EE 2004
Week 11 Homework
Solution

1. Assuming the clock frequency is 4MHz, the baud rate generator of the UART module is chosen to operate in 8-bit mode, and BRGH is set to 1, calculate the values required to be loaded to SPBRG to program the following baud rates:
    a. 1200
    b. 2400
    c. 9600
    d. 19200
    e. 57600

Use the equation $f_{osc}/16(X+1)$. *=> X=(fosc/16/Baud)-1*

Baud Rate	Value in Decimal x	Hexidecimal	SPBRG
1200	207	CF	CF
2400	103	67	67
9600	25	19	19
19200	12	C	C
57600	3	3	3

2. Change BRG16 to 1 and maintain the rest of the settings in Question 1, calculate the values required to be loaded to SPBRGH:SPBRGL for the baud rates listed in Question 1.

Use the equation $f_{osc}/4(X+1)$.

Baud Rate	Value in Decimal	Hexidecimal	SPBRGH	SPBRGL
1200	832	340	3	40
2400	415	19F	1	9F
9600	103	67	0	67
19200	51	33	0	33
57600	16	10	0	10

3. Assume the clock frequency is 4MHz and we are using the 8-bit mode of the BRG to generate a baud rate of 57600 bps when BRGH = 1. Answer the following questions:
    a. What is the value that is required to put in the SPBRG register?
    b. What is the % error of the baud rate? Is this error acceptable? If not, what change can you make to keep it in an acceptable range?

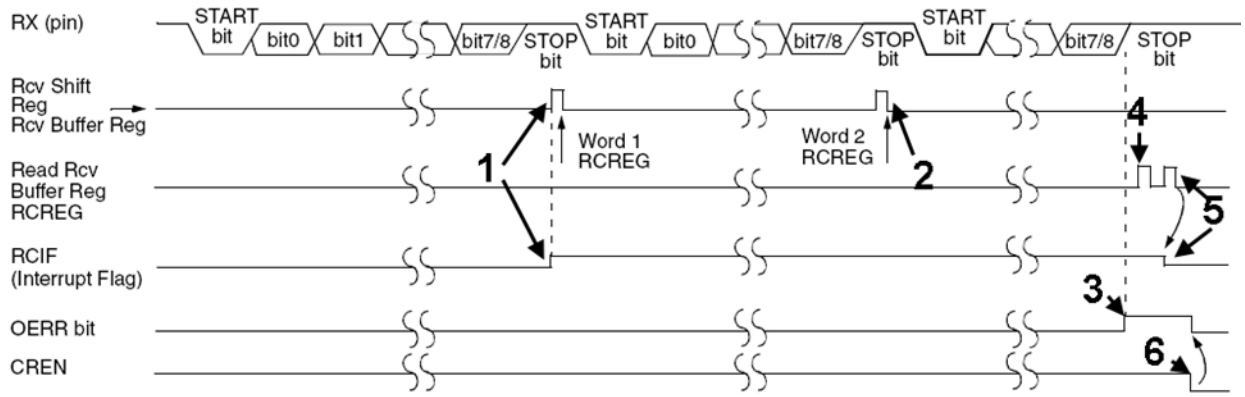
    For 8 bit mode
    Actual Baud = 4MHz/16/(3+1) =62499
    %error =(62499-57600)/57600=8.51%>5%

    For 16bit mode
    Actual Baud =4MHz/4/(16+1) =58823
    %error =(58823-57600)/57600=2.12%<5% acceptable

err%	62500	8.51%	$f_{osc}/16(X+1)$
err%	58823	2.12%	$f_{osc}/4(X+1)$

a. *3 (decimal)*
b. *8.51%. Not acceptable because error > 5%. Set BRG16=1 and SPBRGH:SPBRG = 16 (decimal). The error % is reduced to 2.12%, which is acceptable.*

4. The following diagram is the timing diagram for UART reception. Overflow error occurs at Time Point 3. Explain why overflow error occur at this time point, but not earlier or later? If the OERR flag is raised, what implication does it have with UART reception? How to clear the OERR flag?



See Section 7.1 course notes

- *OERR raised: word 3 arrives before 1st & 2nd in RCREG Buffer being read*
- *Word 3 lost*
- *Read word 1 & 2, reset OERR by clearing CREN after RCIF is cleared*

5. Write a program to send the message “Hello world” through the UART serial port.

- Initialize transmission:
  - Set two bits in TXSTA register:
    - TXEN = 1 – to enable the TX
    - SYNC = 0 – asynchronous TX
    - BRGH optional to 1for 16 bit mode
  - PIR1 register response
    - TXIF (TXREG empty) flag is set automatically when TXEN is set)
  - Set bits in RCSTA
    - SPEN = 1 – to enable the UART module
  - Set SPBRG register with the required baud rate
- Start Transmission:
  - Wait for TRMT flag in TXSTA register is set
  - Write to TXREG register via Working Register WREG.
    - Two Events:
      - Content of TXREG is transferred to TSR register. TXIF is set again.
      - TSR is filled. TRMT (in TXSTA register) flag becomes 0.
    - A byte, framed between the start and stop bit, is transmitted out.
    - Once the stop bit has been sent out, TRMT becomes 1 and a new data can be transmitted.

CSRC	TX9	TXEN	SYNC	0	BRGH	TRMT	TX9D

SPEN	RX9	SREN	CREN	ADDE	FERR	OERR	RX9D

```
                ORG         0x000000
                goto  Main  ;go to start of main code

;-----------------------------------------------------------------
;Start of main program

;
Main:           movlw b'00100100'; TXEN = 1, SYNC = 0, BRGH = 1
                movwf TXSTA
                movlw b'10010000'; SPEN = 1, CREN = 1 optional
                movwf RCSTA
                bcf BAUDCON, BRG16        ;8 bit mode
                movlw 25                  ; baud rate 9600
                movwf SPBRG

                ; Initialize TBLPTR
                movlw upper message
                movwf TBLPTRU
                movlw high message
                movwf TBLPTRH
                movlw low message
                movwf TBLPTRL
; main loop
Loop:           tblrd*+                   ; load data from table
                movf TABLAT, W
                bz Done; If null string is detected, exit
                       ;program
                call putChar
```

```
                bra Loop

Done:           bra $

;transit to UART-----------------------------------------
putChar:        btfss TXSTA, TRMT
                bra   putChar
                movwf  TXREG
                return

                org 0x000500
message db "Hello world', 0

                END
```