity = 0;

dcb.fRtsControl = RTS\_CONTROL\_DISABLE;

dcb.fOutX = 0;

dcb.fInX = 0;

/\* ----------------- misc parameters ----- \*/

dcb.fErrorChar = 0;

dcb.fBinary = 1;

dcb.fNull = 0;

dcb.fAbortOnError = 0;

dcb.wReserved = 0;

dcb.XonLim = 2;

dcb.XoffLim = 4;

dcb.XonChar = 0x13;

dcb.XoffChar = 0x19;

dcb.EvtChar = 0;

/\* -------------------------------------------------------------------- \*/

// set DCB

if(!SetCommState(hComm,&dcb))

{

return false;

}

else

return true;

}

bool setuptimeout(DWORD ReadInterval,DWORD ReadTotalMultiplier,DWORD ReadTotalconstant,DWORD WriteTotalMultiplier,DWORD WriteTotalconstant)

{

COMMTIMEOUTS timeouts;

timeouts.ReadIntervalTimeout=ReadInterval;

timeouts.ReadTotalTimeoutConstant=ReadTotalconstant;

timeouts.ReadTotalTimeoutMultiplier=ReadTotalMultiplier;

timeouts.WriteTotalTimeoutConstant=WriteTotalconstant;

timeouts.WriteTotalTimeoutMultiplier=WriteTotalMultiplier;

if(!SetCommTimeouts(hComm, &timeouts))

return false;

else

return true;

}

void ReceiveChar( )

{

BOOL bRead = TRUE;

BOOL bResult = TRUE;

DWORD dwError = 0;

DWORD BytesRead = 0;

char RXBuff;

for (;;)

{

bResult = ClearCommError(hComm, &dwError, &comstat);

if (comstat.cbInQue == 0)

continue;

if (bRead)

{

bResult = ReadFile(hComm, // Handle to COMM port

&RXBuff, // RX Buffer Pointer

1, // Read one byte

&BytesRead, // Stores number of bytes read

&m\_ov); // pointer to the m\_ov structure

printf("%c",RXBuff);

if (!bResult)

{

switch (dwError = GetLastError())

{

case ERROR\_IO\_PENDING:

{

bRead = FALSE;

break;

}

default:

{

break;

}

}

}

else

{

bRead = TRUE;

}

} // close if (bRead)

if (!bRead)

{

bRead = TRUE;

bResult = GetOverlappedResult(hComm, // Handle to COMM port

&m\_ov, // Overlapped structure

&BytesRead, // Stores number of bytes read

TRUE); // Wait flag

}

}

}

bool WriteChar(BYTE\* m\_szWriteBuffer,DWORD m\_nToSend)

{

BOOL bWrite = TRUE;

BOOL bResult = TRUE;

DWORD BytesSent = 0;

// HANDLE m\_hWriteEvent;

// ResetEvent(m\_hWriteEvent);

if (bWrite)

{

m\_ov.Offset = 0;

m\_ov.OffsetHigh = 0;

// Clear buffer

bResult = WriteFile(hComm, // Handle to COMM Port

m\_szWriteBuffer, // Pointer to message buffer in calling finction

m\_nToSend, // Length of message to send

&BytesSent, // Where to store the number of bytes sent

&m\_ov ); // Overlapped structure

if (!bResult)

{

DWORD dwError = GetLastError();

switch (dwError)

{

case ERROR\_IO\_PENDING:

{

// continue to GetOverlappedResults()

BytesSent = 0;

bWrite = FALSE;

break;

}

default:

{

// all other error codes

break;

}

}

}

} // end if(bWrite)

if (!bWrite)

{

bWrite = TRUE;

bResult = GetOverlappedResult(hComm, // Handle to COMM port

&m\_ov, // Overlapped structure

&BytesSent, // Stores number of bytes sent

TRUE); // Wait flag

// deal with the error code

if (!bResult)

{

printf("GetOverlappedResults() in WriteFile()");

}

} // end if (!bWrite)

// Verify that the data size send equals what we tried to send

if (BytesSent != m\_nToSend)

{

printf("WARNING: WriteFile() error.. Bytes Sent: %d; Message Length: %d\n", BytesSent, strlen((char\*)m\_szWriteBuffer));

}

return true;

}

void SerailInit()

{

if(!openport("com2"))////////////////////////////////////////////////////////////////////////

MessageBox(NULL, "open comport failure\n", "error", MB\_ICONWARNING);

if(!setupdcb(9600))

MessageBox(NULL, "setupDCB failure\n", "error", MB\_ICONWARNING);

if(!setuptimeout(0,0,0,0,0))

MessageBox(NULL, "setuptimeout failure\n", "error", MB\_ICONWARNING);

PurgeComm(hComm, PURGE\_RXCLEAR | PURGE\_TXCLEAR | PURGE\_RXABORT | PURGE\_TXABORT);

// WriteChar((BYTE\*)"s",1);

}

BYTE ScanSerail()

{

static BOOL bRead = TRUE;

static BOOL bResult = TRUE;

static DWORD dwError = 0;

static DWORD BytesRead = 0;

static char RXBuff;

bResult = ClearCommError(hComm, &dwError, &comstat);

if (comstat.cbInQue == 0)

//continue;

return 0;

if (bRead)

{

bResult = ReadFile(hComm, // Handle to COMM port

&RXBuff, // RX Buffer Pointer

1, // Read one byte

&BytesRead, // Stores number of bytes read

&m\_ov); // pointer to the m\_ov structure

//printf("%c",RXBuff);

return RXBuff;

if (!bResult)

{

switch (dwError = GetLastError())

{

case ERROR\_IO\_PENDING:

{

bRead = FALSE;

break;

}

default:

{

break;

}

}

}

else

{

bRead = TRUE;

}

} // close if (bRead)

if (!bRead)

{

bRead = TRUE;

bResult = GetOverlappedResult(hComm, // Handle to COMM port

&m\_ov, // Overlapped structure

&BytesRead, // Stores number of bytes read

TRUE); // Wait flag

}

// system("pause");

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* program end\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

其他:

我们在这里由于驱动问题和CD4543的芯片特性，对数码管是采用静态控制。如果有人要翻做，可以选择动态控制数码管好像更有意思，用一对三极管驱动一段。面白そう！