

```

/*****
/*          S E R I R Q . C          */
/*-----*/
/* Task      : Uses chat program to demonstrate      */
/*          serial port IRQ programming              */
/*-----*/
/* Author      : Michael Tischer / Bruno Jennrich    */
/* Developed on : 04/08/1994                          */
/* Last update  : 04/07/1995                          */
/*-----*/
/* COMPILER OPT. : Disable stack check!              */
/*          Disable vector check!                    */
/*          Disable ALL optimizations !              */
/* COMPILER      : Borland C++ 3.1, Microsoft Visual C++ 1.5 */
/*****
#include <dos.h>
#include <ctype.h>
#include <process.h>
#include <stdio.h>
#include <conio.h>

#include "types.h"
#include "win.h"
#include "serutil.h"
#include "irqutil.h"
#include "args.h"

/*- Document the following lines within project -----*/
#include "args.c"
#include "win.c"
#include "serutil.c"
#include "irqutil.c"

```

```

/*- Global variables -----*/
WINDOW Screen;
WINDOW Remote;
WINDOW Local;
WINDOW Status;

INT iSerPort;
INT iSerIRQ;

/*****
/* GetSer : Interrupt routine, initiated by serial port */
**-----*/
/* Info : This function accepts data from the serial */
/*         port and displays it in the "remote" window. */
/*         This function also contains requirements for */
/*         all other port events that initiate */
/*         an IRQ. */
*****/
VOID __interrupt _FP GetSer( VOID )
{ static BYTE bIRQID;
  static CHAR c[16 + 1];
  static int i;

  bIRQID = ( BYTE )inp( iSerPort + SER_IRQ_ID );
  if( !( bIRQID & SER_ID_PENDING ) ) /* Is IRQ pending ? */
  {
    switch( bIRQID & SER_ID_MASK )
    {
      case SER_ID_RECEIVED: /* After data received */
        i = 0;
        while( ser_IsDataAvailable( iSerPort ) && ( i < 16 ) )

```

```

        {
            c[ i++ ] = ( BYTE )inp( iSerPort + SER_RXBUFFER );
            c[ i ] = '\0';
        }
        win_Print( &Remote, c );
        break;
        case SER_ID_SENT:                                /* After data sent */
            break;
        case SER_ID_MODEMSTATUS: /* After line status change */
            break;
    }
}
irq_SendEOI( iSerIRQ );                                /* End of interrupt */
}

/*****
/* M A I N   P R O G R A M                                     */
*****/
VOID main( INT argc, PCHAR argv[ ] )
{
    static CHAR c[2];
    static INT iCom;
    static LONG lBaud;
    VOID ( _interrupt _FP *lpOldIRQ )();
    PBYTE pSaved;                                       /* Saved screen */
    INT iUART;

    if( FindString( argv, "?", argc ) )
    {
        printf( "Call:\n" );
        printf( "SERIRQ [-COM:comport] [-BAUD:baudrate]\n" );
        printf( "COM port  = 1 or 2      (Default: 1 )\n" );
    }
}

```

```

    printf("Baud rate = 50 - 115200 (Default: 9600)\n");
    exit(0);
}

if( GetArg( argc, argv, "-COM:", _int, &iCom, 1 ) )
{
    if( iCom == 1 )
    {
        iSerPort = SER_COM1;    /* Only COM1 and COM2 are "standardized" */
        iSerIRQ  = SER_IRQ_COM1;
    }
    else
    if( iCom == 2 )
    {
        iSerPort = SER_COM2;    /* Only COM1 and COM2 are "standardized" */
        iSerIRQ  = SER_IRQ_COM2;
    }
    else
    {
        printf("Unsupported COM port!\n");
        exit(0);
    }
}
else
{
    iSerPort = SER_COM1;    /* Only COM1 and COM2 are "standardized" */
    iSerIRQ  = SER_IRQ_COM1;
}

if( !GetArg( argc, argv, "-BAUD:", _long, &lBaud, 1 ) )
    lBaud = 9600L;          /* Maximum baud rate for UART 8450A */
if( lBaud > SER_MAXBAUD )

```

```

{
    printf("Baud rate too high!\n");
    printf("Maximum: %ld Bd\n", SER_MAXBAUD );
}

iUART = ser_Init( iSerPort, lBaud,
                  SER_LCR_8BITS | SER_LCR_1STOPBIT | SER_LCR_NOPARITY );

if( iUART == NOSER )
{
    printf("No port!\n");
    exit( 0 );
}

if( iUART > INS8250 ) ser_FIFOLevel( iSerPort, SER_FIFO_TRIGGER14 );

win_GetScreenSettings( &Screen );                /* Save screen */
pSaved = win_Save( &Screen );

/* Format output window */
win_Init( &Remote, 0, 0, 80, 9, 0x17, 0x1f, WIN_SCROLL | WIN_CRLF );
win_Init( &Local, 0,10, 80, 9, 0x17, 0x1f, WIN_SCROLL | WIN_CRLF |
          WIN_ACTIVE | WIN_HASCURSOR );
win_Init( &Status, 0,20, 80, 2, 0x17, 0x1f, WIN_SCROLL | WIN_CRLF );

win_Clr( &Screen );
win_Clr( &Remote );
win_Clr( &Local );
win_Clr( &Status );

win_printf( &Status, "COM port: 0x%04X, IRQ: 0x%02X, Baud: %ld\n",
            iSerPort, iSerIRQ, lBaud );
win_printf( &Status, "<ESC> = End      ");

```

```

win_printf( &Status, "Use '?' to display parameters          ");
win_printf( &Status, "(c) BHJ, MITI");

lpOldIRQ = ser_SetIRQHandler( iSerPort,          /* Install serial */
                              iSerIRQ,          /* interrupt.      */
                              GetSer,
                              SER_IER_RECEIVED | SER_IER_SENT );

c[0] = 0;
c[1] = 0;

win_GotoXY( &Local, 0, 0);
while( c[0] != 27 )
{
    c[0] = ( BYTE ) _getch();
    if( c[0] )
    {
        /* Output entered character through port... */
        ser_WriteByte( iSerPort, c[0], 0x8000, 0, 0 );
        /* ...and display in window. */
        if( isprint( c[ 0 ] ) ) win_Print( &Local, c );
        else win_printf( &Local, "<%02X>", ( UINT )c[ 0 ] );
    }
    else _getch();
}

/* Restore old handler */
ser_RestoreIRQHandler( iSerPort, iSerIRQ, lpOldIRQ );
win_Restore( pSaved, TRUE ); /* Restore screen contents */
if( iUART > INS8250 ) ser_FIFOLevel( iSerPort, 0 );
}

```