

Microcontroladores

Semestre 2021-0

Semana 7

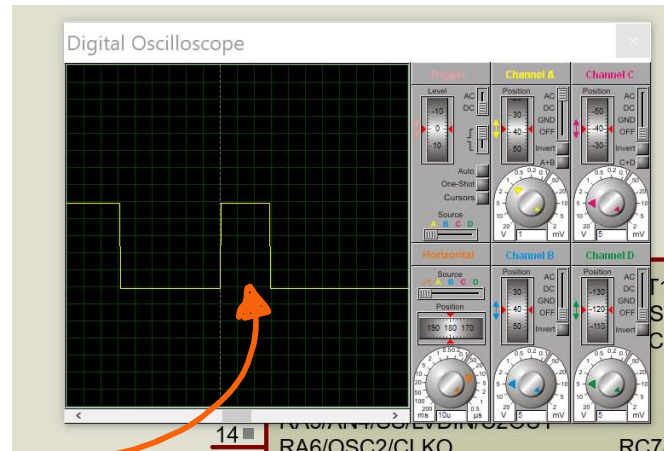
Preguntas previas:

- 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	$F_{osc}/[64 (n + 1)]$

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

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



$$m = \left(\frac{F_{osc}}{B_{rate}} \right) - 1$$

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

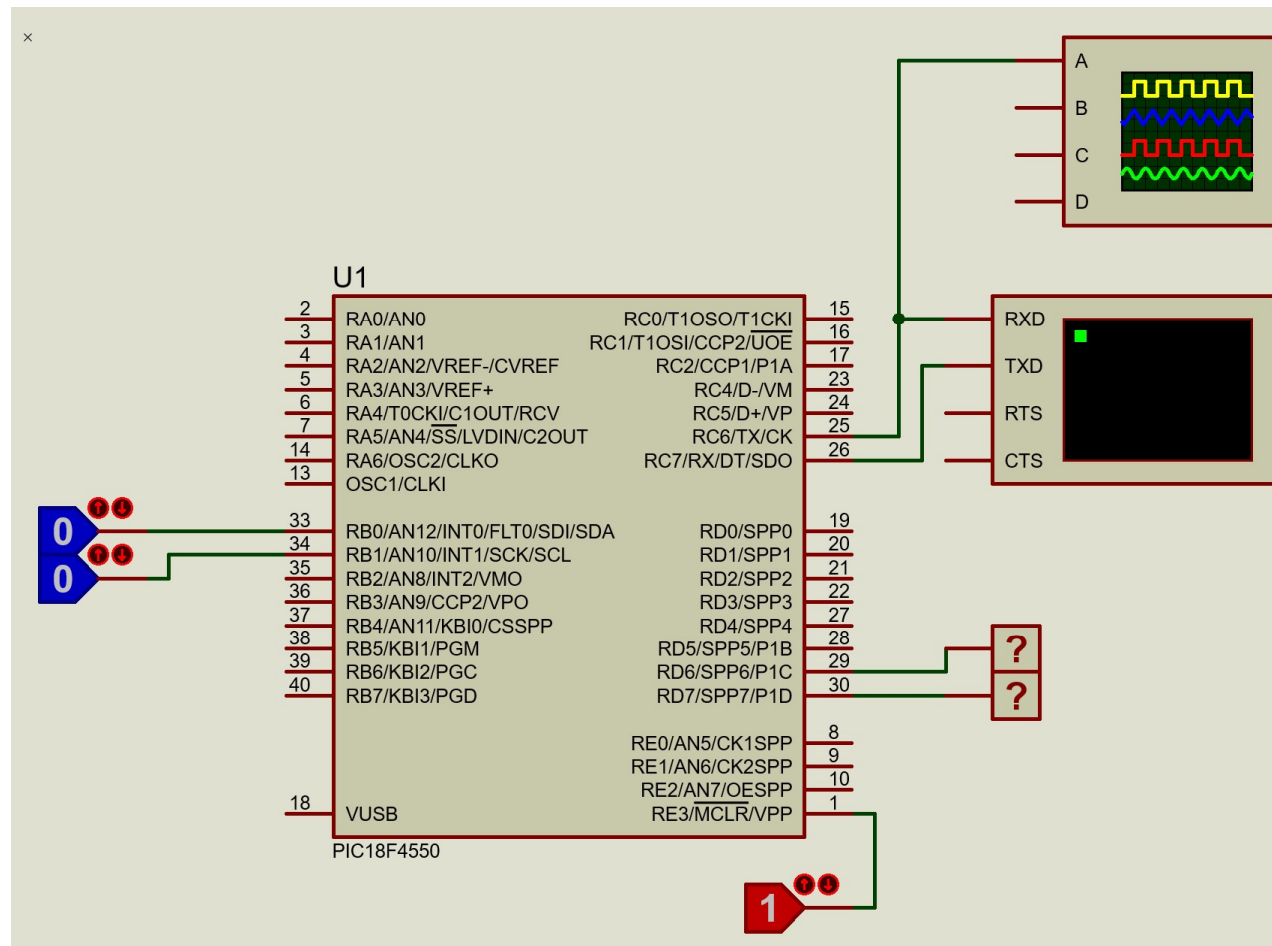
$m = 21$ → ¿Se podrá colocar en el registro BRG?
SI

Verifican la vtx del Terminal serial

Agenda:

- Aplicaciones con la comunicación serial UART

Circuito de prueba



Ejemplo para visualizar un menú por el terminal serial

```
1  #pragma config PLLDIV = 1          // PLL Prescaler Selection
2  #pragma config CPUDIV = OSC1_PLL2 // System Clock Postscaler
3  #pragma config FOSC = XTPLL_XT     // Oscillator Selection bits
4  #pragma config FWRT = ON            // Power-up Timer Enable bit
5  #pragma config BOR = OFF           // Brown-out Reset Enable bit
6  #pragma config WDT = OFF           // Watchdog Timer Enable bit
7  #pragma config CCP2MX = ON         // CCP2 MUX bit (CCP2 input/output)
8  #pragma config PBADEN = OFF        // PORTB A/D Enable bit (PBADEN)
9  #pragma config MCLRE = ON          // MCLR Pin Enable bit (MCLR)
10 #pragma config LVP = OFF           // Single-Supply ICSP Enable bit
11
12 #include <xc.h>
13 #define _XTAL_FREQ 48000000UL      //frecuencia de trabajo
14
15 unsigned char menu_1[] = {"Bienvenidos a la aplicacion"};
16 unsigned char menu_2[] = {"Elija opcion:"};
17 unsigned char menu_3[] = {"(1) Prender LED 1"};
18 unsigned char menu_4[] = {"(2) Apagar LED 1"};
19 unsigned char menu_5[] = {"(3) Prender LED 2"};
20 unsigned char menu_6[] = {"(4) Apagar LED 2"};
21 unsigned char menu_7[] = {"(m) Visualizar el menu"};
22
23 void init_config(void){
24     TRISDbits.RD6 = 0;
25     TRISDbits.RD7 = 0;
26 }
27
28 void EUSART_config(void){
29     SPBRGH = 0; //Ignorado debido a que Baud Rate Prescaler es 1
30     SPBRG = 21; //33600 baudios
31     TRISCbits.RC6 = 0; //Puerto RC6 como salida,
32     RCSTAbits.SPEN = 1; //Encendemos el puerto serial
33     TXSTAbits.TXEN = 1; //Encendemos el transmisor
34 }
```

```
38 void EUSART_siguientelinea(void){
39     TXREG = 0x0A;
40     while(TXSTAbits.TRMT == 0);
41     TXREG = 0x0D;
42     while(TXSTAbits.TRMT == 0);
43 }
44
45 void EUSART_enviacadena(const unsigned char *vector,unsigned char pos){
46     for (unsigned char x=0;x<pos;x++){
47         TXREG = vector[x];
48         while(TXSTAbits.TRMT == 0);
49     }
50 }
51
52 void main(void) {
53     init_config();
54     EUSART_config();
55     EUSART_enviacadena(menu_1, 28);
56     EUSART_siguientelinea();
57     EUSART_enviacadena(menu_2, 28);
58     EUSART_siguientelinea();
59     EUSART_enviacadena(menu_3, 28);
60     EUSART_siguientelinea();
61     EUSART_enviacadena(menu_4, 28);
62     EUSART_siguientelinea();
63     EUSART_enviacadena(menu_5, 28);
64     EUSART_siguientelinea();
65     EUSART_enviacadena(menu_6, 28);
66     EUSART_siguientelinea();
67     EUSART_enviacadena(menu_7, 28);
68     EUSART_siguientelinea();
69     while(1){
70
71     }
72 }
```

Ejemplo: Recepción de datos desde el terminal serial

```

1  #pragma config PLLDIV = 1          // PLL Prescaler Selection bits
2  #pragma config CPUDIV = OSC1_PLL2 // System Clock Postscaler Selection
3  #pragma config FOSC = XTPLL_XT     // Oscillator Selection bits
4  #pragma config PWRT = ON           // Power-up Timer Enable bit
5  #pragma config BOR = OFF           // Brown-out Reset Enable bits
6  #pragma config WDT = OFF           // Watchdog Timer Enable bit
7  #pragma config CCP2MX = ON         // CCP2 MUX bit (CCP2 input/output)
8  #pragma config PBADEN = OFF        // PORTB A/D Enable bit (PORTB)
9  #pragma config MCLRE = ON          // MCLR Pin Enable bit (MCLR)
10 #pragma config LVP = OFF           // Single-Supply ICSP Enable bit
11
12 #include <xc.h>
13 #include <stdio.h>
14 #include <string.h>
15 #define _XTAL_FREQ 48000000UL      //frecuencia de trabajo
16
17 unsigned char menu_1[] = {"Bienvenidos a la aplicacion"};
18 unsigned char menu_2[] = {"Elija opcion:"};
19 unsigned char menu_3[] = {"(1) Prender LED 1"};
20 unsigned char menu_4[] = {"(2) Apagar LED 1"};
21 unsigned char menu_5[] = {"(3) Prender LED 2"};
22 unsigned char menu_6[] = {"(4) Apagar LED 2"};
23 unsigned char menu_7[] = {"(m) Visualizar el menu"};
24 unsigned char error[] = {"Tecla invalida, intente de nuevo"};
25
26 void init_config(void){
27     TRISDbits.RD6 = 0;
28     TRISDbits.RD7 = 0;
29     INTCONbits.GIE = 1; //Interruptor global de interrupciones
30     INTCONbits.PEIE = 1; //Interruptor de interrupciones de EUSART
31     PIR1bits.RCIF = 1; //Habilitamos las interrupciones de EUSART
32 }
33
34 void EUSART_config(void){
35     SPBRGH = 0; //Ignorado debido a que BRG16 es 0
36     SPBRG = 21; //33600 baudios
37     TRISCbits.RC6 = 0; //Puerto RC6 como salida, no como entrada
38     RCSTAbits.SPEN = 1; //Encendemos el puerto serial
39     TXSTAbits.TXEN = 1; //Encendemos el transmisor
40     RCSTAbits.CREN = 1; //Encendemos el receptor
41 }
42
43 void EUSART_siguientelinea(void){
44     TXREG = 0x0A;
45     while(TXSTAbits.TRMT == 0);
46     TXREG = 0x0D;
47     while(TXSTAbits.TRMT == 0);
48 }
49
50 void EUSART_enviacadena(const unsigned char *vector){
51     unsigned char cantidad = 0;
52     cantidad = strlen(vector);
53     for (unsigned char x=0;x<cantidad;x++){
54         TXREG = vector[x];
55         while(TXSTAbits.TRMT == 0);
56     }
57 }
58
59 void EUSART_enviachar(unsigned char papa){
60     TXREG = papa;
61     while(TXSTAbits.TRMT == 0);
62 }
63
64 void vis_menu(void){
65     EUSART_enviacadena(menu_1);
66     EUSART_siguientelinea();
67     EUSART_enviacadena(menu_2);
68     EUSART_siguientelinea();
69     EUSART_enviacadena(menu_3);
70     EUSART_siguientelinea();
71     EUSART_enviacadena(menu_4);
72     EUSART_siguientelinea();
73     EUSART_enviacadena(menu_5);
74     EUSART_siguientelinea();
75     EUSART_enviacadena(menu_6);
76     EUSART_siguientelinea();
77     EUSART_enviacadena(menu_7);
78     EUSART_siguientelinea();
79 }
80
81 void main(void) {
82     init_config();
83     EUSART_config();
84     vis_menu();
85     while(1);
86 }
87
88 void __interrupt() RC_ISR(void){
89     PIR1bits.RCIF = 0;
90     switch(RCREG){
91         case '1':
92             LATDbits.LD6 = 1;
93             break;
94         case '2':
95             LATDbits.LD6 = 0;
96             break;
97         case '3':
98             LATDbits.LD7 = 1;
99             break;
100        case '4':
101            LATDbits.LD7 = 0;
102            break;
103        case 'm':
104            vis_menu();
105            break;
106        default:
107            EUSART_enviacadena(error);
108            EUSART_siguientelinea();
109    }
110 }

```

- 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