

# Microcontroladores

## Semana 13

Sesión Teoría  
Profesor Kalun José Lau Gan

1

Preguntas Previas

2

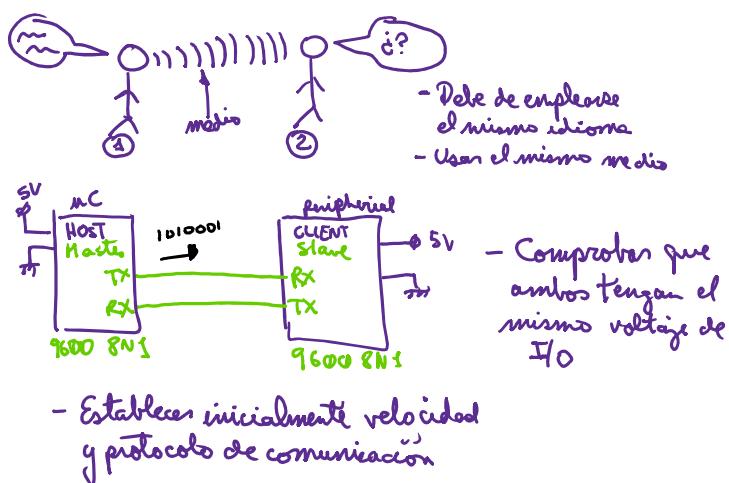
## Agenda

- Comunicación UART
  - Niveles lógicos RS232, TTL, RS485
  - Protocolo de comunicación
- El EUSART del PIC18F4550
  - Generador de baudios
  - Modo transmisión
  - Modo recepción

3

## Comunicación asíncrona UART

- Asíncrona: No hay una señal dedicada de reloj

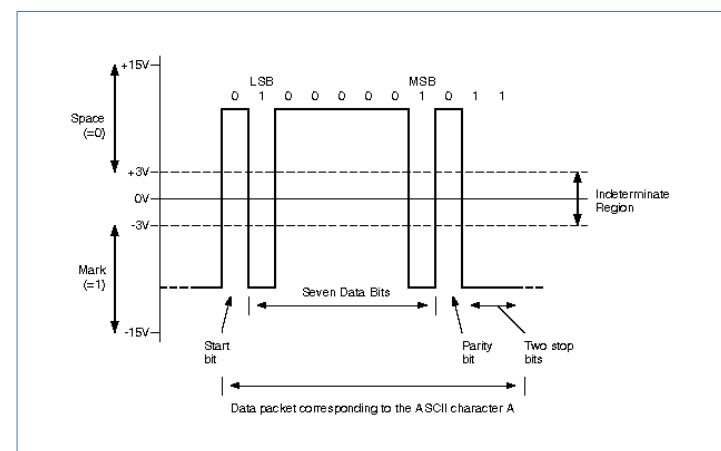
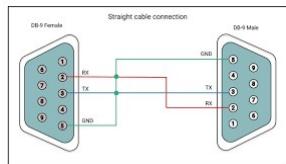
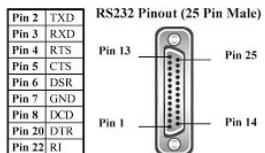


4

## Sobre niveles de voltaje en comunicación UART

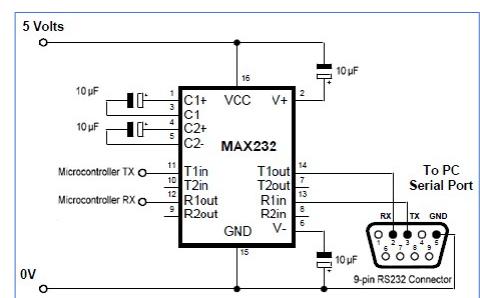
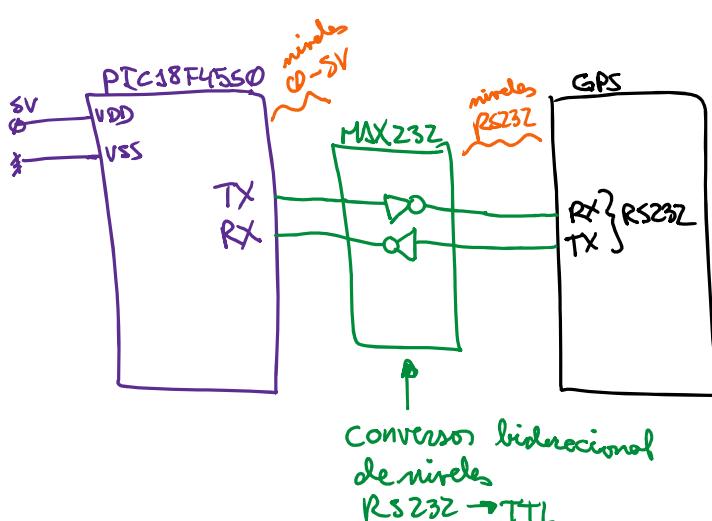
- RS232 (EIA/TIA 232) – Comunicación a distancias medianas (hasta 15 metros)

RS232 25 Pin



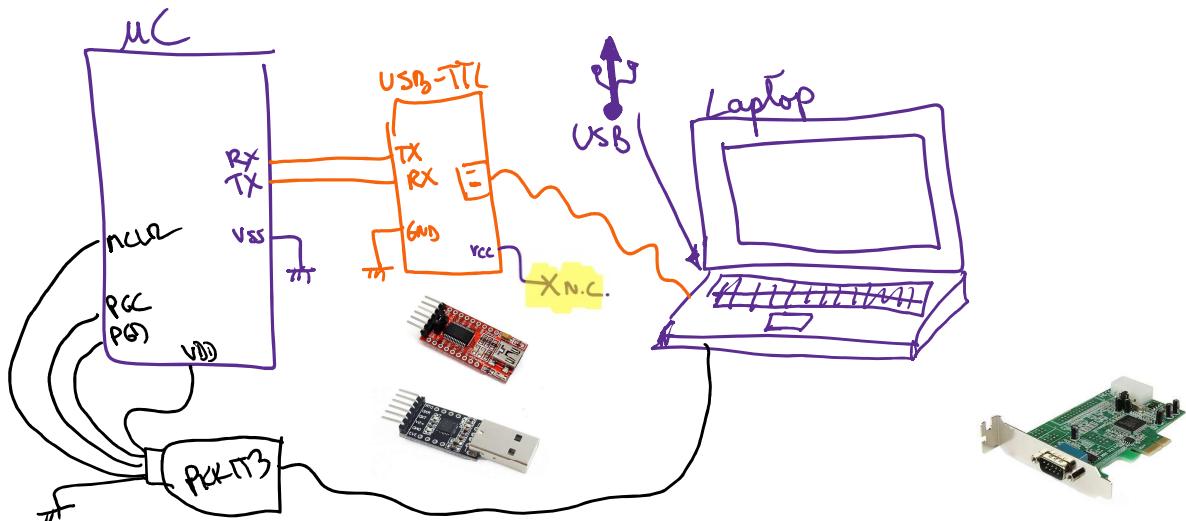
5

## ¿Cómo hago para conectar un dispositivo RS232 al PIC18F4550?



6

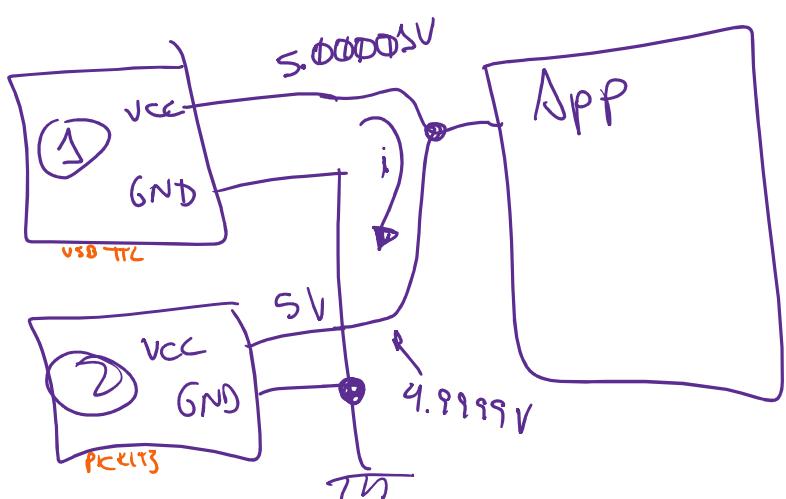
## Cómo conectar un microcontrolador PIC hacia un computador mediante USB



7

Nota: Tener cuidado en no utilizar fuente de alimentación en paralelo.

- El pickit3 suministra la alimentación al circuito de prueba, el conversor USB-TTL también tiene una línea de alimentación que viene directo del puerto USB por lo que solo deben de usar una fuente, o la del pickit3 o la del USB-TTL



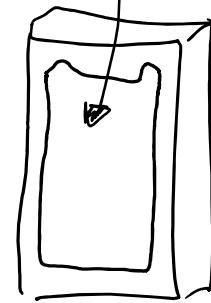
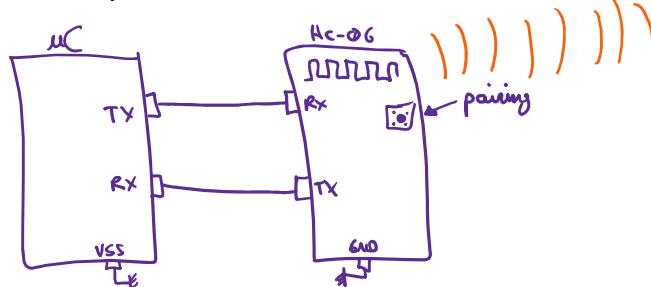
8

## Módulo Bluetooth HC-06



APP Terminal (End)

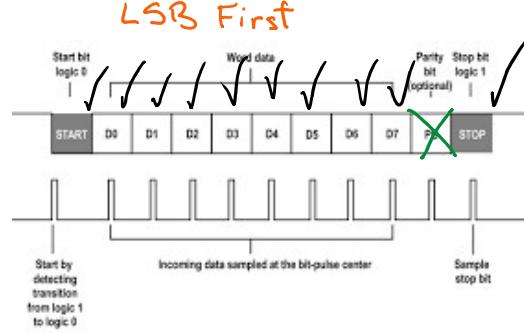
- Interface de comunicación UART a niveles TTL (5V)
- Velocidad de comm por defecto: 9600 8N1



9

## Cálculo de la velocidad en comunicación UART

- Formato completo es
  - Velocidad – Protocolo
  - Ej. 9600 - 8N1
- Protocolo común: 8N1
  - 8: El dato es de 8 bits
  - N: no paridad (O:paridad impar, E:paridad par)
  - 1: un bit de stop



- En total se están enviando 10 bits por cada dato de 8 bits

10

## Cálculo de la velocidad en comunicación UART

- Si tengo  $\overset{V_{TX}}{9600}$  8N1:

$$\text{Tiempo de bit: } T_{\text{bit}} = \frac{1}{V_{TX}} = \frac{1}{9600} = 1.04 \times 10^{-4} \text{ s} \\ = 0.1 \text{ ms}$$

- Si para enviar un dato de 8 bits usamos 10 bits:

$$\text{Tiempo para enviar un dato de 8bit} = 1.04 \times 10^{-4} \text{ s} = 1.04 \text{ ms.}$$

11

## Ejemplo de cálculo de cuánto tiempo demora en enviar datos vía comm. serial:

- ¿Cuánto demoro en enviar un archivo de 100KByte por una canal de comm. serial a 9600 8N1?

Recordar que 1 byte = 8 bit, 1Kbyte = 1024 bytes

1º Hallar cuántos bits se va a enviar:

$$100 \times 1024 = 102400 \text{ bytes}$$

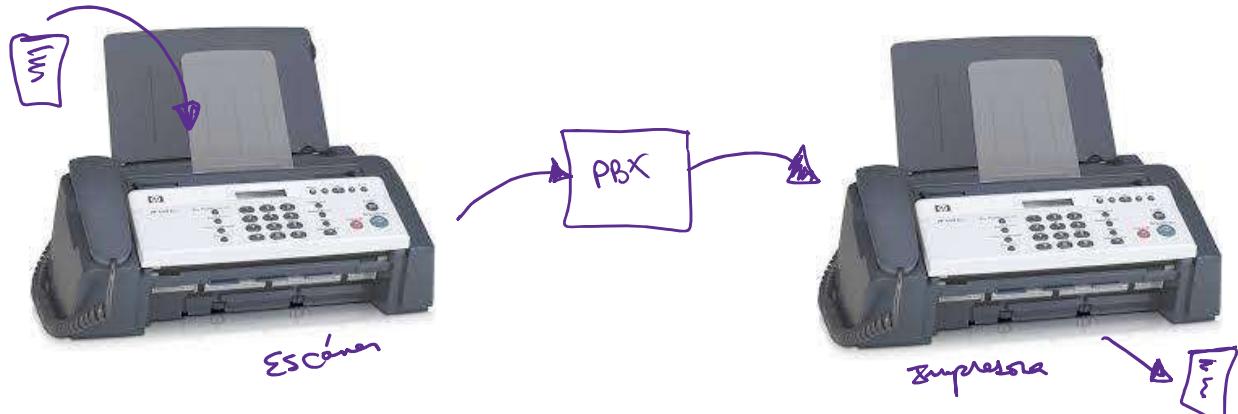
Cada byte representará 80 bits enviados  $\Rightarrow 102400 \times 80$  bits enviados

$$\text{Tiempobit}_{9600} = 1.04 \times 10^{-4}$$

$$\text{Tiempo requerido} = 102400 \times 80 \times 1.04 \times 10^{-4} \text{ segundos}$$

12

## Recordando las máquinas FAX



- Transmitía el documento ingresado por línea telefónica a 9600 8N1

13

Ejemplo de cálculo de cuánto tiempo demora en enviar datos vía comm. serial:

- ¿Cuánto demoro en enviar un archivo de música de 4MByte por una canal de comm. serial a 57600 8N1?

5 minutos

14

Ejemplo de cálculo de cuánto tiempo demora en enviar datos vía comm. serial:

- ¿Cuánto demoro en enviar un archivo de música de 4MByte por una canal de comm. serial a 57600 8N1?

$$4\text{MByte} \Rightarrow 4 \times 1024 = 4096 \text{kbytes}$$

$$4096 \text{k} \times 1024 = 4194304 \text{ bytes}$$

$$\text{Cont. bits a Transmíter: } 4194304 \times 10 = 41943040 \text{ bits}$$

$$\text{Tiempo de bit} \Rightarrow \frac{1}{57600} = 17.361 \mu\text{s}$$

$$\text{Tiempo para transmitir los bits: } 728.177 \mu\text{s}$$

12 minutos y 8.177 segundos

15

## El modulo EUSART

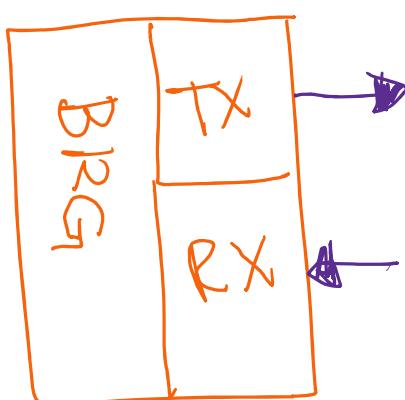


FIGURE 20-3: EUSART TRANSMIT BLOCK DIAGRAM

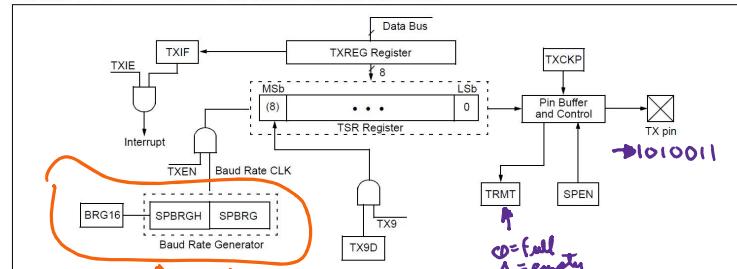
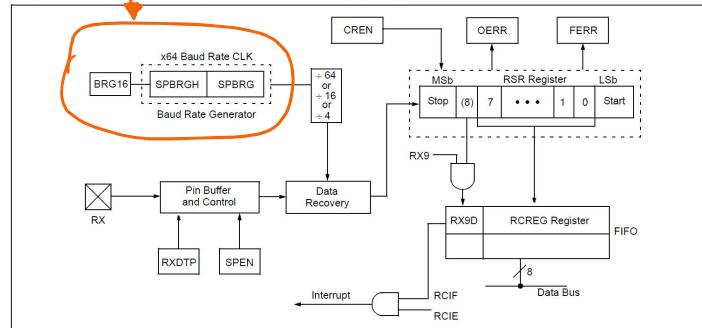


FIGURE 20-6: EUSART RECEIVE BLOCK DIAGRAM



16

## Para transmitir un dato:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate. ✓
2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN. ✓
3. If the signal from the TX pin is to be inverted, set the TXCKP bit. ✗
4. If interrupts are desired, set enable bit, TXIE. ✗
5. If 9-bit transmission is desired, set transmit bit, TX9. Can be used as address/data bit. ✗
6. Enable the transmission by setting bit, TXEN, which will also set bit, TXIF. ✓
7. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D. ✗
8. Load data to the TXREG register (starts transmission). ✓
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set. ✗

10. Esperar a que TRMT cambie de estado ✓

Para el cálculo de BRG:

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

Si Vtx = 9600  $\Rightarrow$  BRG = ? despija 'n'

$$m = \frac{(\text{Fosc})}{64} - 1$$

$n = \underbrace{\text{SPBRGH:SPBRG}}_{\text{registro 16 bit}}$

$$m = \frac{48000000}{9600} - 1$$

pero como bit BRG16 = 0

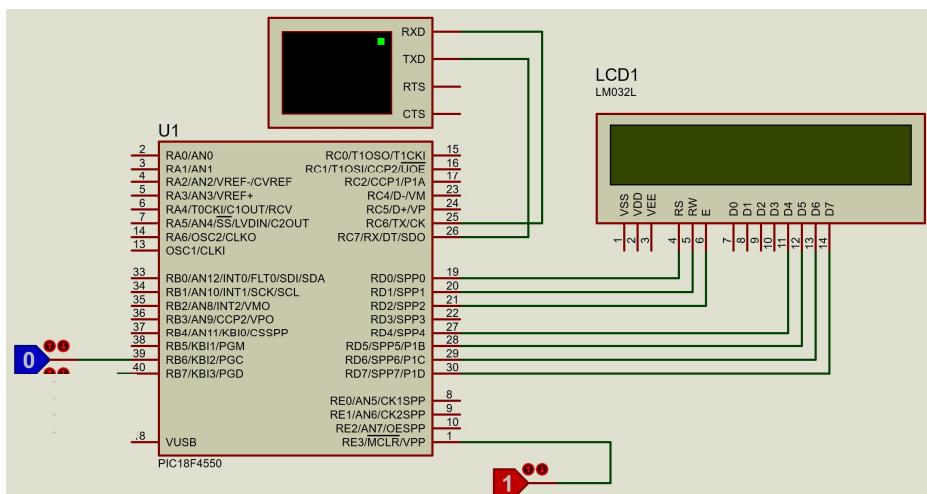
$$m = 77.125 \rightarrow \text{SPBRG} = 77$$

$$\text{Baudrate actual} = \frac{\text{Fosc}}{(\text{SPBRG} + 1)} = \frac{48000000}{[64(77+1)]} = 9615$$

$$\text{Bitrate error: } \frac{9615 - 9600}{9600} \times 100 = 0.16\%$$

17

## Circuito de prueba



18

## 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     TRISChbits.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 }

```

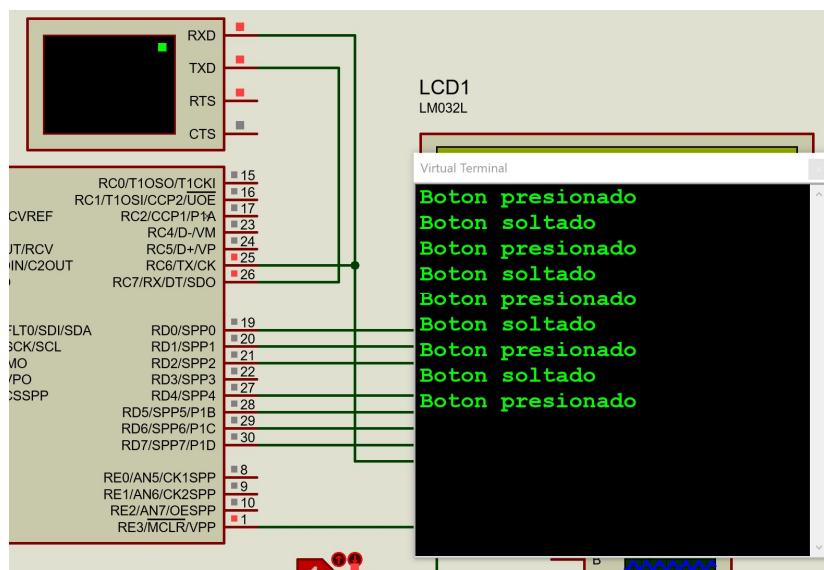
```

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 }

```

19

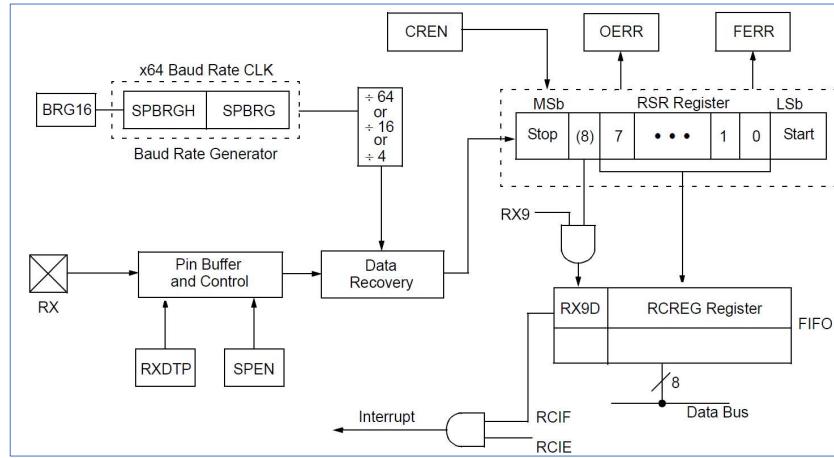
## Simulación:



20

## Receptor del EUSART

- Diagrama de bloques



21

## Receptor del EUSART

- Emplea el mismo SPBRG que el transmisor.
- Seguir el procedimiento de configuración de la hoja técnica
- **Se recomienda el uso de interrupciones en esta etapa de recepción ya que el receptor carece de FIFO.**

22

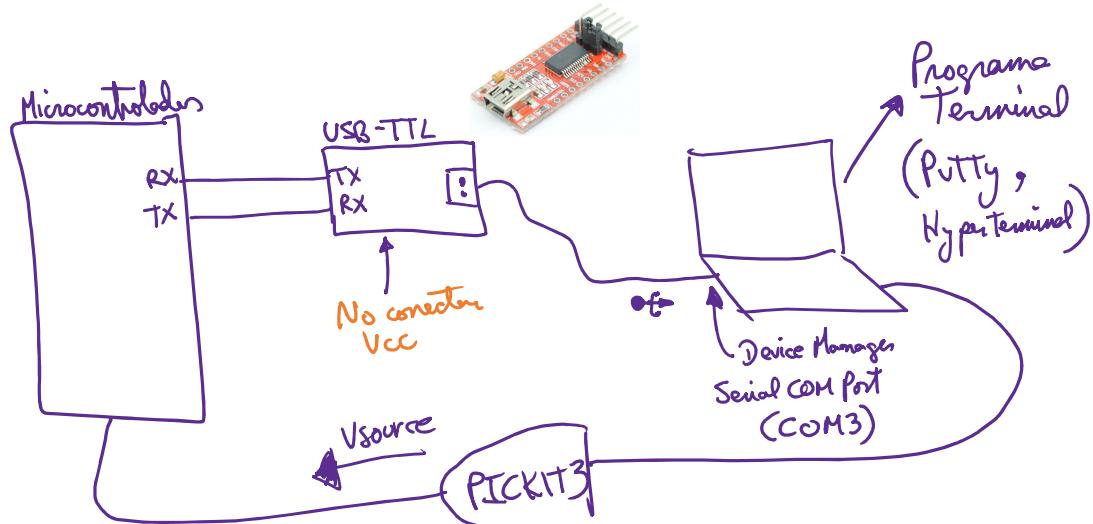
## Receptor del EUSART

- Procedimiento para recibir un dato (según hoja técnica)

- To set up an Asynchronous Reception:
1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
  2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN.
  3. If the signal at the RX pin is to be inverted, set the RXDTP bit.
  4. If interrupts are desired, set enable bit, RCIE.
  5. If 9-bit reception is desired, set bit, RX9.
  6. Enable the reception by setting bit, CREN.
  7. Flag bit, RCIF, will be set when reception is complete and an interrupt will be generated if enable bit, RCIE, was set.
  8. Read the RCSTA register to get the 9th bit (if enabled) and determine if any error occurred during reception.
  9. Read the 8-bit received data by reading the RCREG register.
  10. If any error occurred, clear the error by clearing enable bit, CREN.
  11. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

23

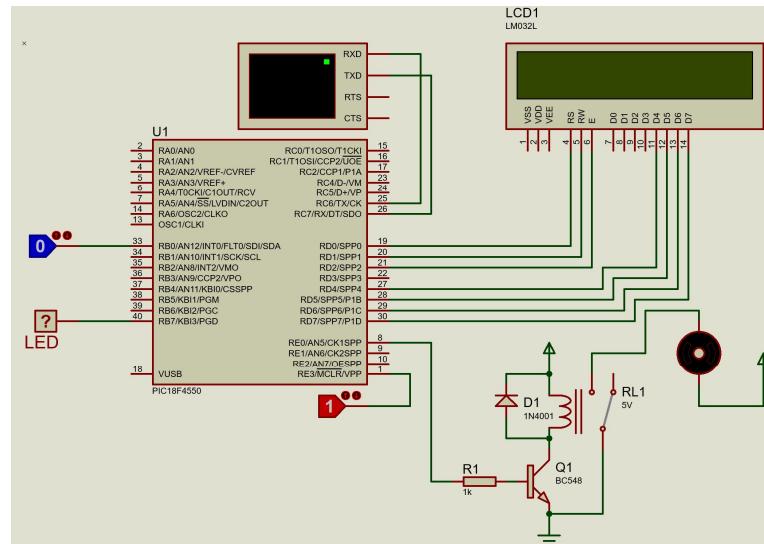
## Ejemplo de comunicación UART entre el microcontrolador y un terminal serial



24

## 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



25

## 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        // PORTB 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[] = {"Selecctione opcion"};
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 EUSART_config(void){
    SPBRGH = 0;                 //Ignorado debido a la velocidad
    SPBRG = 77;                  //Tx es 9600 baudios
    TRISEbits.RC6 = 0;            //Puerto RC6 como salida
    RCSTAbits.SREN = 1;           //Encendemos el puerto
    TXSTAbits.TXEN = 1;           //Encendemos el transmisor
    RCSTAbits.CREN = 1;           //Encendemos el receptor
29 }
30
31 void INT_config(void){
    INTCONbits.GIE = 1;           //Interruptor global habilitado
    INTCONbits.PEIE = 1;           //Interruptor de perifericos habilitado
    PIE1bits.RCIE = 1;             //Habilitado interrupcion por recepcion
32 }
33
34 void EUSART_siguientelinea(void){
    TXREG = 0xA7;                //Enviamos el caracter A
    while(TXSTAbits.TRMT == 0);   //Esperamos que termine de transmitir
    TXREG = 0x0D;                //Enviamos el caracter NUEVO LINEA
    while(TXSTAbits.TRMT == 0);   //Esperamos que termine de transmitir
35 }
36
37 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); //Esperamos que termine de transmitir
    }
38 }
39
40 void show_menu(void){
    EUSART_enviacadena(menu1,22);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu2,22);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu3,22);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu4,22);
    EUSART_siguientelinea();
    EUSART_enviacadena(menu5,22);
    EUSART_siguientelinea();
41 }
42
43 void main(void) {
    PORT_config();
    EUSART_config();
    INT_config();
    show_menu();
    while(1);
44 }
45
46 void __interrupt(high_priority) RC_Isr(void){
    PIR1bits.RC1F = 0;
    if(RCREG == '1'){
        LATBbits.LB7 = 1;
        EUSART_enviacadena(ledon,22);
        EUSART_siguientelinea();
    }
    else if(RCREG == '2'){
        LATBbits.LB7 = 0;
        EUSART_enviacadena(ledoff,22);
        EUSART_siguientelinea();
    }
    else if(RCREG == 0x6D){
        show_menu();
    }
47 }
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95 }

```

26

## Ejemplo de comunicación UART entre el microcontrolador y un terminal serial

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.