

Microcontroladores

Semana 13

Sesión Laboratorio
Profesor Kalun José Lau Gan

1

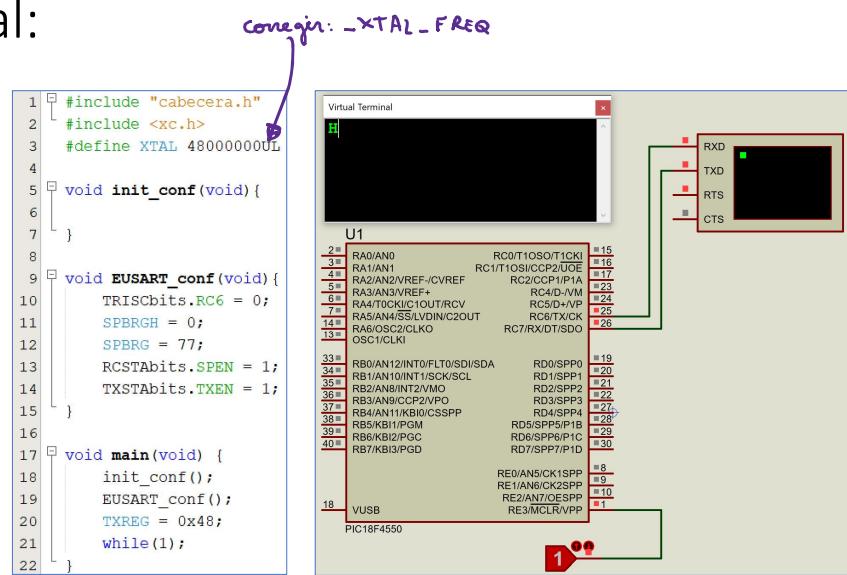
Agenda

- El EUSART del PIC18F4550

2

Ejemplo inicial:

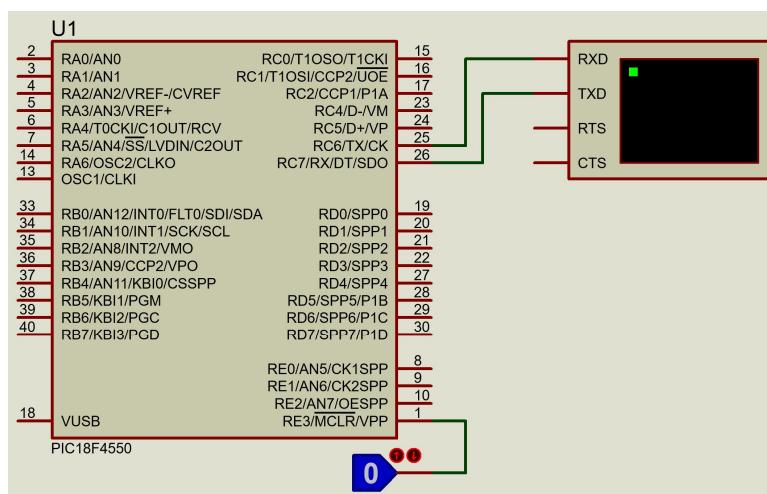
- Configuración del EUSART para 9600 y transmisión de letra H (ASCII 0x48) al terminal



3

Ejemplo de transmisión de datos vía EUSART

- Enviar “Hola mundo!” a través del módulo EUSART empleando 9600 8N1



4

(cont...)

- Configurar el EUSART para que trabaje a 9600 8N1

Aplicando la primera fórmula:

$$m = \frac{\left(\frac{F_{OSC}}{\text{Bitrate}}\right)}{64} - 1 \quad \text{conf. 8bits}$$

$$M = \frac{\frac{48E6}{9600}}{64} - 1$$

$$M = 77.125$$



5

(cont...)

```

1 #include <xc.h>
2 #include "cabecera.h"
3 #define _XTAL_FREQ 48000000UL
4
5 void EUSART_conf(void){
6     SPBRG = 77;           //Vtx 9600
7     RCSTAbits.SPEN = 1;  //Encendemos el EUSART
8     TXSTAbits.TXEN = 1;  //Encendemos el transmisor del EUSART
9 }
```

Falta: TRISChits.RC6 = 0;

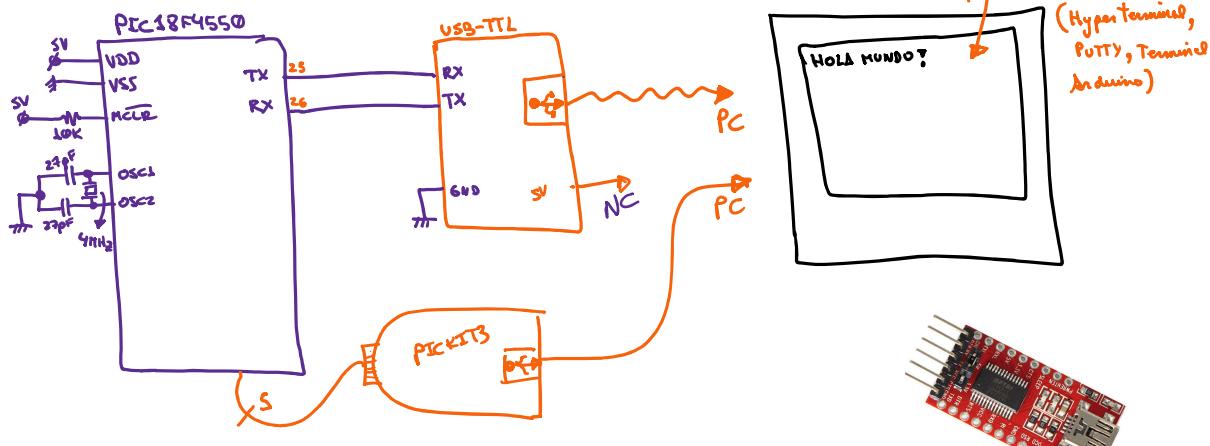
```

11 void main(void) {
12     EUSART_conf();
13     TXREG = 'H';
14     while (TXSTAbits.TRMT == 0);
15     __delay_ms(100);
16     TXREG = 'o';
17     while (TXSTAbits.TRMT == 0);
18     __delay_ms(100);
19     TXREG = '1';
20     while (TXSTAbits.TRMT == 0);
21     __delay_ms(100);
22     TXREG = 'a';
23     while (TXSTAbits.TRMT == 0);
24     __delay_ms(100);
25     TXREG = ' ';
26     while (TXSTAbits.TRMT == 0);
27     __delay_ms(100);
28     TXREG = 'm';
29     while (TXSTAbits.TRMT == 0);
30     __delay_ms(100);
31     TXREG = 'u';
32     while (TXSTAbits.TRMT == 0);
33     __delay_ms(100);
34     TXREG = 'n';
35     while (TXSTAbits.TRMT == 0);
36     __delay_ms(100);
37     TXREG = 'd';
38     while (TXSTAbits.TRMT == 0);
39     __delay_ms(100);
40     TXREG = 'o';
41     while (TXSTAbits.TRMT == 0);
42     __delay_ms(100);
43     TXREG = '!';
44     while (TXSTAbits.TRMT == 0);
45     while(1);
46 }
```

6

(cont...)

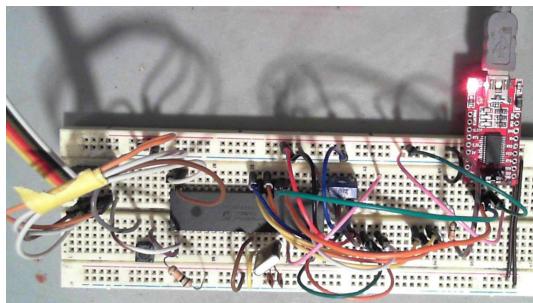
- Implementación física del ejemplo



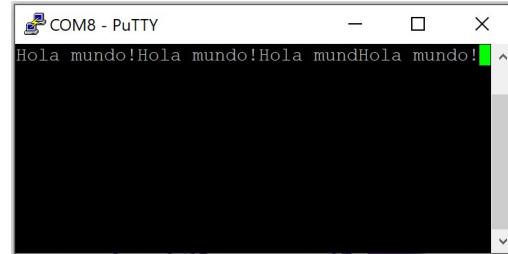
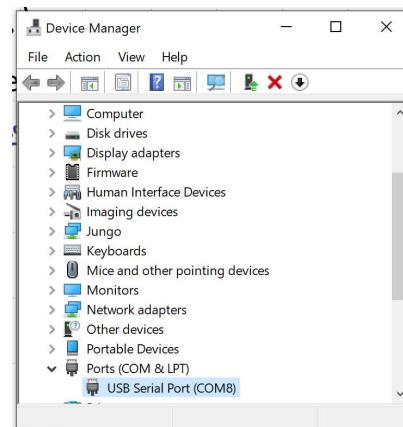
Link de descarga de drivers FTDI: <http://www.ftdichip.com/FTDrivers.htm>

7

(cont...)



- Al conectar el USB-TTL a la PC el Windows debe de detectar correctamente el módulo.
- Ingresar al Administrador de Dispositivos (Device Manager) para verificar el puerto serial COMx creado
- Utilizar un terminal serial (i.e. PuTTY), colocar el COMx y la velocidad configurada en el EUSART
- <https://www.putty.org/>



8

(cont...)

- Optimizando el código:

```

1  #include <xc.h>
2  #include "cabecera.h"
3  #define _XTAL_FREQ 48000000UL
4
5  unsigned char mensaje[]={"Hola mundo!"};
6
7  void EUSART_conf(void){
8      SPBRG = 77;           //Vtx 9600
9      RCSTAbits.SPEN = 1;   //Encendemos el EUSART
10     TXSTAbits.TXEN = 1;   //Encendemos el transmisor del EUSART
11 }
12
13 void main(void) {
14     EUSART_conf();
15     unsigned char x=0;
16     for(x=0;x<11;x++){
17         TXREG = mensaje[x];
18         while (TXSTAbits.TRMT == 0);
19         __delay_ms(100);
20     }
21     while(1);
22 }
```

9

(cont...)

- Parametrizando y creando funciones específicas para la comunicación serial:

```

1  #include <xc.h>
2  #include "cabecera.h"
3  #define _XTAL_FREQ 48000000UL
4
5  const unsigned char mensaje1[]{"Mi nombre es Kalun Lau"};
6  const unsigned char mensaje2[]{"UPC Electronica Mecatronica"};
7
8  void EUSART_conf(void){
9      SPBRG = 77;           //Vtx 9600
10     RCSTAbits.SPEN = 1;   //Encendemos el EUSART
11     TXSTAbits.TXEN = 1;   //Encendemos el transmisor del EUSART
12 }
13
14 void SERIAL_ESCRIBE_MENSAJE(const unsigned char *cadena, unsigned char tam){
15     unsigned char x=0;
16     for(x=0;x<tam;x++){
17         TXREG = cadena[x];
18         while (TXSTAbits.TRMT == 0);
19     }
20 }
21
22 void SERIAL_NEXTLINE(void){
23     TXREG = 0x0A;
24     while (TXSTAbits.TRMT == 0);
25     TXREG = 0x0D;
26     while (TXSTAbits.TRMT == 0);
27 }
28
29 void main(void) {
30     EUSART_conf();
31     SERIAL_ESCRIBE_MENSAJE(mensaje1,22);
32     SERIAL_NEXTLINE();
33     SERIAL_ESCRIBE_MENSAJE(mensaje2,27);
34     while(1);
35 }
```

10

(cont...)

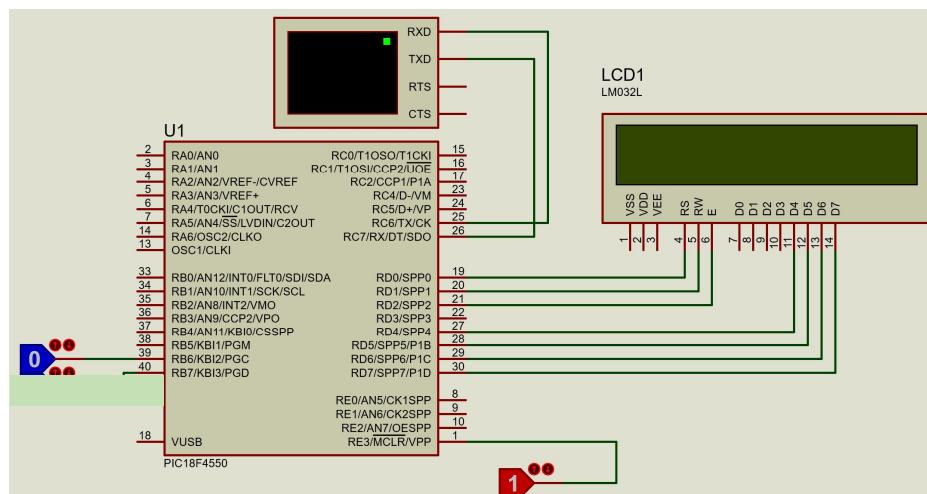
- Empleando la función “strlen” para obtener el numero de caracteres de la cadena y mejorar la función de escritura

```

1  #include <xc.h>
2  #include <string.h>
3  #include "cabecera.h"
4  #define _XTAL_FREQ 48000000UL
5
6  const unsigned char mensaje1[]{"Mi nombre es Kalun Lau"};
7  const unsigned char mensaje2[]{"UPC Electronica Mecatronica"};
8
9  void EUSART_conf(void) {
10    SFBRG = 77;           //Vtx 9600
11    RCSTAbits.SPEN = 1;   //Encendemos el USART
12    TXSTAbits.TXEN = 1;   //Encendemos el transmisor del USART
13 }
14
15 void SERIAL_ESCRIBE_MENSAJE(const unsigned char *cadena){
16    unsigned char tam=0;
17    tam = strlen(cadena);
18    unsigned char x=0;
19    for(x=0;x<tam;x++){
20      TXREG = cadena[x];
21      while (TXSTAbits.TRMT == 0);
22    }
23 }
24
25 void SERIAL_NEXTLINE(void){
26    TXREG = 0x0A;
27    while (TXSTAbits.TRMT == 0);
28    TXREG = 0x0D;
29    while (TXSTAbits.TRMT == 0);
30 }
31
32 void main(void) {
33   EUSART_conf();
34   SERIAL_ESCRIBE_MENSAJE(mensaje1);
35   SERIAL_NEXTLINE();
36   SERIAL_ESCRIBE_MENSAJE(mensaje2);
37   while(1);
38 }
```

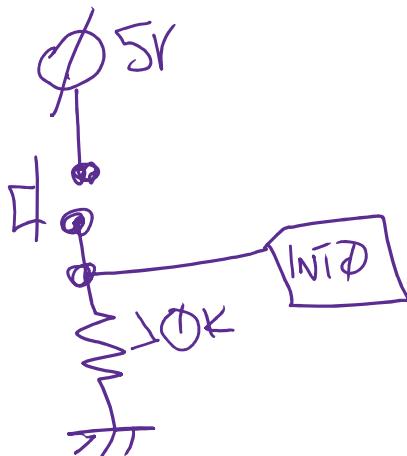
11

Ejemplo: Detectar el estado de RB6 y visualizarlo a través del terminal serial



12

Circuito de pulsador activo en bajo para INT0



13

Código ejemplo:

```

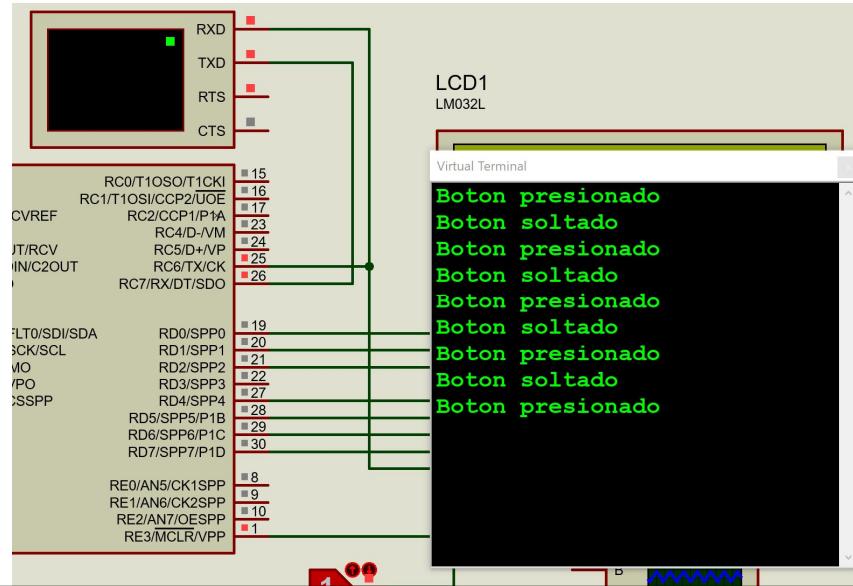
12 #include <xc.h>
13 #define _XTAL_FREQ 48000000UL      //frecuencia
14
15 unsigned char mensaje1[] = {"Boton presionado"};
16 unsigned char mensaje2[] = {"Boton soltado   "};
17 unsigned char indicador = 0;
18
19 void init_conf(void){
20     TRISBbits.RC6 = 0;           //Salida para
21 }
22
23 void EUSART_conf(){
24     SPBRG = 77;                //Vtx = 9600
25     RCSTAbits.SPEN = 1;         //Habilitamos el
26     TXSTAbits.TXEN = 1;         //Habilitamos la
27 }
```

```

29 void main(void){
30     init_conf();
31     EUSART_conf();
32     while(1){
33         if(PORTBbits.RB6 == 1 && indicador == 0){
34             for(unsigned char x=0;x<16;x++){
35                 TXREG = mensaje1[x];
36                 while(TXSTAbits.TRMT == 0); //Esperar a
37             }
38             TXREG = 0x0A;           //Comando para nueva linea
39             while(TXSTAbits.TRMT == 0); //Esperar a que
40             TXREG = 0x0D;           //Comando para retorno de
41             while(TXSTAbits.TRMT == 0); //Esperar a que
42             indicador = 1;
43         }
44         else if(PORTBbits.RB6 == 0 && indicador == 1){
45             for(unsigned char x=0;x<16;x++){
46                 TXREG = mensaje2[x];
47                 while(TXSTAbits.TRMT == 0); //Esperar a
48             }
49             TXREG = 0x0A;           //Comando para nueva linea
50             while(TXSTAbits.TRMT == 0); //Esperar a que
51             TXREG = 0x0D;           //Comando para retorno de
52             while(TXSTAbits.TRMT == 0); //Esperar a que
53             indicador = 0;
54         }
55     }
56 }
```

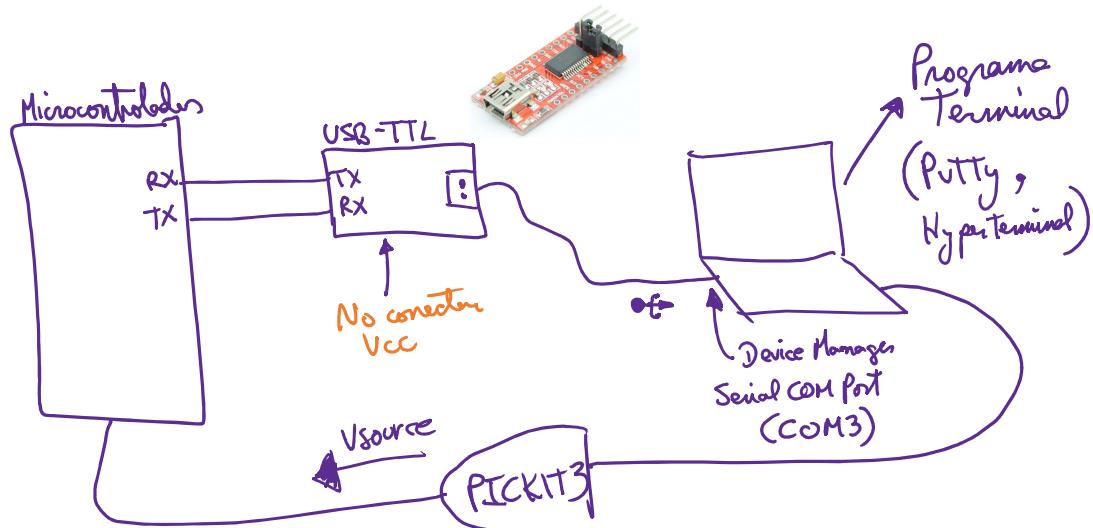
14

Simulación:



15

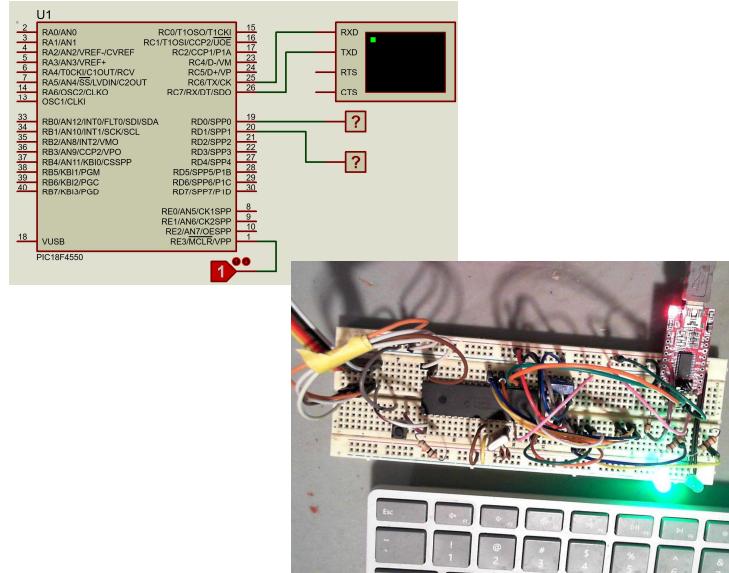
Ejemplo de comunicación serial UART entre el microcontrolador y un terminal serial en la PC



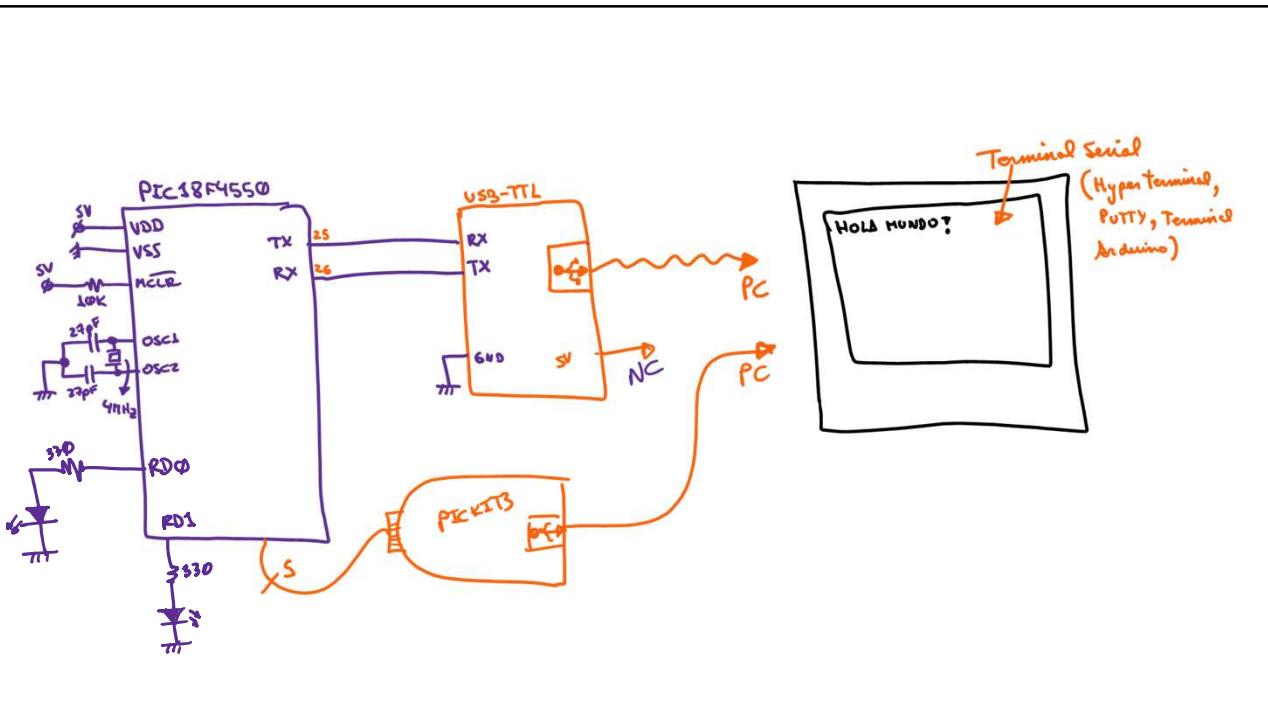
16

Ejemplo de transmisión y recepción en EUSART

- Se han conectado dos LEDs, uno en RD0 y otro en RD1.
- Los LEDs serán controlados a través de la comunicación EUSART con el módulo USB-TTL y conectado a la PC, en la PC corriendo un software de terminal serial (PuTTY) y enviando comandos desde el teclado



17



18

Ejemplo de transmisión y recepción en EUSART (cont...)

```

1 #include <xc.h>
2 #include <string.h>
3 #include "cabecera.h"
4 #define _XTAL_FREQ 48000000UL
5
6 const unsigned char mensaje1[]{"Mi nombre es Kalun Lau"};
7 const unsigned char mensaje2[]{"UPC Electronica Mecatronica"};
8
9 void init_conf(void){
10     INTCONbits.GIE = 1; //interruptor global on
11     INTCONbits.PIE1 = 1; //interruptor de perifericos on
12     PIE1bits.RCIE = 1; //habilitador de int del receptor del EUSART
13     TRISDbits.RD0 = 0;
14     TRISDbits.RD1 = 0;
15 }
16
17 void EUSART_conf(void){
18     SPBRG = 77; //Vtx 9600
19     RCSTAbits.SPEN = 1; //Encendemos el EUSART
20     TXSTAbits.TXEN = 1; //Encendemos el transmisor del EUSART
21     RCSTAbits.CREN = 1; //Encendemos el receptor del EUSART
22 }
23
24 void SERIAL_ESCRIBE_MENSAJE(const unsigned char *cadena){
25     unsigned char tam=0;
26     tam = strlen(cadena);
27     unsigned char x=0;
28     for(x=0;x<tam;x++){
29         TXREG = cadena[x];
30         while (TXSTAbits.TRMT == 0);
31     }
32 }
33
34 void SERIAL_NEXTLINE(void){
35     TXREG = 0x0A;
36     while (TXSTAbits.TRMI == 0);
37     TXREG = 0x0D;
38     while (TXSTAbits.TRMT == 0);
39 }
40
41 void main(void) {
42     init_conf();
43     EUSART_conf();
44     SERIAL_ESCRIBE_MENSAJE(mensaje1);
45     SERIAL_NEXTLINE();
46     while(1);
47 }
48
49 void __interrupt() EUSART_RX_ISR(void){
50     PIR1bits.RCIF = 0;
51     switch(RCREG){
52         case '1':
53             LATDbits.LD0 = 1;
54             SERIAL_ESCRIBE_MENSAJE("Tecla 1 presionada");
55             SERIAL_NEXTLINE();
56             break;
57         case '2':
58             LATDbits.LD0 = 0;
59             SERIAL_ESCRIBE_MENSAJE("Tecla 2 presionada");
60             SERIAL_NEXTLINE();
61             break;
62         case '3':
63             LATDbits.LD1 = 1;
64             SERIAL_ESCRIBE_MENSAJE("Tecla 3 presionada");
65             SERIAL_NEXTLINE();
66             break;
67         case '4':
68             LATDbits.LD1 = 0;
69             SERIAL_ESCRIBE_MENSAJE("Tecla 4 presionada");
70             SERIAL_NEXTLINE();
71             break;
72         default:
73             SERIAL_ESCRIBE_MENSAJE("Tecla erronea");
74             SERIAL_NEXTLINE();
75     }
76 }
77
78 }
```

19

```

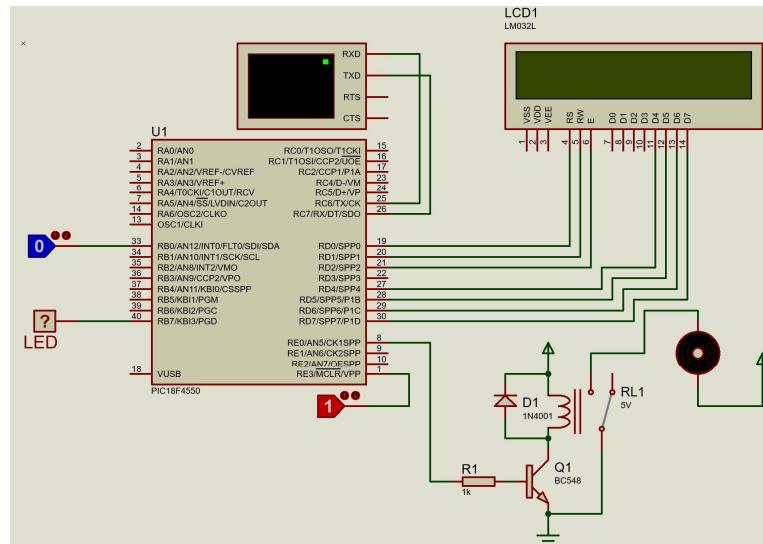
1 #include <xc.h>
2 #include <string.h>
3 #include "cabecera.h"
4 #define _XTAL_FREQ 48000000UL
5
6 const unsigned char mensaje1[]{"Ejercicio Control Remoto"};
7 const unsigned char boton1[] {"Boton INTO presionado"};
8
9 INIT_conf(void){
10     RC0bits.IFEN = 1;
11     INTCONbits.GIEH = 1;
12     INTCONbits.INT0IE = 1;
13     INTCONbits.INT0IF = 1;
14     PIE1bits.RCIE = 1;
15     IFR1bits.RCIP = 0;
16     TRISDbits.RD0 = 0;
17     TRISDbits.RD1 = 0;
18 }
19
20 void EUSART_conf(void){
21     SPBRG = 77;
22     RCSTAbits.SPEN = 1;
23     TXSTAbits.TXEN = 1;
24     RCSTAbits.CREN = 1;
25 }
26
27 void SERIAL_ESCRIBE_MENSAJE(const unsigned char *cadena){
28     unsigned char tam=0;
29     tam = strlen(cadena);
30     unsigned char x=0;
31     for(x=0;x<tam;x++){
32         TXREG = cadena[x];
33         while (TXSTAbits.TRMT == 0);
34     }
35 }
36
37 void SERIAL_NEXTLINE(void){
38     TXREG = 0x0A;
39     while (TXSTAbits.TRMT == 0);
40     TXREG = 0x0D;
41     while (TXSTAbits.TRMT == 0);
42 }
43
44 void main(void) {
45     __delay_ms(500);
46     INIT_conf();
47     EUSART_conf();
48     SERIAL_ESCRIBE_MENSAJE(mensaje1);
49     SERIAL_NEXTLINE();
50     while(1);
51 }
52
53 void __interrupt(high_priority) INTO_ISR(void){
54     INTCONbits.INT0IF = 0;
55     __delay_ms(50);
56     SERIAL_ESCRIBE_MENSAJE(boton1);
57     SERIAL_NEXTLINE();
58 }
59
60 void __interrupt(low_priority) EUSART_RX_ISR(void){
61     PIR1bits.RCIF = 0;
62     switch(RCREG){
63         case '1':
64             LATDbits.LD0 = 1;
65             SERIAL_ESCRIBE_MENSAJE("Tecla 1 presionada");
66             SERIAL_NEXTLINE();
67             break;
68         case '2':
69             LATDbits.LD0 = 0;
70             SERIAL_ESCRIBE_MENSAJE("Tecla 2 presionada");
71             SERIAL_NEXTLINE();
72             break;
73         case '3':
74             LATDbits.LD1 = 1;
75             SERIAL_ESCRIBE_MENSAJE("Tecla 3 presionada");
76             SERIAL_NEXTLINE();
77             break;
78         case '4':
79             LATDbits.LD1 = 0;
80             SERIAL_ESCRIBE_MENSAJE("Tecla 4 presionada");
81             SERIAL_NEXTLINE();
82             break;
83         case '5':
84             LATDbits.LD0 = 1;
85             LATDbits.LD1 = 1;
86             SERIAL_ESCRIBE_MENSAJE("Tecla 5 presionada");
87             SERIAL_NEXTLINE();
88             break;
89         case '6':
90             LATDbits.LD0 = 0;
91             LATDbits.LD1 = 0;
92             SERIAL_ESCRIBE_MENSAJE("Tecla 6 presionada");
93             SERIAL_NEXTLINE();
94             break;
95         default:
96             SERIAL_ESCRIBE_MENSAJE("Tecla erronea");
97             SERIAL_NEXTLINE();
98     }
99 }
```

Código mejorado:

20

Ejemplo de comunicación UART entre el microcontrolador y un terminal serial

- El PIC18F4550 enviará el menú de opciones vía EUSART hacia el terminal virtual a una tasa de 9600 8N1, se tendrá las opciones de encender el LED conectado en RB7



21

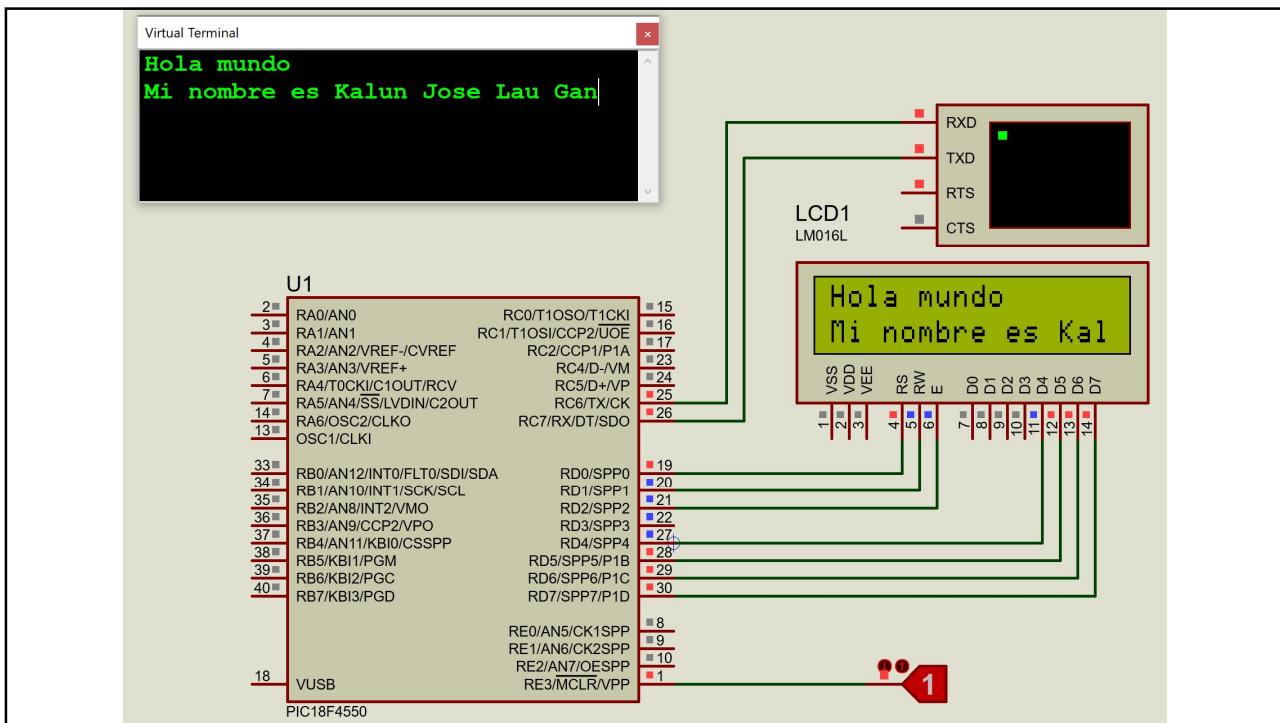
Ejemplo de comunicación UART entre el microcontrolador y un terminal serial

```

1 #pragma config PLLDIV = 1      // PLL Prescaler Selection
2 #pragma config CPUDIV = OSC1_PLL2 // System Clock Frequency Division
3 #pragma config FOSC = XTPLL_XT // Oscillator Selection
4 #pragma config PWRT = ON       // Power-up Timer Enable
5 #pragma config BOR = OFF        // Brown-out Reset
6 #pragma config WDT = OFF        // Watchdog Timer Enable
7 #pragma config CCP2MX = ON      // CCP2 MUX bit (CCP2Mux)
8 #pragma config PBADER = OFF     // PORTA A/D Enable
9 #pragma config MCIRE = ON       // MCLR Pin Enable
10 #pragma config IVP = OFF       // Single-Supply ICSP
11
12 #include <xc.h>
13 #define _XTAL_FREQ 48000000UL //frecuencia de oscilación
14
15 unsigned char menu1[] = {"Bienvenidos al ejemplo"};
16 unsigned char menu2[] = {"*Sección opción "};
17 unsigned char menu3[] = {"*(1) Enciende LED "};
18 unsigned char menu4[] = {"*(2) Apaga LED "};
19 unsigned char menu5[] = {"*(m) Muestra el menu "};
20 unsigned char ledon[] = {"*LED encendido "};
21 unsigned char ledoff[] = {"*LED apagado "};
22
23 unsigned char indicador = 0;
24
25 void PORT_config(void) {
    TRISBbits.RB7 = 0; //Salida en RB7
26 }
27
28 void USART_config(void) {
30     SPBRGH = 0; //Ignorado debido a la velocidad
31     SPBRG = 77; //Tx es 9600 baudios
32     TRISEbits.RE6 = 0; //Puerto RC6 como salida
33     BSCTabits.SREN = 1; //Encendemos el puerto
34     TXSTAbits.TXEN = 1; //Encendemos el transmisor
35     RCSTAbits.CREN = 1; //Encendemos el receptor
36 }
37
38 void INT_config(void){
39     INTCONbits.GIE = 1; //Interruptor global habilitado
40     INTCONbits.PEIE = 1; //Interruptor de perifericos habilitado
41     PIEbits.RCIE = 1; //Habilitado interrupcion por recepcion
42 }
43
44 void USART_siguientelinea(void){
45     TXREG = 0xA0;
46     while(TXSTAbits.TRMT == 0);
47     TXREG = 0x0D;
48     while(TXSTAbits.TRMT == 0);
49 }
50
51 void USART_enviacadena(const unsigned char *vector,unsigned char pos){
52     for (unsigned char x=0;x<pos;x++){
53         TXREG = vector[x];
54         while(TXSTAbits.TRMT == 0);
55     }
56 }
57
58 void show_menu(void){
59     USART_enviacadena(menu1,22);
60     USART_siguientelinea();
61     USART_enviacadena(menu2,22);
62     USART_siguientelinea();
63     USART_siguientelinea();
64     USART_enviacadena(menu3,22);
65     USART_siguientelinea();
66     USART_enviacadena(menu4,22);
67     USART_siguientelinea();
68     USART_enviacadena(menu5,22);
69 }
70
71 void main(void) {
72     PORT_config();
73     USART_config();
74     INT_config();
75     show_menu();
76     while(1);
77 }
78
79 void __interrupt(high_priority) RC_Interrupt(void) {
80     if(RCREG == '1'){
81         LATBbits.LB7 = 1;
82         USART_enviacadena(ledon,22);
83         USART_siguientelinea();
84     }
85     else if(RCREG == '2'){
86         LATBbits.LB7 = 0;
87         USART_enviacadena(ledoff,22);
88         USART_siguientelinea();
89     }
90     else if(RCREG == 0x6D){
91         show_menu();
92     }
93 }
94
95 }
96

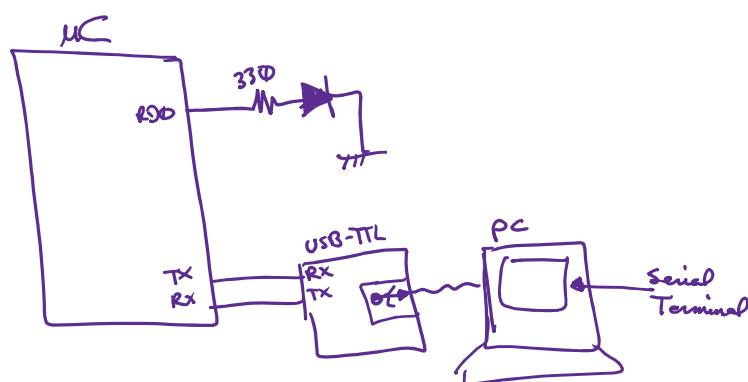
```

22



23

Ejemplo: Comunicación PC hacia PIC18F4550 empleando el USB-TTL



- La cantidad de parpadeos en LED va a estar determinado por el valor ingresado por el teclado de la PC dentro del Serial Terminal.

24

Ejemplo de comunicación UART entre el microcontrolador y un terminal serial

Ejercicios adicionales:

- Agregando una opción para que se ingrese una cadena de caracteres desde el teclado y devuelva por el terminal virtual dicha cadena.
- Cambiando la opción de RB7 por la de RE0 donde esta conectado un relay y un motor DC.