Microcontroladores

Semestre 2021-0

Semana 7

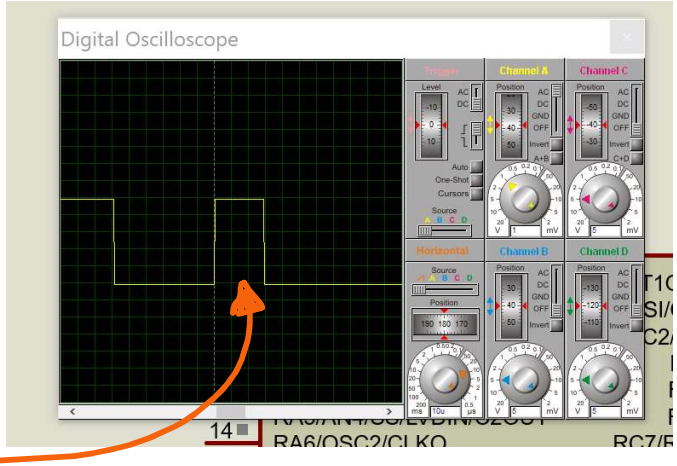
Preguntas previas:

- PC2 es mañana?
  - Si, mañana miércoles 23 de febrero de 19:00 a 21:50
- No me sale la comunicación serial a 33600 8N1 que necesito para el DD

Configuration Bits			BRG/EUSART Mode	Baud Rate Formula
SYNC	BRG16	BRGH		
0	0	0	8-bit/Asynchronous	Fosc/[64 (n + 1)]

$$t_{bit} = \frac{1}{33600}$$

$$t_{bit} = 29.76\,\mu s$$



$$n = \frac{\left(\frac{Fosc}{Bitrate}\right)}{64} - 1$$

$$n = \frac{\left(\frac{48000000}{33600}\right)}{64} - 1$$

$n = 21$ → ¿se podrá colocar en el registro BRG?

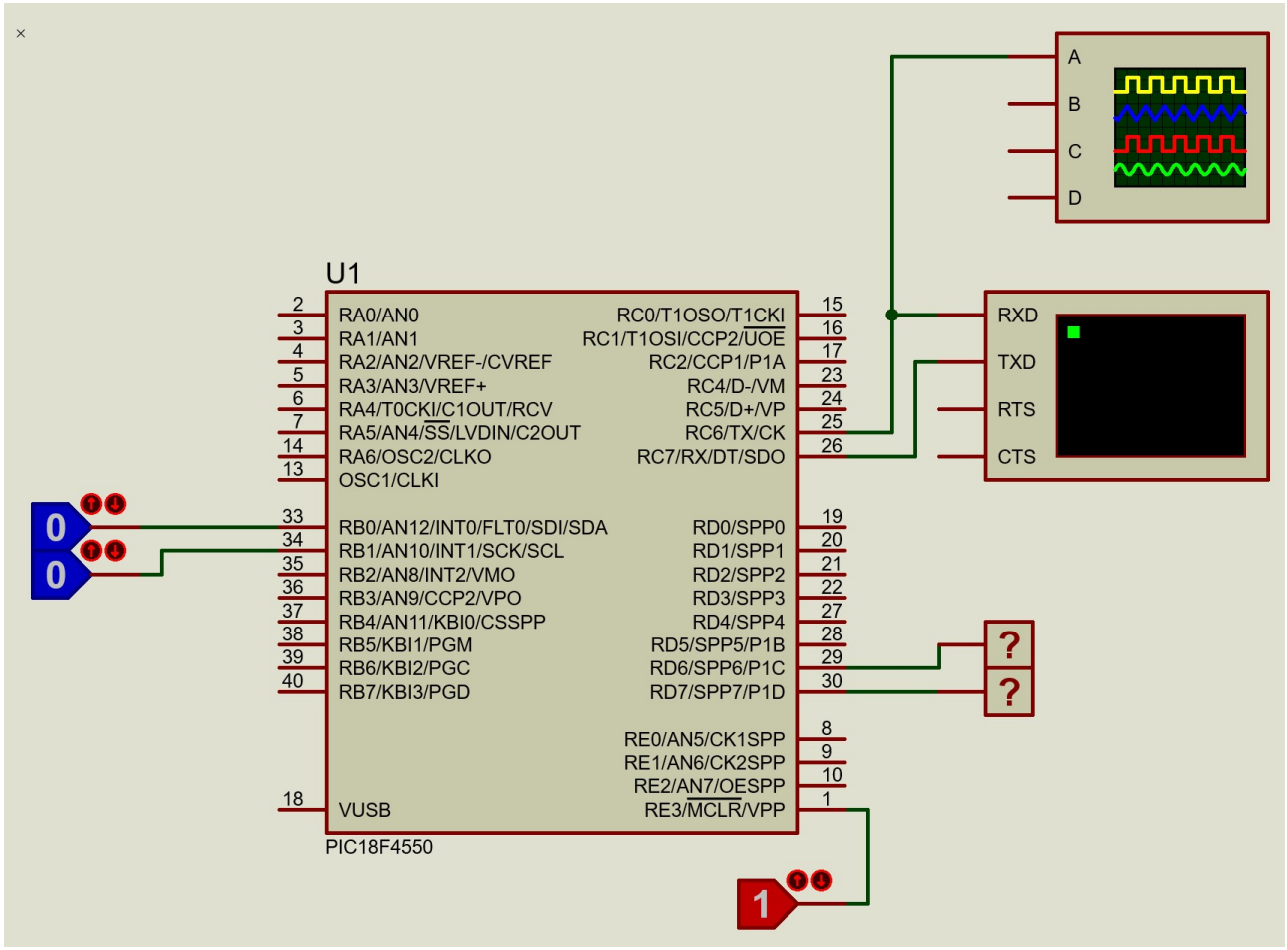
SI

Verificar la vtx del terminal serial

Agenda:

- Aplicaciones con la comunicación serial UART

Circuito de prueba



Ejemplo para visualizar un menú por el terminal serial

```
#pragma config PLLDIV = 1       // PLL Prescaler Selectio
#pragma config CPUDIV = OSC1_PLL2// System Clock Postscal
#pragma config FOSC = XTPLL_XT  // Oscillator Selection b
#pragma config PWRT = ON        // Power-up Timer Enable
#pragma config BOR = OFF        // Brown-out Reset Enable
#pragma config WDT = OFF        // Watchdog Timer Enable
#pragma config CCP2MX = ON      // CCP2 MUX bit (CCP2 inp
#pragma config PBADEN = OFF     // PORTB A/D Enable bit (
#pragma config MCLRE = ON       // MCLR Pin Enable bit (M
#pragma config LVP = OFF        // Single-Supply ICSP Ena

#include <xc.h>
#define _XTAL_FREQ 48000000UL       //frecuencia de traba

unsigned char menu_1[] = {"Bienvenidos a la aplicacion"};
unsigned char menu_2[] = {"Elija opcion:               "};
unsigned char menu_3[] = {"(1) Prender LED 1           "};
unsigned char menu_4[] = {"(2) Apagar LED 1            "};
unsigned char menu_5[] = {"(3) Prender LED 2           "};
unsigned char menu_6[] = {"(4) Apagar LED 2            "};
unsigned char menu_7[] = {"(m) Visualizar el menu      "};

void init_config(void){
    TRISDbits.RD6 = 0;
    TRISDbits.RD7 = 0;
}

void EUSART_config(void){
    SPBRGH = 0;                 //Ignorado debido a que B
    SPBRG = 21;                 //33600 baudios
    TRISCbits.RC6 = 0;          //Puerto RC6 como salida,
    RCSTAbits.SPEN = 1;         //Encendemos el puerto se
    TXSTAbits.TXEN = 1;         //Encendemos el transmiso
}
```

```
void EUSART_siguientelinea(void){
    TXREG = 0x0A;
    while(TXSTAbits.TRMT == 0);
    TXREG = 0x0D;
    while(TXSTAbits.TRMT == 0);
}

void EUSART_enviacadena(const unsigned char *vector,unsigned char pos){
        for (unsigned char x=0;x<pos;x++){
            TXREG = vector[x];
            while(TXSTAbits.TRMT == 0);
        }
}

void main(void) {
    init_config();
    EUSART_config();
    EUSART_enviacadena(menu_1, 28);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_2, 28);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_3, 28);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_4, 28);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_5, 28);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_6, 28);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_7, 28);
    EUSART_siguientelinea();
    while(1){

    }
}
```

Ejemplo: Recepción de datos desde el terminal serial

```c
#pragma config PLLDIV = 1       // PLL Prescaler Selection bit
#pragma config CPUDIV = OSC1_PLL2// System Clock Postscaler Se
#pragma config FOSC = XTPLL_XT  // Oscillator Selection bits
#pragma config PWRT = ON        // Power-up Timer Enable bit
#pragma config BOR = OFF        // Brown-out Reset Enable bits
#pragma config WDT = OFF        // Watchdog Timer Enable bit
#pragma config CCP2MX = ON      // CCP2 MUX bit (CCP2 input/ou
#pragma config PBADEN = OFF     // PORTB A/D Enable bit (PORTB
#pragma config MCLRE = ON       // MCLR Pin Enable bit (MCLR p
#pragma config LVP = OFF        // Single-Supply ICSP Enable b

#include <xc.h>
#include <stdio.h>
#include <string.h>
#define _XTAL_FREQ 48000000UL       //frecuencia de trabajo

unsigned char menu_1[] = {"Bienvenidos a la aplicacion"};
unsigned char menu_2[] = {"Elija opcion:               "};
unsigned char menu_3[] = {"(1) Prender LED 1           "};
unsigned char menu_4[] = {"(2) Apagar LED 1            "};
unsigned char menu_5[] = {"(3) Prender LED 2           "};
unsigned char menu_6[] = {"(4) Apagar LED 2            "};
unsigned char menu_7[] = {"(m) Visualizar el menu      "};
unsigned char error[] = {"Tecla invalida, intente de nuevo"};

void init_config(void){
    TRISDbits.RD6 = 0;
    TRISDbits.RD7 = 0;
    INTCONbits.GIE = 1;     //Interruptor global de interrupci
    INTCONbits.PEIE = 1;    //Interruptor de interrupciones de
    PIE1bits.RCIE = 1;      //Habilitamos las interrupciones d
}

void EUSART_config(void){
    SPBRGH = 0;                 //Ignorado debido a que BRG16=
    SPBRG = 21;                 //33600 baudios
    TRISCbits.RC6 = 0;          //Puerto RC6 como salida, no e
    RCSTAbits.SPEN = 1;         //Encendemos el puerto serial
    TXSTAbits.TXEN = 1;         //Encendemos el transmisor
    RCSTAbits.CREN = 1;         //Encendemos el receptor
}
```

```c
void EUSART_siguientelinea(void){
    TXREG = 0x0A;
    while(TXSTAbits.TRMT == 0);
    TXREG = 0x0D;
    while(TXSTAbits.TRMT == 0);
}

void EUSART_enviacadena(const unsigned char *vector){
    unsigned char cantidad = 0;
    cantidad = strlen(vector);
        for (unsigned char x=0;x<cantidad;x++){
            TXREG = vector[x];
            while(TXSTAbits.TRMT == 0);
        }
}

void EUSART_enviachar(unsigned char papa){
    TXREG = papa;
    while(TXSTAbits.TRMT == 0);
}

void vis_menu(void){
    EUSART_enviacadena(menu_1);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_2);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_3);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_4);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_5);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_6);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu_7);
    EUSART_siguientelinea();
}
```

```c
void main(void) {
    init_config();
    EUSART_config();
    vis_menu();
    while(1);
}

void __interrupt() RC_ISR(void){
    PIR1bits.RCIF = 0;
    switch(RCREG){
        case '1':
            LATDbits.LD6 = 1;
            break;
        case '2':
            LATDbits.LD6 = 0;
            break;
        case '3':
            LATDbits.LD7 = 1;
            break;
        case '4':
            LATDbits.LD7 = 0;
            break;
        case 'm':
            vis_menu();
            break;
        default:
            EUSART_enviacadena(error);
            EUSART_siguientelinea();
    }
}
```

- Se esta empleando interrupciones para la recepción en el EUSART
- La función “strlen” necesita de las librerías “stdio.h y string.h”
- Tener en cuenta que al presionar una tecla en el teclado dentro del terminal serial se enviará su correspondiente código en ASCII

Fin de la sesión