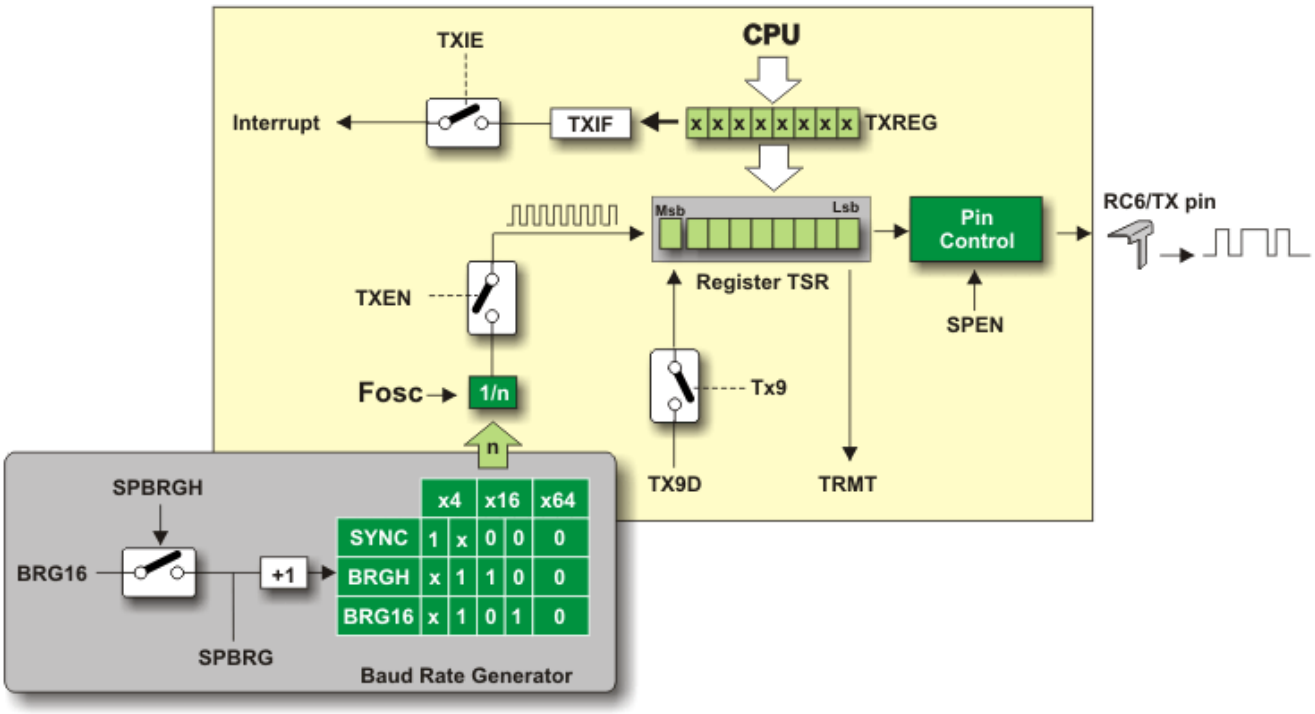
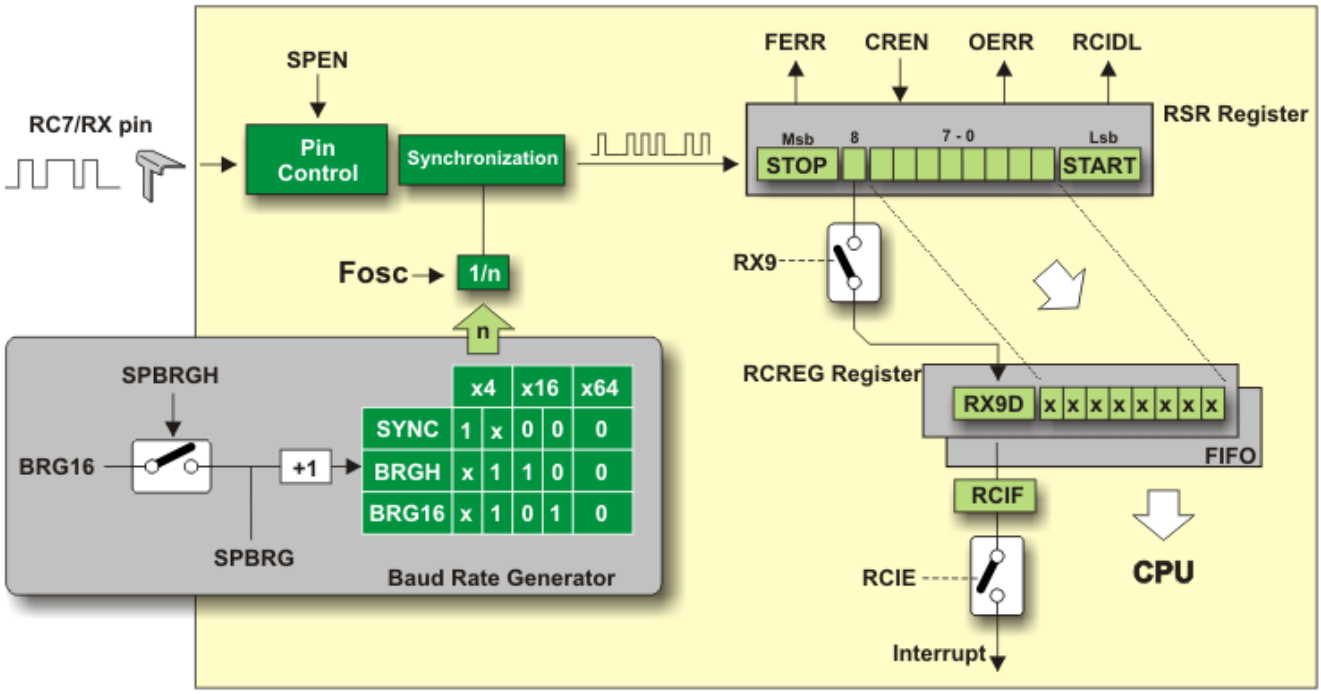
PROGAMACION USART

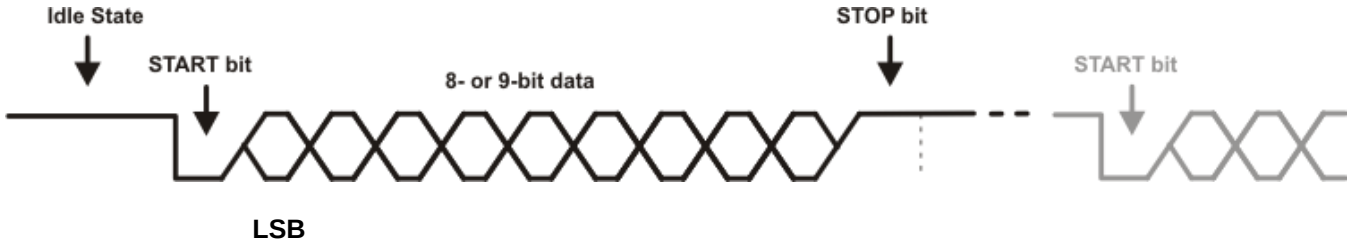
MODULO USART TRANSMISOR



MODULO USART RECEPTOR



Transmisión Asíncrona



Enviando datos:

1. Los baudios a los cuales transmitirá son colocados usando los bits BRGH (Registro TXSTA) y BRG16 (Registro BAUDCTL) y Registros SPBRGH y SPBRG.

2. El bit SINC (TXSTA) se debe poner a 0 y el SPEN debe setearse.

3. Si se transmiten 9 bits, debe colocarse a 1 TX9.

4. La transmisión de datos se habilita colocando a 1 el TXEN. El TXIF del PIR1 se pone en 1 automáticamente.

5. Si se trabaja con interrupciones el bit TXEN la habilita si GIE y PEIE del INTCON son 1.

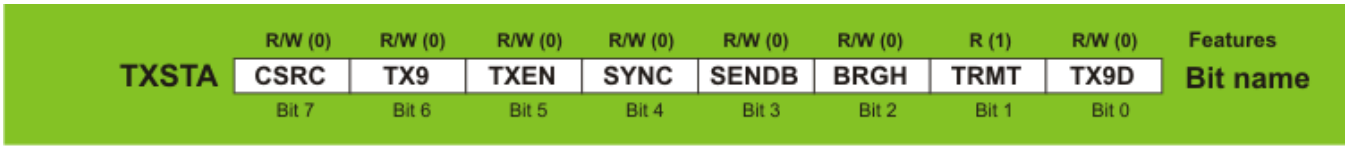
6. Si se transmite con 9 bits de datos, el valor del novena bit debe escribirse en TX9D.

7. La transmisión comienza escribiendo datos de 8 bits en el TXREG.

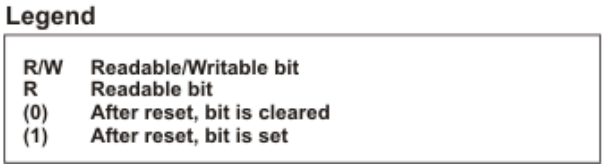
Recibiendo datos:

1. Los baudios a los cuales transmitirá son colocados usando los bits BRGH (Registro TXSTA) y BRG16 (Registro BAUDCTL) y Registros SPBRGH y SPBRG.
2. El bit SINC (TXSTA) se debe poner a 0 y el SPEN debe setearse (RCSTA register) para habilitar el Puerto.
3. Si trabaja con interrupciones el bit RCIE del PIE1 la habilita si GIE y PEIE del INTCON son 1
4. Si recibe 9 bit de datos el RX9 bit del RCSTA register debe ser 1.
5. Los datos recibidos deben ser habilitados poniendo 1 en bit CREN del RCSTA register.
6. El Registro RCSTA puede ser leído para obtener información sobre errores ocurridos durante la transmisión.
7. Los 8 bit de datos almacenados en Registro RCREG debieran ser leídos.

Registro TXSTA:



	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R (1)	R/W (0)	Features
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	



Legend	
R/W	Readable/Writable bit
R	Readable bit
(0)	After reset, bit is cleared
(1)	After reset, bit is set

CSRC – Selecciona Fuente Clock – determina fuente clock. Sólo en modo sincrónico. (1 – Master mode: clock interno, 0 – Slave mode: clock de fuente externa).

TX9 – Habilitación transmisión 9no. bit (1 – 9 bits, 0 – 8 bits)

TXEN – Habilitación Transmisión (1 – Transmisión habilitada, 0 – Transmisión deshabilitada)

SYNC – Selecciona Modo EUSART (1 – EUSART opera en modo sincrónico, 0 – EUSART opera en modo asincrónico)

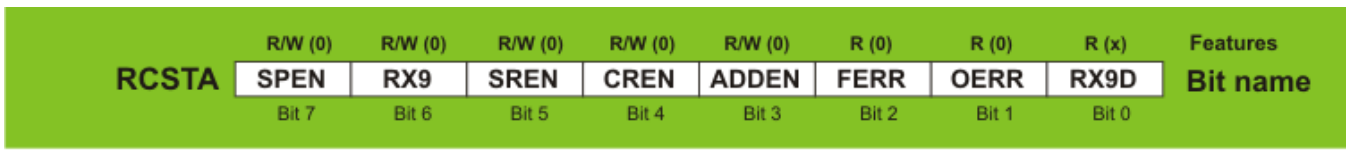
SENDB – Caracter Break enviado (1 – Enviando character Break, 0 – Se completó envío Caracter Break), Sólo en modo asíncrono y cuando lo requiera un bus estándar.

BRGH – Secciona Baud Rate Alto (1 – EUSAT opera a alta velocidad, 0 – EUSART opera a baja velocidad). Sólo en modo asincrónico.

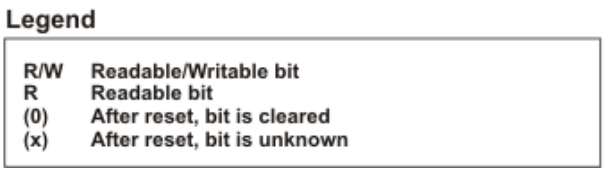
TRMT – Status del Shift Register del Transmisor (1 – Si TSR está vacío, 0 – si TSR está lleno)

TX9D – Noveno bit de datos (puede ser usado como dirección o bit de paridad)

Registro RCSTA



	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R/W (0)	R (0)	R (0)	R (x)	Features
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	



Legend	
R/W	Readable/Writable bit
R	Readable bit
(0)	After reset, bit is cleared
(x)	After reset, bit is unknown

SPEN – Serial Port Enable bit

- 1 – Puerto serie habilitado. Pines RX/DT y TX/CK pins se configuran automáticamente como entrada y salida respectivamente
- 0 – Puerto serie deshabilitado.

RX9 – 9-bit Receive Enable bit

- 1 – Recibe 9 bit de datos via EUSART
- 0 – Recibe 8 bit de datos via EUSART

SREN – Single Receive Enable bit se usa solo en modo sincrónico cuando el micro opera como Master

- 1 – Sólo habilitado para recepción
- 0 – Deshabilitado para recepción

CREN – Continuous Receive Enable bit actúa diferente dependiendo del modo

Modo Asíncrono:

- 1 – Recepción habilitada
- 0 – Recepción deshabilitada.

Modo Sincrónico:

- 1 – Habilita recepción continua hasta que el bit se pone en 0
- 0 – Deshabilita recepción continua.

ADDEN – Address Detect Enable bit se usa solo en modo detecta direcciones

- 1 – Habilita detección de dirección cuando recibe 9 bits
- 0 – Deshabilita detección de dirección. Bit 9 es paridad.

FERR – Framing Error bit

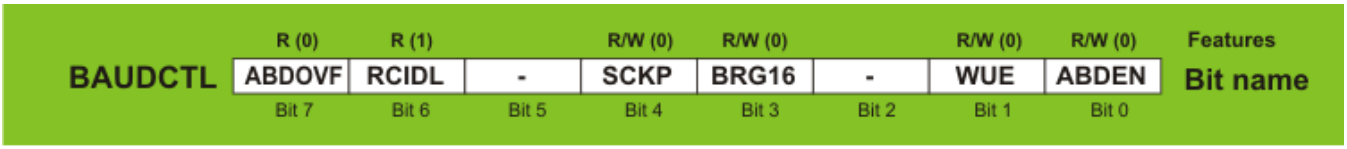
- 1 – En recepción detecta error de framing
- 0 – No hay error de framing

OERR – Overrun Error bit.

- 1 – En recepción se detecta desbordamiento
- 0 – No hay error por desbordamiento

RX9D – Ninth bit of Received Data puede ser usado como dirección o bit de paridad

Registro BAUDCTL



	R (0)	R (1)		R/W (0)	R/W (0)		R/W (0)	R/W (0)	Features
BAUDCTL	ABDOVF	RCIDL	-	SCKP	BRG16	-	WUE	ABDEN	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

Legend

-	Bit is unimplemented
R/W	Readable/Writable bit
R	Readable bit
(0)	After reset, bit is cleared
(1)	After reset, bit is set

ABDOVF – Auto-Baud Detect Overflow bit: se usa sólo en modo asincrónico durante la detección del baud rate

- 1 – Auto-baud timer rebalsó
- 0 – Auto-baud timer no rebalsó.

RCIDL – Receive Idle Flag bit se solo en modo asincrónico.

- 1 – Receptor está ocioso
- 0 – el bit START hs sido recibido y la recepción está en progreso

SCKP – Synchronous Clock Polarity Select bit: actúa diferente según sea modo asíncrono o síncrono

Modo Asíncrono:

- 1 – Transmite datos invertidos en el pin RC6/TX/CK
- 0 – Transmite datos no invertidos en ese pin

Modo Síncrono:

- 1 – Se sincroniza sobre el flanco ascendente del clock
- 0 – Se sincroniza sobre el flanco descendente del clock

WUE Wake-up Enable bit

- 1 – El receptor espera por un flanco descendente en pin RC7/RX/DT pin para sacar al micro del modo sleep.
- 0 – El receptor opera normalmente.

ABDEN – Auto-Baud Detect Enable bit se usa solo en modo asincrónico

- 1 – Modo de detección Auto-baudio habilitada. El bit se pone automáticamente en cero cuando se detecta el baud rate
- 0 – Modo de detección Auto-baudio deshabilitada.

Generador de Baudios – Baud Rate

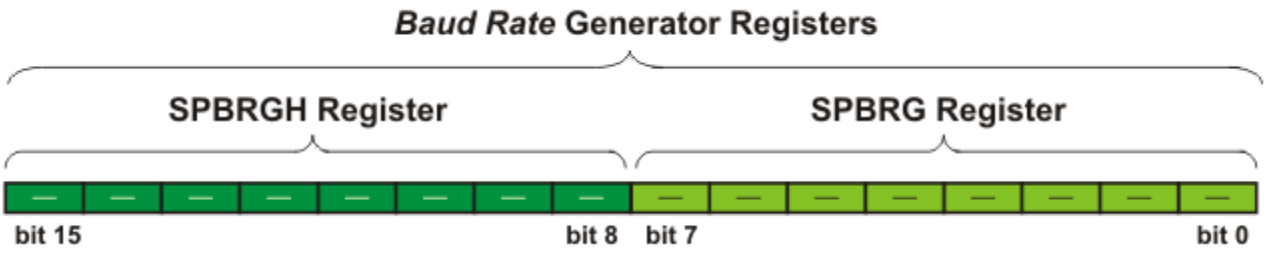
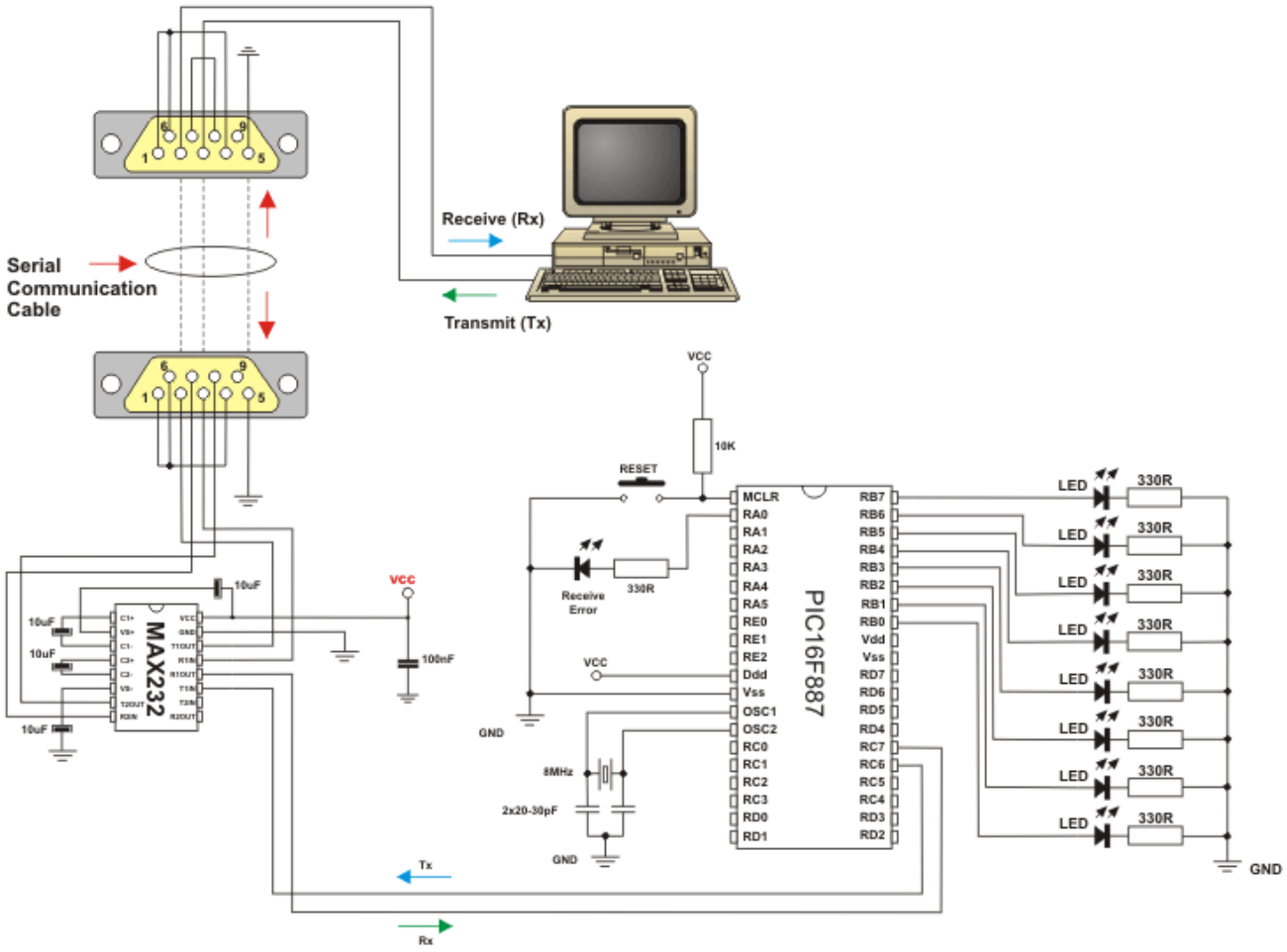


Tabla que determina el Baud Rate:

BITS			BRG / EUSART MODE	BAUD RATE FORMULA
SYNC	BRG16	BRGH		
0	0	0	8-bit / asynchronous	Fosc / [64 (n + 1)]
0	0	1	8-bit / asynchronous	Fosc / [16 (n + 1)]
0	1	0	16-bit / asynchronous	Fosc / [16 (n + 1)]
0	1	1	16-bit / asynchronous	Fosc / [4 (n + 1)]
1	0	X	8-bit / synchronous	Fosc / [4 (n + 1)]
1	1	X	16-bit / synchronous	Fosc / [4 (n + 1)]

Programa Ejemplo:

Este programa muestra cómo se usa el módulo EUSART del PIC16F887. La conexión a la PC se realiza mediante una conexión RS232 estándar. El programa trabaja de la siguiente manera: Cada byte recibido vía comunicación serie se muestra usando diodos LEDs conectado al Puerto B y es automáticamente retornado al transmisor después. Si ocurre un error en la recepción esto se indicará encendiendo el LED correspondiente.



```
;********************** Header ****************************************
;               DEFINING VARIABLES IN PROGRAM

w_temp          EQU     0x7D        ; Variable for saving W register
status_temp     EQU     0x7E        ; Variable for saving STATUS register
pclath_temp     EQU     0x7F        ; Variable for saving PCLATH w register

cblock                  0x20        ; Block of variables starts at address 20 h
Port_A                              ; Variable at address 20 h
Port_B                              ; Variable at address 21 h
RS232temp                           ; Variable at address 22 h
RXchr                               ; Variable at address 23 h
endc                                ; End of block of variables

;*****************************************************************************
                ORG     0x0000      ; Reset vector
                nop
                goto    main        ; Go to beginning of program (label "main")

;*****************************************************************************
                ORG     0x0004      ; Interrupt vector address

                movwf   w_temp      ; Save value of W register
                movf    STATUS,w    ; Save value of STATUS register
                movwf   status_temp
                movf    PCLATH,w    ; Save value of PCLATH register
                movwf   pclath_temp

;*****************************************************************************
; This part of the program is executed in interrupt routine
                banksel PIE1
                btfss   PIE1, RCIE
                goto    ISR_Not_RX232int
                banksel PIE1
                btfsc   PIR1, RCIF
                call    RX232_int_proc

ISR_Not_RX232int
                movf    pclath_temp,w
                movwf   PCLATH      ; PCLATH is given its original value

                movf    status_temp,w
                movwf   STATUS      ; STATUS is given its original value
                swapf   w_temp,f
                swapf   w_temp,w    ; W is given its original value

                retfie              ; Return from interrupt routine
```

```
;*****************************************************************
RX232_int_proc                          ; Check if error has occurred
                banksel     RCSTA
                movf        RCSTA, w
                movwf       RS232temp
                btfsc       RS232temp, FERR
                goto        RX232_int_proc_FERR
                btfsc       RS232temp, OERR
                goto        RX232_int_proc_OERR
                goto        RX232_int_proc_Cont

RX232_int_proc_FERR
                bcf         RCSTA, CREN     ; To clear FERR bit, receiver is first
                                            ; switched off and on afterwards
                nop                         ; Delay ...
                nop
                bsf         RCSTA, CREN
                movf        RCREG, w            ; Reads receive register and clears
                                                ;FERR bit
                bsf         Port_A, 0           ; Switches LED on ( UART error
                                                ;indicator)
                movf        Port_A, w
                movwf       PORTA
                goto        RS232_exit

RX232_int_proc_OERR
                bcf         RCSTA, CREN         ; Clears OERR bit
                nop                             ; Delay ...
                nop
                bsf         RCSTA, CREN
                movf        RCREG, w            ; Reads receive register and clears
                                                ;FERR bit
                bsf         Port_A, 1           ; Switches LED on (UART error indicator)
                movf        Port_A, w
                movwf       PORTA
                goto        RS232_exit

RX232_int_proc_Cont
                movf        RCREG, W            ; Reads received data
                movwf       RXchr
                movwf       PORTB
                movwf       TXREG               ; Sends data back to PC

RS232_exit
                return                          ; Return from interrupt routine
```

```
;*****************************************************************************
; Main program

main
                banksel     ANSEL           ; Selects bank containing ANSEL
                clrf        ANSEL           ; All inputs are digital
                clrf        ANSELH

    ;--------------------------------------
    ; Port configuration
    ;--------------------------------------
                banksel     TRISA
                movlw       b'11111100'
                movwf       TRISA
                movlw       b'00000000'
                movwf       TRISB
    ;--------------------------------------
    ; Setting initial values
    ;--------------------------------------
                banksel     PORTA
                movlw       b'11111100'
                movwf       PORTA
                movwf       Port_A
                movlw       b'00000000'
                movwf       PORTB
                movwf       Port_B
    ;--------------------------------------
    ; USART - setting for 38400 bps
    ;--------------------------------------
                banksel     TRISC
                bcf         TRISC, 6        ; RC6/TX/CK = output
                bsf         TRISC, 7        ; RC7/RX/DT = input

                banksel     BAUDCTL
                bsf         BAUDCTL, BRG16
                banksel     SPBRG
                movlw       .51             ; baud rate = 38400
                                            ; ( Fosc/(4*(SPBRG+1)) ) Error +0.16%
                movwf       SPBRG
                clrf        SPBRGH

                banksel     TXSTA
                bcf         TXSTA, TX9      ; Data is 8-bit wide
                bsf         TXSTA, TXEN     ; Data transmission enabled
                bcf         TXSTA, SYNC     ; Asynchronous mode
                bsf         TXSTA, BRGH     ; High-speed Baud rate


                banksel     RCSTA
                bsf         RCSTA, SPEN     ; RX/DT and TX/CK outputs configuration
```

```
                bcf             RCSTA, RX9          ; Select mode for 8-bit data receive
                bsf             RCSTA, CREN         ; Receive data enabled
                bcf             RCSTA, ADDEN        ; No address detection, ninth bit may be
                                                    ; used as parity bit
                movf            RCREG, W            ;cleared RCIF bit

                blanksel        BAUDCTL
                bsf             BAUDCTL, SCKP       ;set inverted mode

;--------------------------------------
; Interrupts enabled
;--------------------------------------
                banksel         PIE1
                bsf             PIE1, RCIE          ; USART Rx interrupt enabled

                bsf             INTCON, PEIE        ; All peripheral interrupts enabled
                bsf             INTCON, GIE         ; Global interrupt enabled

;--------------------------------------
; Remain here
;--------------------------------------
                goto            $

                end                                 ; End of program
```

Cálculo del Baud Rate del ejemplo:

Baud Rate Deseado: 38.400 baudios

$$38.400 = [Fosc / [4 (n + 1)] \qquad \text{donde } n = SPBRG$$

$$38.400 = [Fosc / [4 (SPBRG + 1)]$$

$$SPBRG = \frac{(Fosc / 38.400) - 4}{4}$$

$$SPBRG = \frac{8x10E6 / 38.400 - 4}{4}$$

$$SPBRG = 51,08333$$