

Dependiendo del estado del pin 9 el siguiente programa se comporta como voltímetro o como termómetro. El dato es transmitido por el pin 10 mediante RS232 a 4800 baudios.

Practicas con Microcontroladores Firtec [2007]

www.firtec.com.ar – informes@firtec.com.ar

```
#include <hidef.h> /* for EnableInterrupts macro */
#include "derivative.h" /* include peripheral declarations */
#define COCO 0x80
interrupt 6 void TIMER();
void delay_baudio(void);
void delay(void);
void Configurar_Timer(void);
void Configurar_Conversor(void);
void Leer_Conversor(void);
void Transmitir_Caracter(void);
const unsigned char Vel_Termo = 254;
const unsigned char Vel_Voltmetro = 100;
unsigned int T_Muestras = 0;
unsigned char Unidad = 0;
unsigned char Decena = 0;
unsigned char Centena = 0;
unsigned char bandera = 0;
unsigned char Dato = 0;
unsigned char Caracter = 0;
unsigned char bit;
unsigned char Temporal = 0;
unsigned char tiempo_bit;
const unsigned char buffer [16] =
    {0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x67,0x77,
     0x7C,0x39,0x5E,0x79,0x71};

/*DEFINICIÓN DE MACROS*/
#define bit_set(bit,ubicacion)(ubicacion |= 1<<bit)
/* Pone un bit en 1 en la posicion indicada */
#define bit_clr(bit,ubicacion)(ubicacion &= ~(1<<bit))
/* Pone un bit en 0 en la posicion indicada */

void main(void) {

    DisableInterrupts;
    PTB = 0;
    PTD = 0;
    DDRD = 0X7F;
    DDRB = 0X7F;
    Configurar_Timer();
    Configurar_Conversor();
    PTD_PTD6 = 1;    //Linea TX en 1
    EnableInterrupts;    //Interrupciones Activas
    while(1) {
```

```

    __RESET_WATCHDOG();

}

}
/*****
/*      Subrutina que maneja el Tranmisor      */
*****/
void Transmitir_Caracter(void){
    //DisableInterrupts; /*Interrupciones Desactivadas*/
    __RESET_WATCHDOG();
    bit=0;
    PTD_PTD6 = 0; //Pongo el bit de inicio
    delay_baudio();
    do{

        if(Caracter & 0x01 == 0x01){

            PTD_PTD6 = 1; //Pongo TX en 1
            delay_baudio();
        }
        else
        {

            PTD_PTD6 = 0; //Pongo TX en 0
            delay_baudio();
        }

        Caracter = Caracter >>1; //Roto para pasar los 8 bits
        bit++;
    }while(bit <=7);

    PTD_PTD6 = 1; //Pongo el bit de Stop
    delay_baudio();

    __RESET_WATCHDOG();
}
/*****
/*      ISR DEL DESBORDE DEL TIMER      */
*****/
interrupt 6 void TIMER(){
    __RESET_WATCHDOG();
    T_Muestras++;
    if(T_Muestras == 5){
        Transmitir_Caracter();
    }
}

switch(bandera){

```

```

case 0:{
    PTD_PTD5 = 0;
    PTB = buffer[Unidad];

    PTD_PTD2 =1;
    bandera++;
    delay();
    break;
}
case 1:{

    PTB = buffer[Decena];

    PTD_PTD2 =0;
    PTD_PTD3 = 1;
    delay();
    if((PTD &0x80) == 0x80){

        if(T_Muestras >= Vel_Termo){

            T_Muestras = 0x00;
            Leer_Conversor();
        }

        bandera=0;
        PTD_PTD3 = 0;
        break;
    }

    bandera = 2;
    break;
}

case 2:{

    PTB = buffer[Centena];

    PTD_PTD3 =0;

    PTD_PTD5 = 1;
    bandera = 0;
    delay();
    if(T_Muestras >= Vel_Voltimetro){

        T_Muestras = 0x00;
        Leer_Conversor();
    }
    break;
}

```

```

}
__RESET_WATCHDOG();
bit_clr(7,TSC); //Borra la bandera de la inte
}
/*****
/*      LEER EL DATO DESDE EL CONVERSIONOR      */
*****/
void Leer_Conversor(void){

    Decena = 0;Centena =0;

    if(ADSCR & COCO == COCO){ /*Espero que termine la conversión.*

    Unidad = ADR;
    Caracter = Unidad;

    while(Unidad >= 100){
    Unidad = Unidad - 100;
    Centena++;
    }
    Unidad + 100;
    while(Unidad >= 10){
    Unidad = Unidad - 10;
    Decena++;
    }
    }

}

/*****
/*      CONFIGURACION DEL TIMER POR DESBORDE      */
*****/
void Configurar_Timer(void){

    TSC = 0X76; /*Stop & reset,interrupcion activa, prescaler=64*/
    TSC0 = 0; /*Resto de funciones desactivadas*/
    TSC1 = 0;
    TMODH = 0;
    TMODL = 0x01;
    TSC ^= 0X10; /*Un-reset TImeR*/

    TSC ^= 0X20; /*Start Timer*/
}
/*****
/*      CONFIGURACION DEL CONVERSIONOR A/D      */
*****/

void Configurar_Conversor(void){

    ADICLK = 0X00;
    ADSCR = 0X27;
}

```

```

/*****
/*      Generador de Baudios XTAL de 4MHZ      */
*****/

//  bit = 24 -> 2400 Baudios
//  bit = 10 -> 4800 Baudios
*****/

void delay_baudio(void){

    for(tiempo_bit=0;tiempo_bit <=10;tiempo_bit++){

        }

    }

/*****/
void delay(void){
unsigned char a;

for(a=0; a<=254 ;a++){

}

}

```



