Discrete Logic Replacement

Asynchronous Serial Transmit and Receive

Author: *Jose Luiz Pinto Souto*
H & S Projetos
Florian Ûpolis Brazil
email: souto@cryogen.com

APPLICATION OPERATION

This brief is an Asynchronous Serial Transmit & Receive routine to communicate at speeds up to 115200 bps.

It requires two pins, one for TxD and another for RxD. EPROM usage equals 57 words; RAM usage equals 3 bytes and can be shared with other routines, since they are volatile. The baudrate timing is dependent on the number of instructions executed. For best BPS match, select a crystal like 3.6864 MHz. There is no support for interrupts. The main code has to pool the RxD pin for start bit Low. One suggestion is to receive a BREAK before the communication starts to let the firmware detect it and immediately call `RxSerial` to get the incoming bytes. If the protocol allows, the firmware may start the communication and you may not need the BREAK.

APPENDIX A: SOURCE CODE

```
                PROCESSOR       12C508
                RADIX           HEX
                LIST            C=132,N=0,X=OFF
                TITLE           "Serial Communication routines"
                NOEXPAND
                LIST
;*****************************************************************************
; sio.asm
;*****************************************************************************
;
; (c) Copyright 1995, 1997 Jose Luiz Pinto Souto
;
;     Av. Almirante Lamego 748-D/504
;     88015-600, Florianopolis, SC, Brazil
;
;     Tel   : +55 (48) 223-7595
;             +55 (48) 244-2698
;
;     e-mail: souto@cryogen.com
;
; Contact me before using this code in commercial applications.
;
; Those routines were used to transmit and receive asynchronously at
; speeds up to 115200 bps and implemented in two commercial products.
; It depends on the CPU Crystal. I've been using the 3.57 MHz crystal
; for easy availability. For best BPS match try using the 3.6864 MHz
; crystal.
;
; I used it with PIC16C5x CPUs, and I had no interrupts to detect the start
; bit. But, to let the firmware know that Host wants to start sending bytes,
; I forced Host to send a BREAK for, let's say, 10 bytes-time to allow for the
; firmware main-loop detect it. After it, Host starts sending a block of
; bytes to firmware. No errors were reported. It Works.
; (In case you don't know how to force a break, try switching the Host speed
; to 600 bps {for 9600 comm.} and send one 0x00).
;
; The routines rely on the number of instructions to send and receive bits.
; In order to use other crystals, remember to recalculate the number of
; cycles necessary to your communication speeds.
;
; In this fragment, the routine is set @ 9600 bps w/a 3.57 MHz crystal.
;
; Any two unused ports may be used. This code fragment doesn't program
; the ports as _TxOUT:output & _RxIN:Input.
;
; Three bytes are used as scratch and may be shared with some other routines
; with a careful design.
;
;*****************************************************************************
; sio.asm
;*****************************************************************************

                INCLUDE P12CXX.INC

; those Define numbers are only an example

#DEFINE         _TxOUT          PORTA,0
#DEFINE         _RxIN           PORTA,1

txBuf           EQU             0x08
rxBuf           EQU             0x09
idx             EQU             0x0A

;---------------------------------{ TxSerial }-------------------------------*
```

```
; Function : Transmit a byte @ 9600 bps
;-------------------------------------------------------------------------*
; Transmits 1 start bit Lo, 8 data bits & 1 stop bit Hi
; No parity implemented (up to you).
; Byte time = 1.040mS.
; Bit duration 93T = 104.038 uS (1.24% error w/3.57 Mhz crystal)
;
; At each shift right, a Hi bit is inserted in the transmit buffer. After 8
; data bits the stop bit Hi will be transmitted automatically since counter
; "idx" started with 10.
;
; Input    : W = byte to be transmitted
;
; Output   : byte transmitted by serial port
;
; Variables: txBuf,
;            rxBuf,
;            idx.
;
; Date     : 27/Jul/95 JLPS
; Revision : 22/Nov/95 JLPS
; EPROM    : 22 words
;-------------------------------------------------------------------------*


TxSerial:       movwf   txBuf           ; save byte to transmit
                movlw   d'10'
                movwf   idx             ; start counter w/10 bits

; start Tx_ing w/start bit Lo in carry

                clrc                    ; Start bit = 0
                goto    TxB_2           ; skip the rrf

; Tx loop

TxB_1:          rrf     txBuf,f         ; 1T    bit0 -> Carry

TxB_2:          skpnc                   ; 1/2T  Tx bit "0" ?
                goto    TxB_4           ; 2T    no, skip ("1")
                setc                    ; 1T    set carry & waste 1T

; [93T]
; [0T] tx bit 0

                bcf     _TxOUT          ; 1T    TxD pin = 0
                movlw   d'28'           ; 1T    28 for 9600 match
                movwf   rxBuf           ; 1T    used as a scratch counter

; [3T]

TxB_3:          decfsz  rxBuf,f         ; 1/2T
                goto    TxB_3           ; 2T

; [3T + 27*3T+2T] = [3T + 83T] = [86T] - bit 0
; [6T + 26*3T+2T] = [6T + 80T] = [86T] - bit 1

                decfsz  idx,f           ; 1/2T  tx all 10 bits ?
                goto    TxB_1           ; 2T    no, back
; [89T]
                return

; tx bit 1

; [93T]
; [0T]
```

```
TxB_4:          bsf     _TxOUT          ; 1T    TxD pin = 1
                movlw   d'27'           ; 1T    27 for 9600 match
                movwf   rxBuf           ; 1T    used as a scratch counter
                nop                     ; 1T    1T for match 93T
                goto    TxB_3           ; 2T    back


;--------------------------------{ RxSerial }-----------------------------*
; Function : Receive a byte @ a 9600 bps
;-------------------------------------------------------------------------*
; Reads start bit LO,  8 data bits and the stop bit at 93T rate (9600 bps)
; No parity implemented (up to you).
; Byte time = 1.040mS.
; Bit time : 93T = 104.038 uS (1.24% error W/3.57 Mhz crystal)
;
; Check Start bit & and false start
;
; The timeout for the start bit is 4.3 mS.
;
; Input    : *
;
; Output   : Carry Set   - success
;            rxBuf,w     - data byte
;            Carry Clear - timeout error or ou stop bit = 0.
;
; Variables: rxBuf,
;            txBuf,
;            idx.
;
; Date     : 27/Jul/95 JLPS
; Revision : 25/Nov/95 JLPS (4,3ms)
; EPROM    : 35 words (can be 31 w/ just 1.43 mS timeout)
;-------------------------------------------------------------------------*

RxSerial:       movlw   d'3'
                movwf   txBuf           ; timeout - 4.29 ms (3*1.43 mS)
                clrf    idx             ; timeout - 1.43 mS

; 5T (5,59uS) loop : wait for "start_bit low"

RxS_1:          btfss   _RxIN           ; 2/1T  Start bit Lo ?
                goto    RxS_2           ; 2T    yes, skip
                decfsz  idx,f           ; 1/2T  no, count time
                goto    RxS_1           ; 2T    back to hunt

; 255*5T+4T = 1279T (1430.8uS)

                decfsz  txBuf,f         ; 1/2T
                goto    RxS_1           ; 2T

; 2*(1279T+3T)+(1279T+2T) = 2*1279T+1281T = 3839T = 4294.65 uS

                clrc                    ; timeout
                return

; detect false start bit

RxS_2:          btfsc   _RxIN           ; 1/2T  still "0" ?
                goto    RxS_1           ; 2T    no, false start

; delay of [139.5T] = 1 & 1/2 bits (139T in fact)
; to start reading in the middle of each bit.
; we already wasted 5T after the start-bit

RxS_3:          movlw   d'43'           ; 1T    43 for 1 & 1/2 delay
```

```
                movwf   idx             ; 1T
RxS_4:          decfsz  idx,f           ; 1/2T
                goto    RxS_4           ; 2T

                clrf    rxBuf           ; 1T    clear Rx buffer
                movlw   h'01'           ; 1T
                movwf   txBuf           ; 1T    mask to receive 9 bits
                clrc                    ; 1T    carry starts w/"0"

; [5T+2T+(42*3T+2T)+4T] = [5T+2T+(126T+2T)+4T] = 139T

; bit reading loop @ 93T rate (attention:incoming bit in bit.1)

RxS_5:          bsf     txBuf,1         ; 1T    assume bit 1
                btfss   _RxIN           ; 1/2T  RxD pin = "1" ?
                bcf     txBuf,1         ; 1T    no, bit "0"
                rrf     txBuf,f         ; 1T
                rrf     rxBuf,f         ; 1T
                movlw   d'28'           ; 1T    28 for 9600 match
                movwf   idx             ; 1T

; [7T]

RxS_6:          decfsz  idx,f           ; 1/2T
                goto    RxS_6           ; 2T

; [7T+(27*3T+2T)]=90T

                skpc                    ; 1/2T  9 bits readed ?
                goto    RxS_5           ; 2T    no, back

; end of 9 bits

                rrf     txBuf,w         ; stop bit -> carry
                movfw   rxBuf           ; W <- rxBuf : data byte
                return


; That's All folks

             END
```

NOTES: