PIC to PC terminal through USART

Objective

This project provides a 'main' program that illustrates how to connect a PIC18F45k20 microcontroller to a PC through USART. The program transmits a simple text string through a 9-pin to USB connector. Tera-Term is the program used on Windows to display the information passed through the serial port, though there are a number of programs that will serve the same purpose

Requirements

The following hardware and software

Hardware	Software
• Breadboard • Assorted Jumper Wires • PIC18F45K20 microcontroller • PICKIT 3 programmer • USB to RS-232 serial interface	• MPLAB X v2.20 • XC8(v1.33) compiler for MPLAB X • Tera Term (terminal to display transmitted data)

Functional Descriptions

In this tutorial we have opted to create a simple 'main' program as opposed to functions that can simply be copied into any use. The reason for this is that when dealing with any sort of serial connection there are a number of parameters that you will want to independently manipulate to suit your particular needs. Such parameters include baud rate, buffer size, and parity bits will all vary from project to project.

The implementation of this project is fairly simple, the important and difficult portion is configuring the registers pertaining to serial communication. Mainly, TXSTA, RCSTA, BAUDCON, OSCCON, and SPBRG. Each of these registers plays an integral part in the functioning of your serial connection. We will give a brief description of what each one is used for, but you will need to refer to the data sheet for the pic18f45k20 to completely understand these registers.

Useful link to learn about registers:
http://ww1.microchip.com/downloads/en/DeviceDoc/41303G.pdf

TXSTA – configuration registers for transmission of data through a serial connection
RCSTA – configuration register for reception of data through a serial connection
*not used in this project but important if you want to take it further and receive data

BAUDCON – Baud rate control register, allows you to invert data transmit
OSCCON – configuration of internal clock (important as it affects all values pertaining to baud rate)
SPBRG – This register is set to a number based on your clock speed and desired baud rate. Set to the nearest rounded value based on the equation

Once you have all of your settings configured, assuming they are all correct the actual transmission of you data is simple. As long as **TXIF** (your transmit status register) is set high, then you can simply set your TXREG (your transmit data register) to whatever byte you want to go to the serial port. Then you set TXIF = 0 to reset the process.

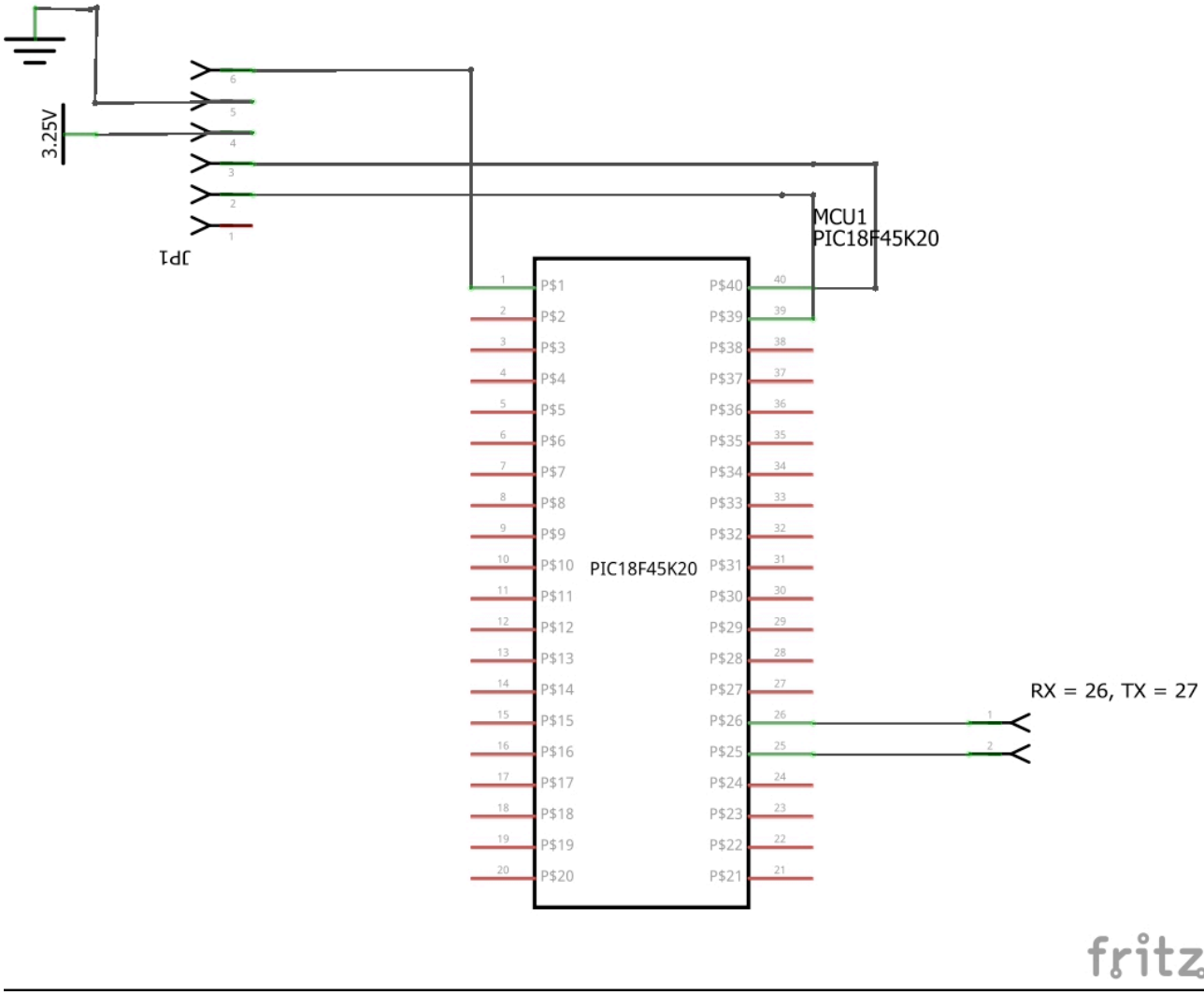
You'll want to note that this setup will only allow you to send data, you will not be able to receive data from your computer terminal. This is because the logic levels between your microcontroller and RS232 cables are not compatible. If you would like to be able to receive data you would need to add a level shifter between your microcontroller and the RS232 cable.

Issues

The biggest issue you may encounter is syncing up baud rates between your computer terminal and the microcontroller. If these do not match up exactly then you will get strange characters as opposed to the ones you were intending. It turns out that when you are setting the BAUDCON register you must invert Data Transmit so that it is active low. This allows the RS232 to USB cable to recognize correct voltage values and transmit your data correctly.

The only other minor issue that you might have as a beginner with this project is making sure you know what clock frequency you are operating on. The clock frequency is the basis of your baud rate calculation, and if you are using the wrong value, your project will simply never work!

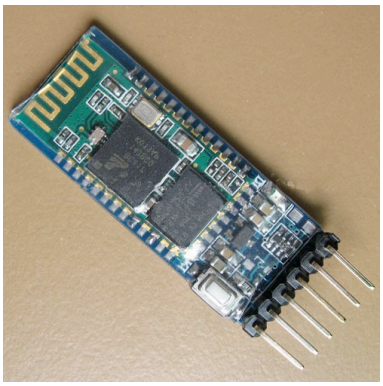
Schematic



Auxilary project: USART connection using bluetooth

If you'd like to take this project further consider adding some Bluetooth capability to your serial connection. Most of the code for the project stays the same the only difference will be that you no longer need to change to active low.

The JY-MCU Bluetooth module can be used. Note that you do not need the same level shift to active low like you do in the RS232 version of this project.



This Bluetooth module is pretty straightforward to work with, you only need three wires to start sending data to your computer.

VCC – the module is powered between 3.6v-6v

GND – common ground between your microcontroller and the power line of the USB module

RXD – This is the receive line for the module, which means you will connect the TX line from your microcontroller here.

Once you have powered the Bluetooth module it should appear as a Bluetooth device on your computer (make sure your computer has Bluetooth or you'll need to purchase a USB dongle for this to work!). Connect to the device and once it is connected it should work exactly as the original version did. It will show up as a COM port on whatever terminal program you are using.

References

ES310 lab 10 – Nick Ellis