

### **Problema PTC0004-22**

Realizar un programa en C que configure el puerto serie del PC, transmita 2 caracteres cada segundo y escriba en pantalla lo que recibe.

Datos:

Velocidad: 600 bps

Longitud del carácter (número de bits de datos): 7

Paridad: Impar

Número bits de parada: 2

Primer carácter: 52 (hexadecimal)

Segundo carácter: 25 (hexadecimal)

### **Solución PTC0004-22**

El puerto serie de un PC está constituido por una UART de la serie 8250 (National Semiconductor). Este dispositivo, hoy obsoleto, ha sido seguido por otros tales como el 16450 y el 16550. Existen numerosos tutoriales de cómo manejar este dispositivo y, en cualquier caso, la hoja de características del propio dispositivo (disponible en Internet).

Con estos datos estamos en condiciones de escribir un programa como el descrito en cuyos comentarios se van explicando cada uno de los pasos.

```

/*****
Programa para manejo del puerto serie
*****/

#include <stdio.h>

#define PUERTO 0x3F8 /* COM1 */

#define THR (PUERTO+0) /*THR: Transmitter Holding Register (con DLAB=0; en escritura)*/
#define RBR (PUERTO+0) /*RBR: REceiver Buffer Register (con DLAB=0; en lectura)*/

#define DLL (PUERTO+0) /*DLL: Divisor Latch Least Significant Byte (con DLAB=1;
lectura/escritura)*/
#define DLM (PUERTO+1) /*DLS: Divisor Latch Most Significant Byte (con DLAB=1;
lectura/escritura)*/

#define IER (PUERTO+1) /* IER: Interrupt Enable Register (con DLAB=0; lectura/escritura) */
#define ERBFI 0x01 /*Bit 0 (ERBFI): Enable Received Data Available Interrupt*/
#define ETBI 0x02 /*Bit 1 (ETBI): Enable Transmitter Holding Register Empty Interrupt*/
#define ELSI 0x03 /*Bit 2 (ELSI): Enable Receiver Line Status Interrupt*/
#define EDSSI 0x04 /*Bit 3 (EDSSI): Enable Modem Status Interrupt*/

#define LCR (PUERTO+3) /* LCR: Line Control Register (lectura/escritura) */
#define longitud_5 0 /*Longitud 5 bits*/
#define longitud_6 1 /*Longitud 6 bits*/
#define longitud_7 2 /*Longitud 7 bits*/
#define longitud_8 3 /*Longitud 8 bits*/

#define bitsparada_1 0 /*Bits de parada: 1 bits*/
#define bitsparada_2 4 /*Bits de parada: 2 bits*/

#define paridad_no 0 /*Paridad: no*/
#define paridad_si 8 /*Paridad: si*/

#define paridad_impar 0 /*Paridad impar*/
#define paridad_par 16 /*Paridad par*/

#define SPE 0x20 /* Bit 5:(SPE) Stick Parity Enable; con PEN=1 y EPS=1 transmite paridad 0;
con PEN=1 y EPS=0 transmite paridad 1*/
#define BC 0x40 /* Bit 6: (BC) Break Control; fuerza condición de Break en la línea (en bajo) */
#define DLAB 0x80 /* Bit 7 (DLAB): Divisor Latch Access Bit */

#define LSR (PUERTO+5) /*LSR: Line Status Register (lectura/escritura)*/
#define DR 0x01 /*Bit 0 (DR): Data Ready*/
#define OE 0x02 /*Bit 1 (OE): Overrun Error*/
#define PE 0x04 /*Bit 2 (PE): Parity Error*/
#define FE 0x08 /*Bit 3 (FE): Framing Error*/
#define BI 0x10 /*Bit 4 (BI): Break Interrupt*/
#define THRE 0x20 /*Bit 5 (THRE): Transmitter Holding Register (THR) Empty*/
#define TEMT 0x40 /*Bit 6 (TEMT): Transmitter Empty */
#define RFE 0x80 /*Bit 7: (RFE) Error in RCVR FIFO*/

#define ESCAPE 27

/* Velocidad: se calcula como divisor (x16) de un reloj de 1.8432MHz (los valores reales son
aproximaciones a los nominales)*/
#define velocidad_50 2304 /*Velocidad 50 bps*/
#define velocidad_75 1536 /*Velocidad 75 bps*/
#define velocidad_150 768 /*Velocidad 150 bps*/
#define velocidad_300 384 /*Velocidad 300 bps*/

```

```

#define velocidad_600      192    /*Velocidad 600 bps*/
#define velocidad_1200    96     /*Velocidad 1.200 bps*/
#define velocidad_2400    48     /*Velocidad 2.400 bps*/
#define velocidad_4800    24     /*Velocidad 4.800 bps*/
#define velocidad_9600    12     /*Velocidad 9.600 bps*/
#define velocidad_19200   6      /*Velocidad 19.200 bps*/
#define velocidad_56000   2      /*Velocidad 56.000 bps*/
#define velocidad_128000  1      /*Velocidad 128.000 bps*/

void main(void)
{
    unsigned char a,c,configuracion,caracter=1;
    unsigned int velocidad,terminar=0;

    /* Configurar velocidad */
    a=inportb(LCR);
    a=a|DLAB; /* Pone DLAB=1 */
    outportb(LCR,a);
    velocidad=velocidad_600;
    outportb(DLL,velocidad&0xFF);
    outportb(DLM,(velocidad>>8)&0xFF);

    /* Configurar paridad, longitud y stop */
    a=a&(!DLAB); /* Pone DLAB=0*/
    configuracion=longitud_7|bitsparada_2|paridad_si|paridad_impar;
    outportb(LCR,configuracion);

    /* Deshabilita interrupciones */
    outportb(IER,0); /*0000.0000 (con DLAB=0)*/

    /* Transmitir de forma continua dos caracteres seguido de pausa */

    while(1 && !terminar)
    {
        a=inportb(LSR);
        if(a&THRE) /* comprueba si el transmisor está vacío*/
        {
            if(caracter==1)
            {
                outportb(THR,0x52); /* Envía el primer carácter*/
                caracter=2;
            }
            else
            {
                outportb(THR,0x25); /* Envía el segundo carácter*/
                caracter=1;
                delay(1000); /* Espera en milisegundos*/
            }
        }
        if(a&DR) /* Comprueba si se ha recibido algún dato*/
        {
            c=inportb(RBR); /* Lee el dato recibido*/
            printf("%x; ",c);
        }

        if(kbhit()!=0) /* Comprueba si el usuario ha pulsado una tecla (para terminar)*/
            if(getch()==ESCAPE) terminar=1;
    }
}

```