EE2361 - Lecture 21
10/26/16

Return HW 2

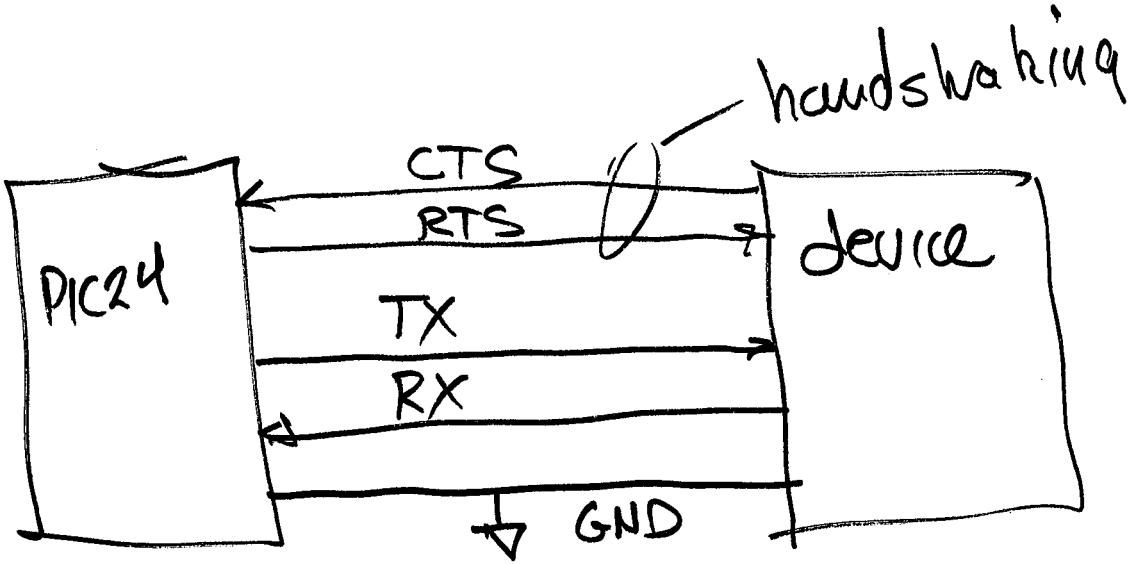
UART

Ref: See UART example in Lucio Di Jasio, Programming 16-bit Microcontrollers in C (chapter on asynchronous communication)

Example : Set up the PIC24 UART

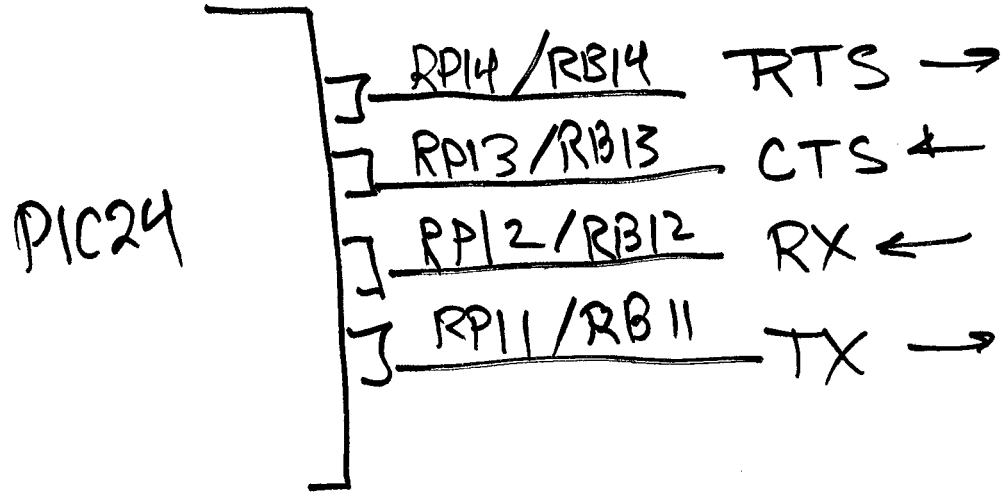
to do: Baud rate of 19200 (bits/sec)
8 data bits
No Parity
One Stop bit
Hardware handshaking
(bit banging)

⇒ use 5-wire RS-232



We configure this by setting bits in the registers!

Use PIC24FJ64GA002



Usual Overhead to our c-code

```
#include <xc.h>
(configuration pragmas)
16 MHz, Tcy = 62.5 ns

#define CTS _RB13 // CTS
#define RTS _RB14 // RTS
```

Finding the value to put in the Baud Rate Register

$\Rightarrow F_{cy} = 16\,MHz$

From the datasheet or FRM (Family Ref Manual)

$$\text{Baud Rate} = \frac{F_{cy}}{16 \cdot (Ux\,BRG + 1)} \qquad x = 1 \text{ for UART1}$$

$$\Rightarrow U1BRG = \frac{F_{cy}}{16 \cdot \text{Baud Rate}} - 1 = \frac{16 \times 10^6}{16 \times 19200} - 1$$

$$= 51.08$$

Need an integer value (which produces the least error 52 or 51)

May need to see which produced the least error

$$\Rightarrow \text{Baud Rate} = \frac{16 \times 10^6}{16 \times (51+1)} = 19230$$

$$\text{Error?} \quad \frac{19230 - 19200}{19200} = 0.001563$$

$$\Rightarrow 0.1563\% \text{ error}$$

Use BRGH=0

This is is much less than 3%

so OK we use 51

To set up the UART1

1. Initialize U1BRG with 51
2. Set the number of data bits (8), number of stop bits (1), and parity (none) in U1MODE
3. Interrupts configured if desired
4. enable the UART.
5. Enable transmission, sets the U1TXIF flag
6. write the data to U1TXREG

3 registers are used:

3 registers

```
U1BRG = 0x0033;     // set to 51
U1MODE = 0x1000;    // enable UART,
U1STA = 0x0400;        8 data bits, 1 stop, no parity
```

Also need to correctly set PPS and pin direction and make sure it is digital

---

→ May need need to unlock pins if they are locked

Function to send a character with RTS/CTS flow control

```
int sendU1(int c)
{
    while(CTS);          // clear to send if low?
    while(U1STAbits.U1TXBF); // buffer empty?
    U1TXREG = c;         // write data to buffer
    return c;
}
```

How many instruction cycles to send a 10-bit frame at 1923 bits/sec?

$$\frac{16 \times 10^6 \text{ instructions/sec}}{1923 \text{ frames/sec}} = 8320.3328$$

no. of instructions while we are sending

So we waste a lot of time (instr cycles) if there ~~are~~ other things to do

Receive function

```
char receiveU1(void)
{
    RTS = 0;        // ready, assert low
    while(!U1STAbits.URIDA); wait
    RTS = 1;        // deassert
    return U1RXREG; // read the receive
                       buff.
}
```

Friday start SPI