Apéndice C

Biblioteca Serial.h

Esta sección contiene el código de la biblioteca *Serial.h*, la cual define metodos de configuracion y uso de los puertos SCI, que sintetizan los procedimientos definidos por el conjunto de archivos de Control Suite.

```
/*
 * Serial.h
 *
 */


//   Definiciones para el BaudRate
#define BR9600   0
#define BR19200  1
#define BR38400  2
#define BR57600  3
#define BR115200 4

volatile bool intrRx = false;       //variable que indica que
                            //se recibió un dato

void Serial_Process(void);
interrupt void sciaRxFifoIsr(void);

void Serial_Configure(Uint16 BR);
void Serial_Init(void);
void Serial_Start(void);

void Serial_putchar(int a);
void Serial_print(char *msg);


/*************************************************************
 * Configura el puerto serial A (SCI-A)                      *
 *                                                           *
```

```
 * Bits = 8                                                        *
 * Paridad = NO                                                    *
 * Modo = Asíncrono                                                *
 * Stop = 1 bit                                                    *
 *                                                                 *
 * BR: baud rate a emplear (9600,19200,38400,57600,115200)         *
 *                                                                 *
 *******************************************************************/
void Serial_Configure(Uint16 BR){

        SciaRegs.SCICCR.all = 0x0007;
        SciaRegs.SCICTL1.all = 0x0003;
        SciaRegs.SCICTL2.bit.TXINTENA = 1;
        SciaRegs.SCICTL2.bit.RXBKINTENA = 1;

        //LSPCLK = 25MHz
        //BaudRate = LSPCLK/( (BRR+1)*8 )
        switch(BR){
                case 1:
                        // BRR = 161 = 19200 Bauds
                        SciaRegs.SCIHBAUD.all = 0x00;
                        SciaRegs.SCILBAUD.all = 0xA1;
                        break;
                case 2:
                        // BRR = 80 = 38400 Bauds
                        SciaRegs.SCIHBAUD.all = 0x00;
                        SciaRegs.SCILBAUD.all = 0x50;
                        break;
                case 3:
                        // BRR = 52 = 57600 Bauds
                        SciaRegs.SCIHBAUD.all = 0x00;
                        SciaRegs.SCILBAUD.all = 0x34;
                        break;
                case 4:
                        // BRR = 26 = 115200 Bauds
                        SciaRegs.SCIHBAUD.all = 0x00;
                        SciaRegs.SCILBAUD.all = 0x1A;
                        break;
                default:
                        // BRR = 324 = 9600 Bauds
                        SciaRegs.SCIHBAUD.all = 0x01;
                        SciaRegs.SCILBAUD.all = 0x44;
                        break;
        }

        SciaRegs.SCIFFTX.all = 0xC020;
        SciaRegs.SCIFFRX.all = 0x0021;
        SciaRegs.SCIFFCT.all = 0x0;
```

```
}//end Serial_Configure


/*************************************************************
* Inicializa el puerto serial A (SCI-A), para funcionar a    *
* través del puerto USB, configura la interrupción de        *
* recepción de datos.                                        *
*                                                            *
* TX = GPIO84                                                *
* RX = GPIO85                                                *
*                                                            *
*************************************************************/
void Serial_Init(void){

        EALLOW;
        //Configura los GPIO del puerto SCI-A
        GpioCtrlRegs.GPCMUX2.bit.GPIO84 = 1;
        GpioCtrlRegs.GPCMUX2.bit.GPIO85 = 1;
        GpioCtrlRegs.GPCGMUX2.bit.GPIO84 = 1;
        GpioCtrlRegs.GPCGMUX2.bit.GPIO85 = 1;

        //PieVectTable.SCIA_RX_INT = &sciaRxFifoIsr; //define la
                // rutina de interrupción
        EDIS;

        //PieCtrlRegs.PIECTRL.bit.ENPIE = 1; //habilita el bloque
                                             // del PIE
        //PieCtrlRegs.PIEIER9.bit.INTx1 = 1; //habilita la
        //interrupción del PIE Grupo 9, INT1 (SCIA_RX)


}//end Serial_Init



/*************************************************************
* Inicia la operación del puerto serial A (SCI-A)            *
*                                                            *
 *************************************************************/
void Serial_Start(void){

        SciaRegs.SCICTL1.all = 0x0023;  // Reinicia el puerto
                                        // serial
        SciaRegs.SCIFFTX.bit.TXFIFORESET = 1;
        SciaRegs.SCIFFRX.bit.RXFIFORESET = 1;

    //IER = 0x100; // Habilita CPU INT
```

```
    //EINT;

}//end Serial_Start



/*************************************************************
* Envía un caracter a través del puerto serial A (SCI-A)     *
*                                                            *
 *************************************************************/
void Serial_putchar(int a){

        while (SciaRegs.SCIFFTX.bit.TXFFST != 0);//espera que se
                                                 // pueda transmitir
        SciaRegs.SCITXBUF.all = a;

}//end Serial_putchar

/*************************************************************
* Envía una cadena de caracteres a través puerto serial      *
* A (SCI-A), la cadena debe terminar con "\0"                *
*                                                            *
*************************************************************/
void Serial_print(char *msg){
        int i = 0;

        while(msg[i] != '\0'){
                Serial_putchar(msg[i]);
                i++;
        }

}//end Serial_print


/*************************************************************
*       Rutina de interrupción de SCI-A                      *
*************************************************************/
interrupt void sciaRxFifoIsr(void){

        intrRx = true;
        Serial_Process();
        SciaRegs.SCIFFRX.bit.RXFFOVRCLR = 1;    // Limpia bandera de sobreflujo
        SciaRegs.SCIFFRX.bit.RXFFINTCLR = 1;    // Limpia bandera de intrrupción

        PieCtrlRegs.PIEACK.all |= 0x100;

}//end sciaRxFifoIsr
```