RFM22B Example Project: EWS_RFM22B_Chat_Example

A. Purpose:

To design and build a simple wireless chat example project, demonstrating use of the RFM22B wireless transceiver module.

B. Background:

About the most basic operation one can do with a wireless module is transmit a byte from one point to another. This is the basis for this project. Two identical boards are required for use, each connected to a PC. A user types a message at one PC, which will be transmitted and read by a user at another. Both sides of the wireless link are capable of transmitting and receiving. This project is a bare-bones example of how to get two RFM22B's communicating quickly and easily.

C. Design:

Figure 1 describes the path of a message from one PC to the other:

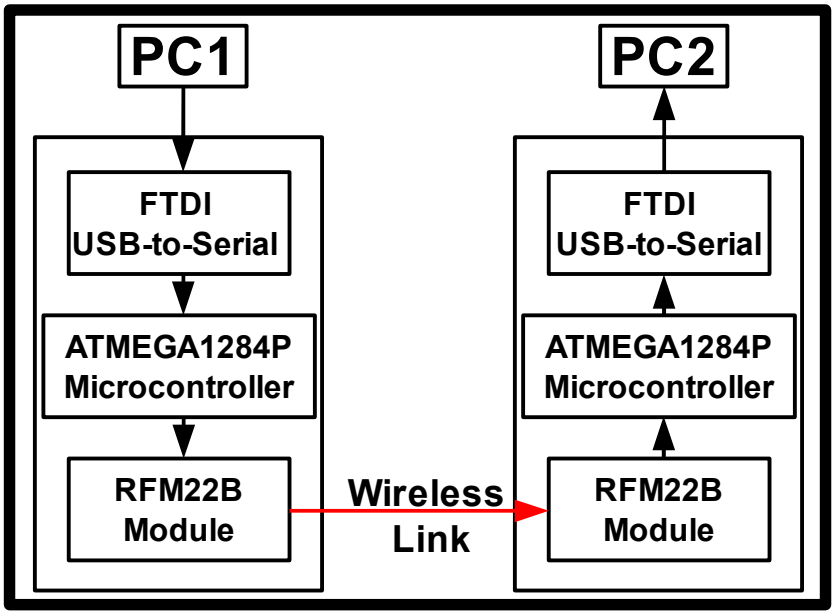

Figure 1: Communication Path Example - PC1 sends message To PC2

Both sides of the wireless link are identical, and are capable of transmitting and receiving. The system is normally in a listening state, waiting for incoming messages. When a user wishes to send a message to the other user, s/he types the message in a terminal session (such as Hyperterminal or Bray's Terminal). When the <ENTER> key is pressed, the message is transmitted wirelessly to the other user's station. After transmission is complete, the system goes back to a listening state.

D. Hardware:

This project consists of three major components: 1) PC, 2) EWS ATMEGA1284P Development Board (Atmel ATMEGA1284 MCU), 3) RFM22B Breakout Board.

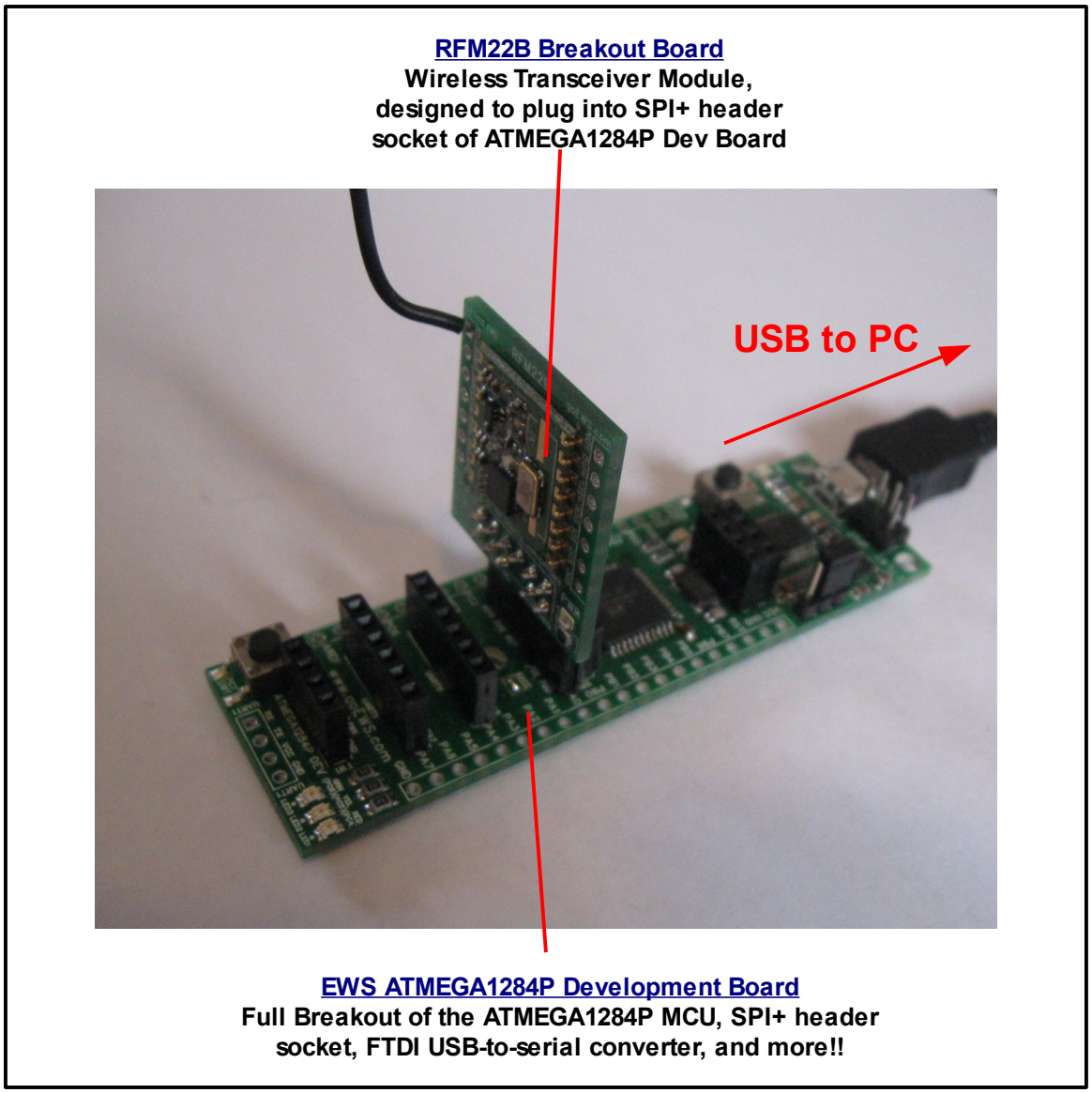

Figure 2: EWS_RFM22B_CHAT_Example – Breadboard Setup

E. Software/IDE:

Program development was in C, and undertaken using WINAVR. Programming of the AVR microcontroller is via bootloader programming (optiboot) on the ATMEGA1284P Development Board. However, programming can also be done with the same code in AVRStudio & Atmel Studio. (click here to download project source code).

The code was based on the RFM22B example provided on the mfr's website.

F. Testing/Results:

For testing the project, I opened two terminal sessions (Hyperterminal, Bray's Terminal, TeraTerm, etc) to interface and communicate with each RFM22B station (see Figure 3 below for example of communication). The FTDI chip on-board the EWS ATMEGA1284P Development Board made programming and communication simple and efficient.

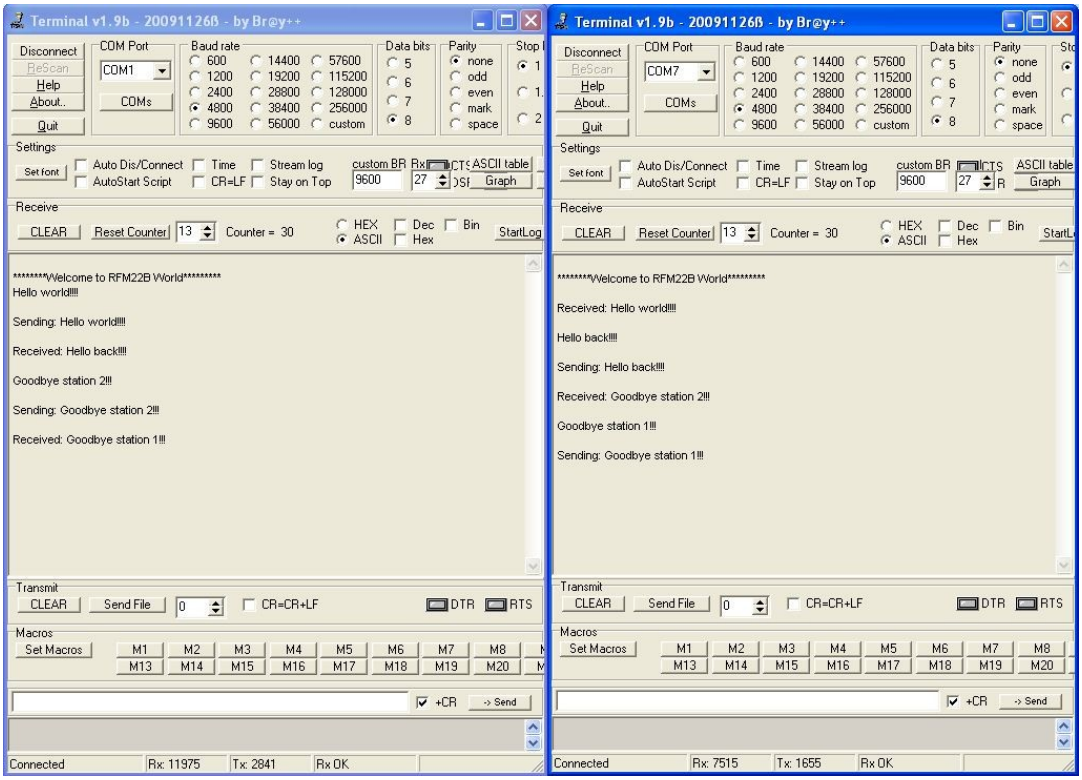

Figure 3 - RFM22B Terminal Sessions (Bray's Terminal)

The RFM22B proved to be easy to use (once you understand how the registers should be programmed for setup and mode change)and handles a lot of the processing involved in communication. Sometimes a packet is lost when the two sides are trying to transmit at the same time. A checksum was used in the transmitted packet, so if the checksum byte is invalid, the packet is ignored. With more overhead in code, this problem could be easily avoided (handshaking to check if packet was received properly – resend if there was a problem).

G. Conclusions:

This is a great wireless module. So many pluses – **inexpensive**, data rate, fairly easy-to-use, great communication range vs. power consumption. This module really opens up quite a realm of possibilities for adding wireless communication to projects.

The learning curve comes in with the module setup – there is an extensive register list to understand before communication can take place.

Please contact us via our website if you have any questions or comments about the design/development of this project.