```
;;;
;;; 485 tutorial
;;; r <raffi@media.mit.edu>
;;; 2/11/03
;;;

        LIST    p = 16f876

#include p16f876.inc

        __config        _HS_OSC & _WDT_OFF & _PWRTE_ON & _BODEN_OFF & _CP_OFF & _LVP_OFF

LED_GREEN       EQU     0
LED_RED         EQU     1

PIN_SRX         EQU     7
PIN_STX         EQU     6
PIN_SDE         EQU     5

TIMER0          EQU     0x20    ; use this for a "clock"


;;;
;;; MACROS
;;;

#define BIT(i) (1<<(i))


;;; a macro to move a literal into a memory address
MVL     MACRO   l, dest
        MOVLW   l
        MOVWF   dest
        ENDM


;;; a macro to transmit out a literla
TXL     MACRO   l
        MOVLW   l
        CALL    tx
        ENDM



;;;
;;; PROGRAM CODE
;;;

        ORG     0x0000
        GOTO    init
        ORG     0x0004
        RETFIE

init:
        CLRF    STATUS
        CLRF    PORTC

        CLRF    T1CON           ; setup to use TIMER1
        CLRF    TMR1H
        CLRF    TMR1L
        CLRF    INTCON

        BSF     STATUS, RP0     ; bank 1
        MVL     BIT(PIN_SRX), TRISC ; port c outputs except rx pin

        CLRF    PIE1            ; disable peripheral irrupts

        ;; setup USART at 38.4kbps
        MVL     d'32', SPBRG
        MVL     BIT(TXEN) | BIT(BRGH), TXSTA
        BCF     STATUS, RP0     ; bank 0
        MVL     BIT(SPEN) | BIT(CREN), RCSTA
        BCF     RCSTA, CREN     ; reset the USART to be in receive
        BSF     RCSTA, CREN

        ;; start TIMER1 and lets start keeping track of time
        CLRF    PIR1
        MVL     BIT(T1CKPS1) | BIT(T1CKPS0) | BIT(TMR1ON), T1CON
        MVL     d'10', TIMER0

        BSF     PORTC, LED_GREEN; turn on the LEDs
        BSF     PORTC, LED_RED

main:
        BTFSC   PIR1, TMR1IF    ; see if the clock has "ticked"
        GOTO    timer
_main_after_timer:
        BTFSC   RCSTA, OERR     ; if there is a serial error, then reset
        GOTO    rx_reset
        BTFSC   PIR1, RCIF      ; if there is a received byte, then deal
        GOTO    rx
_main_after_serial:
        GOTO    main            ; restart the main loop


;;; handle the timer tick -- whenever it does tick (approx once a second then
;;; we are going to transmit out Hi!
timer:
        BCF     PIR1, TMR1IF    ; reset the timer
        MVL     d'10', TIMER0
        TXL     d'72'           ; H
        TXL     d'105'          ; i
        TXL     d'33'           ; !
        TXL     d'13'           ; /CR
        TXL     d'10'           ; /LF
        GOTO    _main_after_timer


;;; handle the incoming bytes -- we just flash the LED when that happens
rx:
        MOVF    RCREG, W
        BTFSC   RCSTA, FERR
        GOTO    rx_reset

        MOVLW   BIT(LED_RED)    ; flash the red light
        XORWF   PORTC, F
        GOTO    _main_after_serial


;;; reset the serial port
rx_reset:
        BCF     RCSTA, CREN
        BSF     RCSTA, CREN
        GOTO    _main_after_serial


;;; the transmit character
tx:
        BCF     RCSTA, CREN     ; turn off receive on serial
        BSF     PORTC, PIN_SDE  ; enable transmit
        MOVWF   TXREG           ; move W -> TXREG
```

```
        BSF     STATUS, RP0     ; bank 1
_tx_wait:
        BTFSS   TXSTA, TRMT     ; wait until TRMT is high
        GOTO    _tx_wait
        BCF     STATUS, RP0     ; bank 0
        BCF     PORTC, PIN_SDE  ; disable transmit
        BSF     RCSTA, CREN     ; enable receive on serial
        MOVLW   BIT(LED_GREEN)  ; flash the green light
        XORWF   PORTC, F
        RETURN

        END
```