EWS BlueGiga WT12/WT32 Example Project: Remote Temperature Logger

A. Purpose:

To demonstrate basic functionality of the BlueGiga WT12 and WT32 Bluetooth modules (this application/code example applies to either module – Figure 1); namely device discovery, pairing, establish/close connection, and data transmitting.

Figure 1: BlueGiga Bluetooth Modules (WT32 and WT12)
(image source: http://www.bluegiga.com/Bluetooth_Modules)

B. Background:

The BlueGiga bluetooth modules are the most versatile Bluetooth modules that we've found. They have an easy-to-use, intuitive firmware (iWrap) loaded with functionality. iWrap is well documented, and full of example shots of the firmware commands in use.

This project gives a starting point for using the basic functions in Bluetooth communication. To make it a more interesting application tutorial, we took advantage of the on-board temperature sensor (caveat: it is not terribly accurate, but seems to work fine if you need to know the temperature within 5 degrees).

This project uses two identical boards, each consisting of a Bluetooth module and microcontroller. One board (**REMOTE DEVICE**) takes periodic temperature readings (program is set for 30 minute intervals), and connects to the other board (**BASE DEVICE**) via Bluetooth, then transmits the temperature reading and closes the Bluetooth connection. To access the data, the base device is connected to a PC via an RS232 connection. When the PC is ready with a terminal session running, the user can just hit any key and the logged data will be displayed in an easy to read format. With this project, we monitored ambient temperature

fluctuations over a chosen time period (the project uses a buffer that stores temperature readings at 30 minute intervals).

C. Design:

Figure 2 gives an overview of the system:

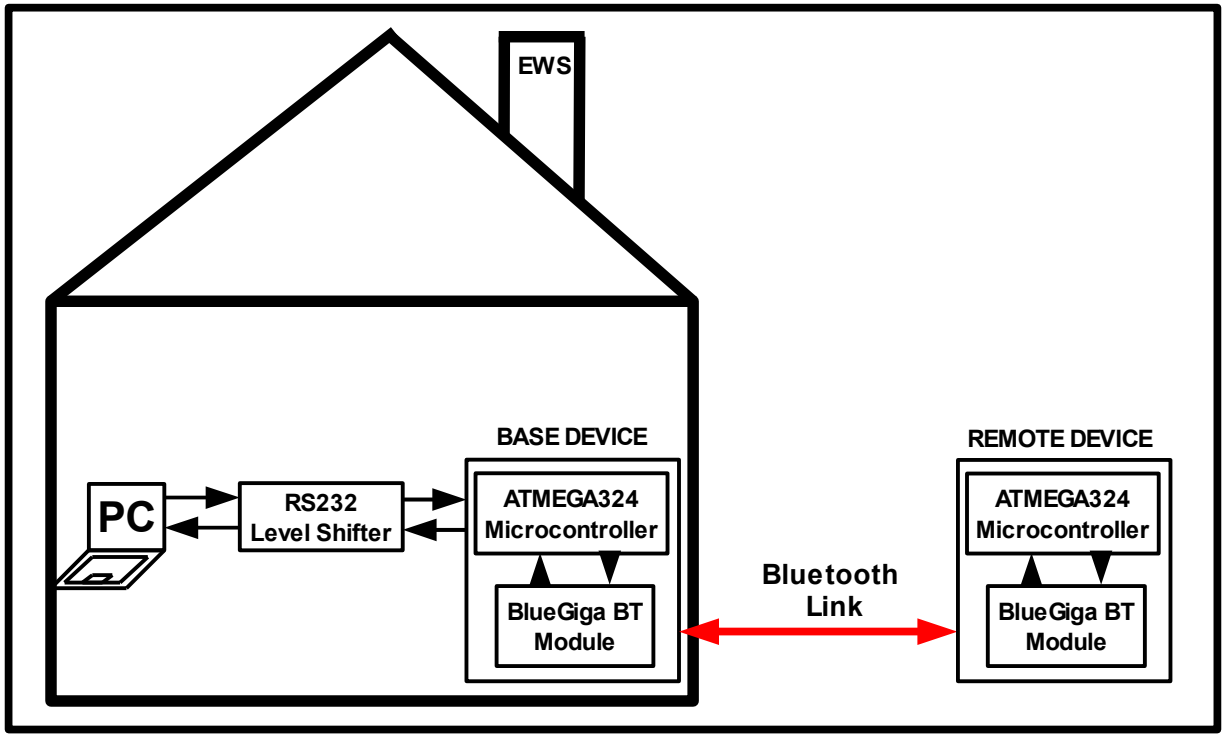

Figure 2:
EWS BlueGiga Example Application Overview –
Remote Temperature Monitoring System

In this system, two identical devices are used, but their individual roles are chosen by the user. Upon power-up each device is inherently a **BASE DEVICE**, however one will be changed to a **REMOTE DEVICE** when the user pushes the button connected to pin 7 on PORTC. This remote device will connect to the **BASE DEVICE** via Bluetooth every 30 minutes (user specified time interval) and send ambient temperature readings, then terminate the connection.

Communication between the microcontroller and the BlueGiga WTxx module takes place via serial communication (RS232). The microcontroller sends a command to the WTxx module, then monitors the incoming response from the module for a specific character string (newline character, “temp”, “pair”, etc, - depends on the command that was used), indicating that a valid response has been received. Then, the receive buffer is parsed/examined for the received response. The functionality in this program implements a timeout to allow the program to abort in the case that a valid response is not received.

When the **REMOTE DEVICE** is initiated by the user via the push-button, it runs a Bluetooth Device Discovery to detect the **BASE DEVICE**, and pairs with the first Bluetooth device that it finds, assuming that it is the correct one (if the button is pressed a second time, the **REMOTE**

DEVICE reverts back to a **BASE DEVICE**). If there are multiple Bluetooth devices in range and thus a problem with this functionality, the user should delete this Discovery/Pairing option and manually enter the Bluetooth address of the **BASE DEVICE** in the microcontroller code of this example. Note: if there are connecting problems with this example, make sure to delete all pairings from the WTxx modules and retry. Past pairings can give erroneous operation, since the program accesses the pairing records of the WTxx module when looking for a Bluetooth address to connect to.

When the **BASE DEVICE** receives a temperature reading from the **REMOTE DEVICE**, it stores it (up to 127 – 2 digit readings can be stored as the program is currently set up). The user can access these stored temperature readings by implementing the following protocol:

1. Connect the **BASE DEVICE** microcontroller to a PC via an RS232 Transceiver (level converter).

2. Open a terminal session with the **BASE DEVICE** microcontroller (Bray's Terminal, Hyperterminal, etc) (Baud rate = 115,200 bps)

3. On the PC, hit any key to display the stored temperature readings in an easy-to-read format (Figure 3). Note: the temperature storage buffer will be emptied by this action.

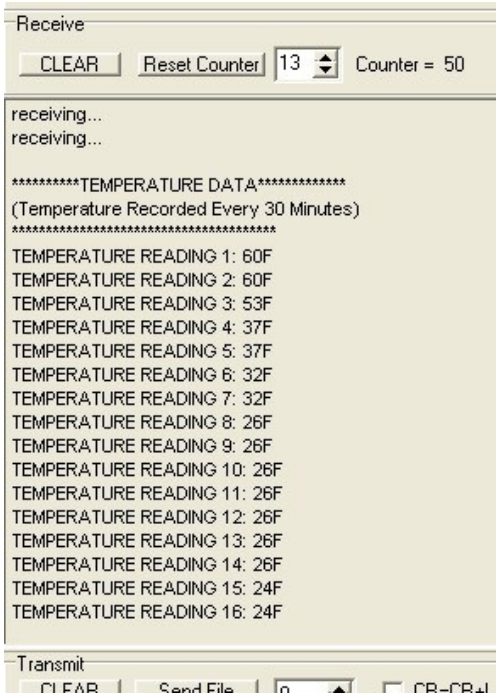

Figure 3:
Stored Temperature Data Displayed on PC

D. Hardware:

This project consists of four major components: 1) PC, 2) RS232 Transceiver (Level Converter), 3) Microcontroller, 4) BlueGiga WT12 or WT32 Bluetooth Module (BlueGiga module with iWrap firmware). The Atmel ATMEGA164/324/644 microcontroller was used in the implementation of this project.

E. Software/IDE:

Program development was undertaken using WINAVR (WinAVR-20100110). Programs were written using Programmer's Notepad (written in C programming language). Programming of the AVR microcontroller is via the 6-pin ISP port using the ATMEL MKII programmer.

F. Testing/Results:

The laptop that I was using has only one serial port, so throughout development and testing, I used a USB to serial converter (Figure 4) for connecting the second breadboard setup to the same machine through a USB port. Thus, I was able to have two terminal sessions open simultaneously through two different COM ports (Figure 5). Also, fewer and fewer machines are manufactured with a serial port, so the USB/serial converter here, or a similar product, would be handy for serial communication between the PC and microcontroller.

Figure 4: USB to Serial (RS232) Converter

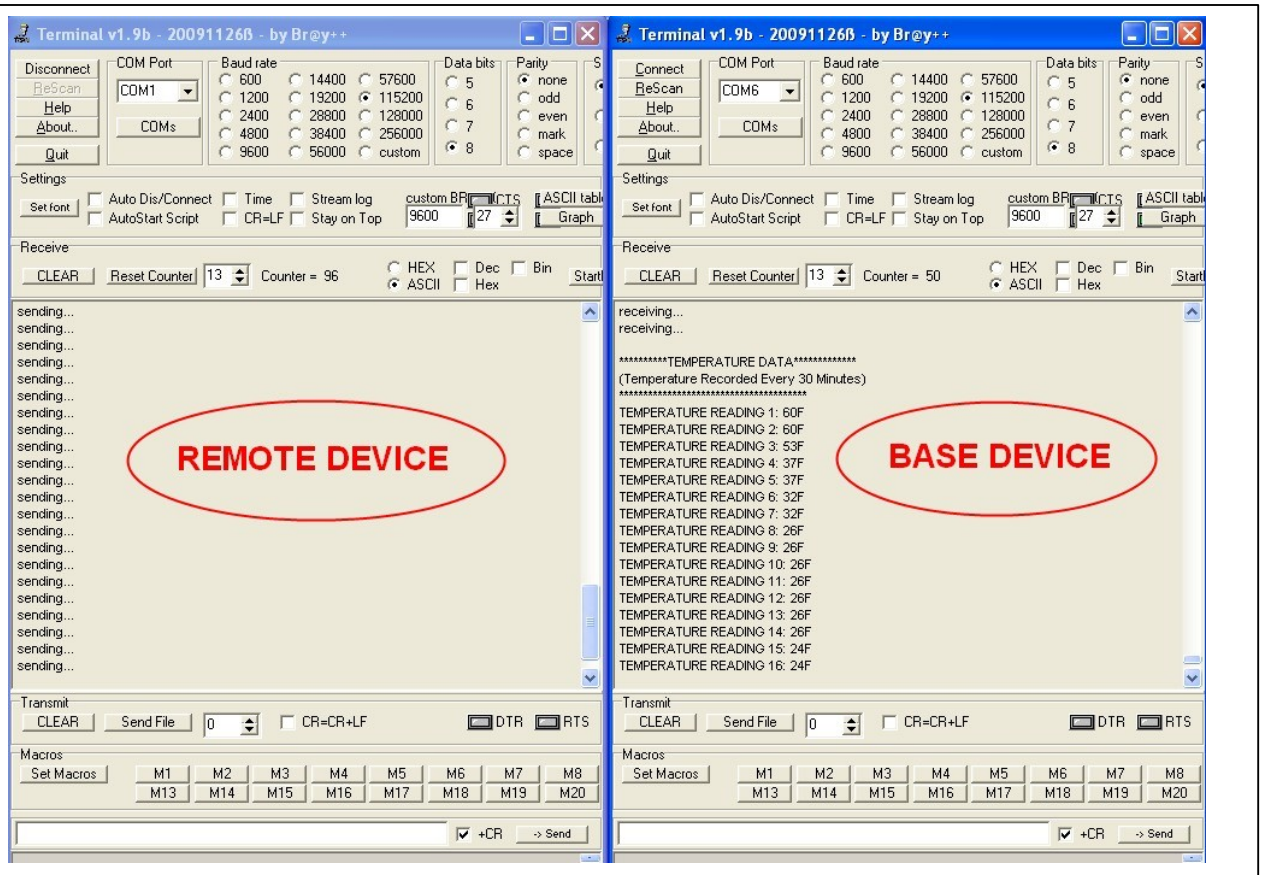

Figure 5: Outputs of Both Devices Displayed in Two Instances of Bray's Terminal

The project was a successful demonstration of simple Bluetooth communication between two modules, laying out the basics of use. The following set of Bluetooth functionality (followed by the corresponding iWrap command in parenthesis) was shown in this example:

1. Setting the device passkey ("set bt auth")
2. Nearby device discovery ("inquiry")
3. Pairing with another device ("pair")
4. Take temperature reading ("temp")
5. Connect with another Bluetooth device ("call")
6. Close Bluetooth Connection ("close")

This example should be used as a guide – a starting point for using the WT12 and WT32 modules from BlueGiga. The example code is not a finished product; it is purposely stripped of much error-checking/handling code to maintain brevity. The purpose was not an air-tight program, but a short, simple program to introduce these devices simply to a new user.

G. Conclusions:

These modules are top-notch. BlueGiga is a favorite worldwide for Bluetooth modules. Their iWrap firmware is powerful, easy-to-use, and widely accepted and well-known. These modules (BlueGiga WT12/BlueGiga WT32) pack a plethora of high-level functionality, but also lend themselves well to just a basic wireless RS232 link – they are perfect for either the novice or the advanced user.

We have also used this module for applications connecting to a cell phone (Nokia-Java ME/JSR-82 and HTC-Windows Mobile phones), and have been very satisfied with their successful and easy use.

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