

```

/*****
/*          S E R U T I L . H          */
/**-----**/
/* Task      : Include file for SERUTIL.C */
/**-----**/
/* Author     : Michael Tischer / Bruno Jennrich */
/* Developed on : 04/08/1994 */
/* Last update  : 04/07/1995 */
/**-----**/
/* COMPILER    : Borland C++ 3.1, Microsoft Visual C++ 1.5 */
/*****
#ifndef _INC_SER_UTIL_H
#define _INC_SER_UTIL_H

#include <dos.h>

#include "types.h"
#include "win.h"

#define SER_COM1 0x3F8          /* Base address COM1 */
#define SER_COM2 0x2F8          /* Base address COM2 */

#define SER_IRQ_COM1 4          /* IRQ 4 = vector 0x0C */
#define SER_IRQ_COM2 3          /* IRQ 3 = vector 0x0B */

#define SER_TXBUFFER 0x00       /* Transmit register */
#define SER_RXBUFFER 0x00       /* Receive register */
#define SER_DIVISOR_LSB 0x00     /* Baud rate divisor LSB */
#define SER_DIVISOR_MSB 0x01     /* Baud rate divisor MSB */
#define SER_IRQ_ENABLE 0x01      /* Interrupt enable register */
#define SER_IRQ_ID 0x02          /* Interrupt ID register */
#define SER_FIFO 0x02           /* FIFO register */

```

```

#define SER_2FUNCTION      0x02      /* Alternate function register */
#define SER_LINE_CONTROL  0x03      /* Line control */
#define SER_MODEM_CONTROL 0x04      /* Modem control */
#define SER_LINE_STATUS    0x05      /* Line status */
#define SER_MODEM_STATUS   0x06      /* Modem status */
#define SER_SCRATCH        0x07      /* Scratch register */

/* IRQ enable register bits (Enable/disable interrupts) */
#define SER_IER_RECEIVED 0x01      /* IRQ after data received */
#define SER_IER_SENT      0x02      /* IRQ after byte sent */
#define SER_IER_LINE      0x04      /* IRQ after line status change */
#define SER_IER_MODEM     0x08      /* IRQ after modem status change */

/* IRQ ID bits (What initiated IRQ?) */
#define SER_ID_PENDING    0x01      /* Is serial IRQ pending? */
#define SER_ID_MASK       0x06      /* ID coded in bits 1 and 2 */
#define SER_ID_LINESTATUS 0x06      /* Line status (error or break) */
#define SER_ID_RECEIVED   0x04      /* Data received */
#define SER_ID_SENT       0x02      /* Byte was sent */
#define SER_ID_MODEMSTATUS 0x00      /* CTS, DSR, RI or RLSD change */

/* Bit assignment in FIFO register (if UART 16550A or later) */
#define SER_FIFO_ENABLE    0x01
#define SER_FIFO_RESETRCEIVE 0x02
#define SER_FIFO_RESETRANSMIT 0x04

/* FIFO bits (Number of bytes in FIFO after which IRQ occurs) */
#define SER_FIFO_TRIGGER0  0x00      /* Normal */
#define SER_FIFO_TRIGGER4  0x40      /* 4 bytes */
#define SER_FIFO_TRIGGER8  0x80      /* 8 bytes */
#define SER_FIFO_TRIGGER14 0xC0      /* 14 bytes */

```

```

        /* Line control register bits (transmission parameters) */
#define SER_LCR_WORDLEN      0x03    /* Number of bits being transmitted */
#define SER_LCR_5BITS        0x00
#define SER_LCR_6BITS        0x01
#define SER_LCR_7BITS        0x02
#define SER_LCR_8BITS        0x03
#define SER_LCR_2STOPBITS    0x04    /* 2 or 1.5 stop bits */
#define SER_LCR_1STOPBIT     0x00    /* 1 stop bit */

#define SER_LCR_NOPARITY     0x00    /* Disable parity check */
#define SER_LCR_ODDPARITY    0x08    /* Odd parity */
#define SER_LCR_EVENPARITY   0x18    /* Even parity */
#define SER_LCR_PARITYSET    0x28    /* Parity bit always set */
#define SER_LCR_PARITYCLR    0x38    /* Parity bit always cleared */
#define SER_LCR_PARITYMSK    0x38

#define SER_LCR_SENDBREAK    0x40    /* Send break as long as bit is set */
#define SER_LCR_SETDIVISOR   0x80    /* For access to baud rate divisor */

        /* Modem control register bits (signal control) */
#define SER_MCR_DTR          0x01    /* Set DTR signal */
#define SER_MCR_RTS          0x02    /* Set RTS signal */
#define SER_MCR_UNUSED       0x04
#define SER_MCR_IRQENABLED   0x08    /* Inform IRQ controller of IRQs */
#define SER_MCR_LOOP         0x10    /* Self-test */

        /* Line status register bits (transmission error) */
#define SER_LSR_DATARECEIVED 0x01    /* Data word (5 - 8 bits) received */
#define SER_LSR_OVERRUNERROR 0x02    /* Previous data word lost */
#define SER_LSR_PARITYERROR   0x04    /* Parity error */
#define SER_LSR_FRAMINGERROR  0x08    /* Start/stop bit error */
#define SER_LSR_BREAKDETECT   0x10    /* Break detected */

```

```

#define SER_LSR_ERRORMSK (SER_LSR_OVERRUNERROR|SER_LSR_PARITYERROR|\
                        SER_LSR_FRAMINGERROR|SER_LSR_BREAKDETECT)
#define SER_LSR_THREMPY    0x20
#define SER_LSR_TSREMPY    0x40

/* Modem status register bits (Which signals are set?) */
/* Delta... bits indicate whether status of corresponding */
/* signals have changed since the last */
/* read on modem status register */
#define SER_MSR_DCTS 0x01 /* Delta CTS (CTS status) */
#define SER_MSR_DDSR 0x02 /* Delta DSR (DSR status) */
#define SER_MSR_DRI 0x04 /* Delta RI (RI status) */
#define SER_MSR_DCD 0x08 /* Delta CD (CD status) */
#define SER_MSR_CTS 0x10 /* Clear To Send set */
#define SER_MSR_DSR 0x20 /* Data Set Ready set */
#define SER_MSR_RI 0x40 /* Ring Indicator set */
#define SER_MSR_CD 0x80 /* Carrier Detect set */

#define NOSER 0
#define INS8250 1 /* National Semiconductor UART's */
#define NS16450 2
#define NS16550A 3
#define NS16C552 4

#define SER_MAXBAUD 115200L /* Maximum baud rate */

#define SER_SUCCESS 0
#define SER_ERRSIGNALS 0x0300
#define SER_ERRTIMEOUT 0x0400

/* Prototypes */
INT ser_UARTType ( INT iSerPort );

```

```

INT  ser_Init          ( INT iSerPort, LONG lBaudRate, BYTE bParams );
VOID ser_FIFOLevel     ( INT iSerPort, BYTE bLevel );
VOID ser_CLRIRQ        ( INT iSerPort );
VOID ser_SETIRQ        ( INT iSerPort );
INT  ser_IsDataAvailable ( INT iSerPort );
INT  ser_IsWritingPossible( INT iSerPort );
INT  ser_IsModemStatusSet( INT iSerPort, BYTE bTestStatus );
VOID ser_SetModemControl ( INT iSerPort, BYTE bNewControl );
INT  ser_WriteByte     ( INT iSerPort, BYTE bData, UINT uTimeOut,
                        BYTE bSigMask, BYTE bSigVals );
INT  ser_ReadByte      ( INT iSerPort, PBYTE lpData, UINT uTimeOut,
                        BYTE bSigMask, BYTE bSigVals );
INT  ser_WritePacket   ( INT iSerPort, PBYTE lpData, INT iLen,
                        UINT uTimeOut, BYTE bSigMask, BYTE bSigVals );
INT  ser_ReadPacket    ( INT iSerPort, PBYTE lpData, INT iLen,
                        UINT uTimeOut, BYTE bSigMask, BYTE bSigVals );
VOID ( _interrupt _FP *ser_SetIRQHandler( INT iSerPort, INT iSerIRQ,
                        VOID ( _interrupt _FP *lpHandler ) (),
                        BYTE bEnablers ) )();
VOID ser_RestoreIRQHandler( INT iSerPort, INT iSerIRQ,
                        VOID ( _interrupt _FP *lpHandler ) () );
VOID ser_PrintError     ( PWINDOW pWin, INT e );
VOID ser_PrintModemStatus ( PWINDOW pWin, INT iSerPort );
VOID ser_PrintLineStatus ( PWINDOW pWin, INT iSerPort );
LONG ser_GetBaud        ( INT iSerPort );
VOID ser_PrintCardSettings( PWINDOW pWin, INT iSerPort );

#endif

```