EmbeddedBlue™ 500

User Manual

Part Number 0000033 – Revision E

Last revised on April 6, 2005 – Printed in the United States of America

A7 Engineering, Inc.
12860 C Danielson Court
Poway, CA 92064

Life Support Policy and Use in Safety-Critical Applications:

A7's products are not authorized for use in life-support or safety-critical applications. Use in such applications is done at the sole discretion of the customer and A7 will not warrant or authorize the use of its devices in such applications.

Table of Contents

Table of Figures

Table of Tables

Introduction

Congratulations on your purchase of the EmbeddedBlue 500 (eb500) serial Bluetooth module. The eb500 is an add-on component to the Parallax BS2, BS2e, BS2sx, BS2p, BS2pe, and Javelin Stamp microcontroller modules; enabling wireless communications with other Bluetooth devices including cellular phones, handheld computers, PCs, and other serial port adapters. Hobbyists, developers, and OEMs can take advantage of advanced wireless connectivity with this easy to use module.

The eb500 module provides a point to point connection much like a standard serial cable. Connections are made dynamically and can be established between two eb500 modules or an eb500 module and a standard Bluetooth v1.1 or v1.2 device. Devices can be dynamically discovered and connected in an ad-hoc manner.

Manual Conventions

Below is a list of typographical conventions used in this manual:

`Text in this font`

- `Is used to show data that is sent to the eb500.`
- `Inside a gray box is used to show data that is sent from the eb500.`

`Text in this font`

- `Is used to show source code`

In the command set section of this manual

- Required parameters and placeholders appear in standard lowercase type.
- Placeholders appear in *italics*. For example, if *address* shows up in a syntax line, the actual address of the device must be entered.
- Required parameter options are separated by a vertical bar |.
- Optional parameters are enclosed in brackets [ ].

Getting More Information

The Bluetooth website, www.bluetooth.com, contains the Bluetooth specification, profiles, and other documents relevant to Bluetooth.

General information regarding the eb500 module, EmbeddedBlue, and other Bluetooth products from A7 Engineering can be found on the A7 website at www.a7eng.com.

Parallax provides technical support through email support@parallax.com, an online group http://groups.yahoo.com/group/basicstamps, and by telephone. It is recommended that you use email as the first line of support because common questions can be answered quickly and in greater detail in this manner. Questions that involve Parallax products and their use with the eb500 module should be directed to Parallax technical support.

A7 Engineering provides technical support for EmbeddedBlue products through an online discussion forum at www.a7eng.com/support/forum/forum.htm. When you visit the forum you can search through previously asked questions for information or post new ones. The forum is monitored by A7 Engineering employees so that your question will be answered in a thorough and timely manner. If your question involves sensitive information you can request private support by sending an email to support@a7eng.com.

A7 Engineering also provides professional design services on a contract basis to anyone requiring assistance with their design and/or development of Bluetooth products. For further information visit the A7 Engineering website www.a7eng.com/services/services.htm.

Bluetooth Overview

What is Bluetooth?

To put it simply, Bluetooth is a technology standard for electronic devices to communicate with each other using short-range radio. It is often referred to as a “cable replacement” technology, because it is commonly used to connect things, such as cameras, headsets, and mobile phones that have traditionally been connected by wires. Bluetooth is much more than simply a way to cut the cord between today’s existing electronic devices. It is an enabling technology that will take these devices to new levels of productivity and functionality and enable a whole new class of devices designed with communications and connectivity in mind.

The Bluetooth Special Interest Group (SIG) defines Bluetooth a bit more broadly as the "worldwide specification for small-form-factor, low-cost radio solutions that provide links between mobile computers, mobile phones, other portable devices, and connectivity to the Internet." In defining Bluetooth, the SIG has taken a very different approach than the IEEE 802.11 Committees did. Rather than build Bluetooth as an adjunct to TCP/IP, it was defined as a standalone protocol stack that includes all layers required by an application. This means that it encompasses not only wireless communications but also service advertisement, addressing, routing, and a number of application-level interfaces referred to as profiles.

Bluetooth is based on a frequency hopping spread spectrum (FHSS) modulation technique. The term spread spectrum describes a number of methods for spreading a radio signal over multiple frequencies, either simultaneously (direct sequence) or in series (frequency hopping.) Wi-Fi devices are based on direct sequence spread spectrum transmission which uses multiple channels simultaneously. While this technique increases the speed of transmission (for example in Wi-Fi from 1.5MHz to 11MHz), it is more susceptible to interference from other radio sources as well as being a greater source of interference to the surrounding area.

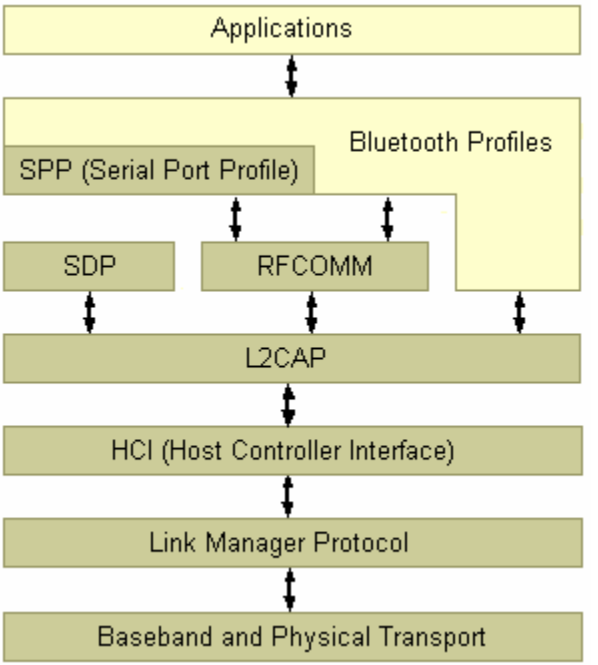
In contrast, Bluetooth utilizes the frequency hopping method of spread spectrum which uses multiple radio channels to reduce interference and increase security. The signal is rapidly switched from channel to channel many times per second in a pseudo-random pattern that is known by both the sender and receiver(s). This provides robust recovery of packet errors caused by interference from another radio source at a particular frequency. Also, data is generally more secure because it is not possible to receive more than a fraction of the data

unless the hopping pattern is known. Bluetooth utilizes frequency hopping in the 2.4GHz radio band and hops at a relatively fast pace with a raw data rate of about 1 Mbps. This translates to about 700 kbps of actual useful data transfer. The eb500 module supports a maximum sustained bidirectional data rate of 230.4kbps.

What is a Profile?

Bluetooth devices can support interoperability with one or more types of devices. In order for two Bluetooth devices to communicate with each other, they must share at least one common profile. If I want a Pocket PC to communicate with my EmbeddedBlue radio I need to make sure that they both support the same profile. EmbeddedBlue devices support the Serial Port Profile (SPP) which is one of the earliest and most widely supported profiles.

The main elements of the Bluetooth stack are shown in the figure to the right. As with a typical diagram of the TCP/IP stack, there are a number of details that are hidden by the apparent simplicity of the stack. Specifically, there are a number of profiles that sit roughly on top of the L2CAP layer that provide much of the power (and also the complexity) of the Bluetooth protocols.



These profiles are the primary entry into the stack for an application. Essentially, they define the set of services that are available to that application. Currently there are more than 25 different profiles defined or in the process of being defined by the Bluetooth SIG. With so much variety, acquiring an in-depth understanding of Bluetooth is not a trivial task. However, the abstraction by a single profile can provide an application the use of the profile without such detailed knowledge.

There are a number of profiles that are exposed in very familiar forms. The eb500 module, for instance, implements the SPP profile which enables it to appear like a traditional serial port. This virtually eliminates the need for the user to have specific Bluetooth knowledge and allows the radios to be integrated into applications very quickly.

Bluetooth and Wi-Fi

Bluetooth and Wi-Fi are often compared to each other because they are both capable of providing networking on the 2.4GHz consumer frequency band. Many of the differences between these two technologies can be traced to the fact that networking was not the primary design goal for Bluetooth as it was for Wi-Fi. With a greater transmission range (about 100 meters indoors) and larger bandwidth (about 11Mbps), Wi-Fi is typically the better choice for wireless LANs and Internet connectivity.

Bluetooth on the other hand was designed for driverless, cordless connectivity between devices. Because Bluetooth transmitters are smaller in size, have lower power demands, a more limited range (10 - 100 meters) and narrow bandwidth (1Mbps), they are better suited for use in embedded and mobile devices that exchange smaller amounts of information while conserving power and space.

While their functionality does not compete directly, 802.11b and Bluetooth do compete for the airwaves. Since they both operate on the 2.4GHz band of the ISM radio spectrum, these two wireless technologies may interfere with each other. Bluetooth devices minimize interference by employing a frequency-hopping spread spectrum scheme that changes the frequency used about 1600 times per second. Unfortunately, since Wi-Fi uses a direct sequence spread spectrum method, this also means that Bluetooth transmissions will collide with those of any nearby 802.11b devices and slow Wi-Fi data transmission rates. The Bluetooth SIG and its member companies have put a lot of effort into coexistence solutions for these two standards and are very committed to ensuring that these devices work well together.

While 802.11b was designed solely for data communications, Bluetooth takes things quite a bit further. A key component of the Bluetooth standard is its notification and service discovery mechanism. This allows Bluetooth devices to identify themselves and describe their capabilities to other Bluetooth devices in the area. For instance, the Dial-Up Networking profile defines how discoverability can be used to locate and connect to other devices such as a cellular phone that supports the same profile. The profile then describes how to dial the phone, connect to either analog or data services, and control the connection seamlessly. This combination of dynamic discovery of services and built in definitions of the services goes well beyond anything offered by the 802.11b protocol.

Security

Bluetooth security is defined by three main elements: availability, access, and confidentiality. It is important to distinguish between these elements because Bluetooth security is also highly configurable so that it can meet the needs of devices in many different scenarios. An understanding of the basics will provide the knowledge that you need to choose a security strategy for your device.

The first important element of Bluetooth security is availability. If a device cannot be seen or connected with, it is obviously quite secure. Bluetooth defines both of these features as part

of the security model and they are exposed by the EmbeddedBlue device through the *set visible* and *set connectable* commands. This is a very coarse level of control, but it is also quite effective and can be used in combination with other security features.

The second and most complex element of Bluetooth security is access control. This type of security is only relevant when the module is *connectable* and is designed to provide protection in this case. The general idea is that remote devices must become trusted before they will be allowed to connect and communicate with the EmbeddedBlue module. In order to become trusted, a remote device must present a passkey that matches the stored local passkey. This only needs to be done once, as both devices will remember their trusted status and allow future connections with that specific device without exchanging passkeys again.

The EmbeddedBlue module uses the *set security* command to configure access control. There are three possible settings for security, *off*, *open*, and *closed*. When security is turned *off*, connection attempts will be allowed from all remote devices. When security is set to *open,* connections are only allowed from trusted devices, but new devices can become trusted by presenting the correct passkey. Forming a trusted relationship is carried out automatically in this mode the first time that a remote device connects with the EmbeddedBlue module. When security is set to *closed*, only connections from trusted devices will be allowed and no new devices may become trusted. Closed security is the most restrictive setting and therefore the most secure.

The last element of Bluetooth security is confidentiality. Once a link with a trusted device has been established, it may be important to know that the data being transmitted cannot be intercepted by a third party. All transmitted data can be encrypted by configuring the *encrypt* setting to *on*. This only has an effect when security is set to either *open* or *closed*. The EmbeddedBlue module supports 56-bit encryption by default, but 128-bit encryption is available. Due to export restrictions to certain countries, firmware supporting 128-bit encryption is only available with proper approval from A7 Engineering.

The Basics

Most of the complexity of working with Bluetooth has been encapsulated in the EmbeddedBlue module in order to make it easier to use. The specific application profile that is supported is SPP, or the Serial Port Profile. This is the most popular and convenient protocol for many embedded applications of Bluetooth since it emulates a simple serial port link between devices. Once the connection is set up, it is a simple matter to communicate between the endpoints of that connection using familiar and well-supported programming constructs as will be shown by the numerous programming examples throughout this manual.

PC Prototyping

One of the best ways to learn about anything new is through hands-on experimentation. To make this easy with the eb500, the following sample will enable you to communicate directly from your PC to the radio module.

To perform this exercise, as documented, you will need one eb600 RS232 interface board, one eb500 module, and a PC with an available serial port.

Step 1: Insert the eb500 Module into the eb600 RS232 interface board

In this step we will attach an eb500 module to the eb600 RS232 interface board and apply power to the device.

1. Insert an **eb500 module** into the **eb600 PC Adapter** header; assuring that Pin 1 of the eb500 module is inserted into Pin 1 of the header on the eb600 PC Adapter.
2. Connect the **eb600 PC Adapter** to a serial port on the PC using the **provided straight through serial cable**.

   The PC serial port must be available for HyperTerminal use.
3. Apply **power** to the **eb600 PC Adapter**.

Step 2: HyperTerminal Setup

In this step we will setup the Windows HyperTerminal application to establish a connection with the eb500 module.

1. Open **HyperTerminal**.

   This will display the Connection Description dialog.

2. In the **Name** box, type the name of your connection. (e.g. eb500).
3. Click **OK**.

   This will display the Connect To dialog.

4. In the **Connect using** dropdown, select the **serial port** to which you have connected the eb600 interface board.
5. Click **OK**.

   This will display the properties dialog.

6. In the **Bits per second** dropdown, select **9600**.
7. In the **Data bits** dropdown, select **8**.
8. In the **Parity** dropdown, select **None**.
9. In the **Stop bits** dropdown, select **1**.
10. In the **Flow control** dropdown, select **None**.
11. Click **OK**.

    This will establish a connection to the serial port assuming that no other devices are using the serial port. If another device is using the serial port, disconnect the other device using the associated application or choose a different serial port to connect the eb500.

12. On the **Call** menu, click **Disconnect**.

    This will disconnect the connection just established so that we can modify the connection properties as follows.

13. On the **File** menu, click **Properties**.

    This will display the properties dialog.

14. On the **Settings** tab, click **ASCII Setup**.

    This will display the ASCII Setup dialog.

15. Check the **Send line ends with line feeds** checkbox.
16. Check the **Echo typed characters locally** checkbox.
17. Check the **Append line feeds to incoming line ends** checkbox.
18. Check the **Wrap lines that exceed terminal width** checkbox.

19. Click **OK**.

    This will return to the properties dialog.

20. Click **OK**.

21. On the **Call** menu, click **Call**.

    This will establish a connection with the serial port.

22. Press the **"Enter"** key to send a carriage return to the eb500 module.

    You should see a caret appear in HyperTerminal; this is the prompt for the eb500.

Step 3: Execute a few eb500 commands

In this step we use the connection that has been established to execute a few simple eb500 commands.

1. Using HyperTerminal, get the **version** of the **eb500** module by using the Version command.

   Example:

```
>ver all
ACK
Firmware Version: 2.0
Firmware Build: 247
Model Number: eb500
Serial Number: 238
Manufacturer: A7 Engineering
>
```

2. Using HyperTerminal, get the **address** of the **eb500** module by using the Get command.

   Example:

```
>get address
ACK
00:0C:84:00:05:29
>
```

Experiment with other commands simply by typing them in to the HyperTerminal window. The HLP command is a great place to start.

Command Mode

The eb500 supports two main operating modes: command mode and data mode. Upon power up, the eb500 enters command mode and is ready to accept serial commands. The factory default communication parameters are 9600 Baud, 8 Data Bits, 1 Stop Bit, No Parity, and No Flow Control. The eb500 supports commands to modify the baud rate and flow control settings.

In this mode there are a number of commands that can be sent to change the baud rate, locate other devices that are in range, check the firmware version, etc. All commands are sent using visible ASCII characters (123 is 3 bytes “123”). Upon the successful transmission of a command, the ACK string will be returned. If there is a problem in the syntax of the transmission then a NAK string is returned. After either the ACK or NAK, a carriage-return <CR> character is returned. When a prompt (<CR> followed by a ‘>’) is returned, it means that the eb500 radio is in the idle state and is waiting for another command. White space is used to separate arguments of the command and a carriage-return <CR> (ASCII 13) is used to mark the end of the command.

Data Mode

Once the eb500 radio is connected to another Bluetooth device, the eb500 automatically switches into data mode. All data transmitted while in this mode will be sent to the remote device and, therefore, NO further commands can be sent until the eb500 radio is disconnected or switched back to command mode by use of the mode control I/O line or the Switch to Command Mode sequence.

The connection status line of the eb500 module can be monitored to determine if there is an active connection. Additionally, whenever a connection is present, the Connection Status LED on the eb500 module will be on.

I/O Lines

The eb500 module features a 20 pin header for connecting to the Parallax AppMod header. A full device pinout is available in the Technical Specifications section of this manual. There are several pins that are important when performing the exercises in the Establishing a Connection and Communications sections of this manual.

Pin 3 of the eb500 module, which aligns with the pin designated “P0” of the AppMod header, is the UART data output pin.

Pin 4 of the eb500 module, which aligns with the pin designated “P1” of the AppMod header, is the UART data input pin.

Pin 8 of the eb500 module, which aligns with the pin designated “P5” of the AppMod header, is the Connection Status pin. A BASIC Stamp application can interrogate this pin to determine the connection status of the eb500 radio.

Pin 9 of the eb500 module, which aligns with the pin designated “P6” of the AppMod header, is the Mode Control pin. A BASIC Stamp application can drive this pin high to enter Data Mode or low to enter Command Mode.

Resetting the eb500 to the Factory Default Settings

There are two different mechanisms to reset the eb500 module to the factory default settings. Either by shorting the STATUS and MODE pins (pin 8 and pin 9) and then applying power to the eb500 module, or by issuing the reset command to the eb500; see the command set reference at the back of this manual for the syntax of the reset command.

Switching between Data Mode and Command Mode

When a Connection command is issued, the eb500 attempts to establish a connection to the device with the address specified in the command. Once a connection is established, the eb500 switches into data mode. At this point all data sent to the eb500 is transmitted to the remote Bluetooth device over the wireless link. It is possible to switch from data mode to command mode, issue commands, and then return to data mode, while maintaining a connection. The eb500 allows you to switch between data mode and command mode by issuing the Switch to Command Mode and Return to Data Mode commands or by driving the MODE control I/O line (P6) of the eb500 module.

The following BASIC Stamp application uses the Switch to Command Mode and Return to Data Mode serial commands to switch between data mode and command mode. This application is available in electronic form on the accompanying CD in the Samples folder in the file CmdModeSoft.bs2.

```
'{$STAMP BS2}
szData VAR BYTE(20)
'Wait for the eb500 radio to be ready
PAUSE 1000

'Connect to the remote device
SEROUT 1,84,["con 00:0C:84:00:05:29",CR]
SERIN 0,84,[WAIT("ACK",CR)]

'Wait for the connection to be established
WaitForConnection:
```

```
        IF in5 = 0 THEN WaitForConnection
DEBUG "Connection established",CR

SEROUT 1,84,["This string is sent in data mode",CR]

'Switch to Command Mode
PAUSE 2000
SEROUT 1,84,["+++"]
SERIN 0,84,[WAIT(CR,">")]

DEBUG "In Command Mode",CR

'Get local eb500 Bluetooth Address
SEROUT 1,84,["get address",CR]
SERIN 0,84,[WAIT("ACK",CR)]

'Read the local address from the get command
SERIN 0,84,[STR szData\17]
SERIN 0,84,[WAIT(CR,">")]
szData(17) = 0
DEBUG “Local eb500 address: “,STR szData\17,CR

'Return to Data Mode
SEROUT 1,84,["ret",CR]
SERIN 0,84, [WAIT(CR,">")]

‘Send data through eb500
SEROUT 1,84,["My eb500 address is ",STR szData,CR]

'Switch to Command Mode
PAUSE 2000
SEROUT 1,84,["+++"]
SERIN 0,84,[WAIT(CR,">")]

DEBUG "In Command Mode",CR
```

```
'Disconnect from remote device
SEROUT 1,84,["dis",CR]
SERIN 0,84,[WAIT(CR,">")]

DEBUG "Disconnected",CR
```

The following BASIC Stamp application uses the mode control I/O line of the eb500 module to switch between data mode and command mode. Switching between data mode and command mode via the mode control I/O line is preferred, as it is faster than the serial method. This application is available in electronic form on the accompanying CD in the Samples folder in the file CmdModeHard.bs2.

```
'{$STAMP BS2}
szData VAR BYTE(20)
'Wait for the eb500 radio to be ready
PAUSE 1000

'Connect to the remote device
SEROUT 1,84,["con 00:0C:84:00:05:29",CR]
SERIN 0,84,[WAIT("ACK",CR)]

‘Wait for the connection to be established and switch into data mode.
‘When switching into data mode, a 300ms timeout is required to give the
‘module enough time to make the change.
WaitForConnection:
       IF in5 = 0 THEN WaitForConnection
HIGH 6
PAUSE 300

DEBUG "Connection established",CR

SEROUT 1,84,["This string is sent in data mode",CR]

'Switch to Command Mode
LOW 6
SERIN 0,84,[WAIT(CR,">")]
```

```
DEBUG "In Command Mode",CR

'Get local eb500 Bluetooth Address
SEROUT 1,84,["get address",CR]
SERIN 0,84,[WAIT("ACK",CR)]

'Read the local address from the get command
SERIN 0,84,[STR szData\17]
SERIN 0,84,[WAIT(CR,">")]
szData(17) = 0
DEBUG “Local eb500 address: “,STR szData\17,CR

'Return to Data Mode
HIGH 6
PAUSE 300

SEROUT 1,84,["My eb500 address is ",STR szData,CR]

'Switch to Command Mode
LOW 6
SERIN 0,84,[WAIT(CR,">")]

DEBUG "In Command Mode",CR

'Disconnect from remote device
SEROUT 1,84,["dis",CR]
SERIN 0,84,[WAIT(CR,">")]

DEBUG "Disconnected",CR
```

Hardware Connections

The eb500 module is designed to interface with a 5V CMOS signal environment. It supports a power supply of 5 – 12V and can be connected directly to boards supporting the Parallax AppMod header. When inserting the eb500 module into any of the supported Parallax boards, it is important that Pin 1 of the eb500 module, marked with a white dot and a square (Figure 1), is inserted into the VSS pin of the AppMod header on the Parallax boards. A full device pin out is available in the Technical Specifications section of this manual.

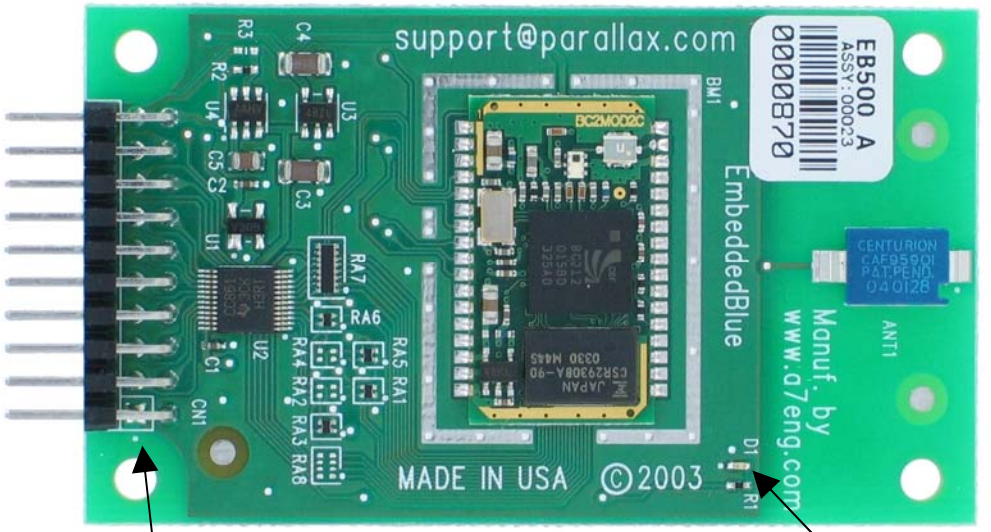


Figure 1: eb500 Module

Pin 1 Marked by White Dot and Square.

Connection Status LED

Basic Stamp Activity Board

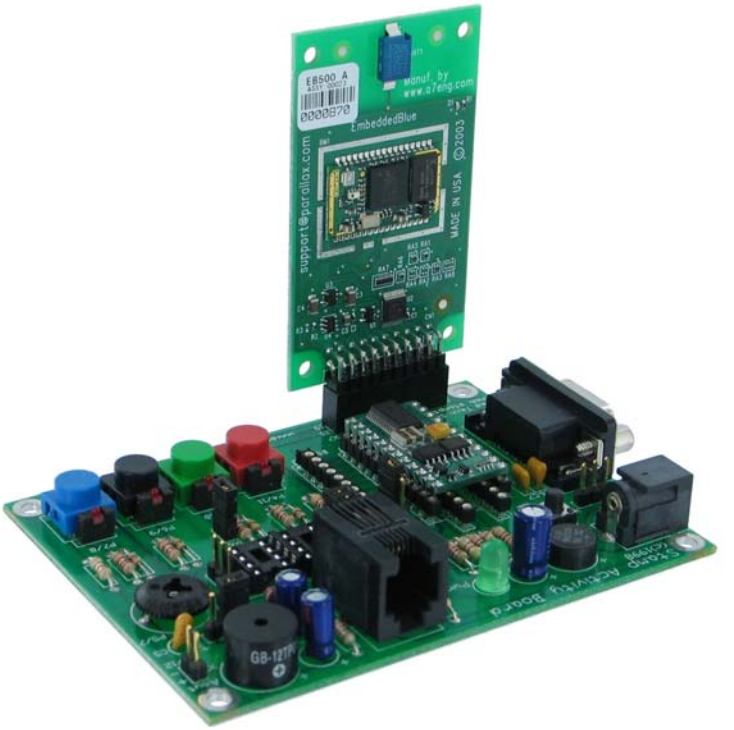


Figure 2: Basic Stamp Activity Board

> ⚠ **Prior to installing the eb500 module into the AppMod header, please ensure that the Stamp IO line P5 is not configured as an output. Failure to do so may result in damage to the eb500 module.**

The Basic Stamp Activity Board contains an AppMod header and supports a direct connection with the eb500 module. On the Basic Stamp Activity Board, the AppMod header is labeled X7. When inserting the eb500 module into the Basic Stamp Activity Board AppMod header, assure that you insert Pin 1 of the eb500 module, marked with a white dot and a square, into the VSS pin (Pin 1) of the AppMod header as pictured in Figure 2.

Board Of Education Board

Figure 3: Board of Education Board

> ⚠ **Prior to installing the eb500 module into the AppMod header, please ensure that the Stamp IO line P5 is not configured as an output. Failure to do so may result in damage to the eb500 module.**

The Board Of Education (BOE) contains an AppMod header and supports a direct connection with the eb500 module. On the Board of Education, the AppMod header is labeled X1. When inserting the eb500 module into the Board of Education AppMod header, assure that you insert Pin 1 of the eb500 module, marked with a white dot and a square, into the VSS pin of the AppMod header as pictured in Figure 3.

BS2p Demo Board

Figure 4: BS2p Demo Board

> ⚠ **Prior to installing the eb500 module into the AppMod header, please ensure that the Stamp IO line P5 is not configured as an output. Failure to do so may result in damage to the eb500 module.**

The BS2p Demo Board contains an AppMod header and supports a direct connection with the eb500 module. On the BS2p Demo Board, the AppMod header is labeled X7. When inserting the eb500 module into the BS2p Demo Board AppMod header, assure that you insert Pin 1 of the eb500 module, marked with a white dot and a square, into the VSS pin of the AppMod header as pictured in Figure 4.

BS2p24/40 Demo Board

Figure 5: BS2p24/40 Demo Board

> ⚠ **Prior to installing the eb500 module into the AppMod header, please ensure that the Stamp IO line P5 is not configured as an output. Failure to do so may result in damage to the eb500 module.**

The BS2p24/40 Demo Board contains an AppMod header and supports a direct connection with the eb500 module. On the BS2p24/40 Demo Board, the AppMod header is labeled X1. When inserting the eb500 module into the BS2p24/40 Demo Board AppMod header, assure that you insert Pin 1 of the eb500 module, marked with a white dot and a square, into the VSS pin of the AppMod header as pictured in Figure 5.

Javelin Stamp Demo Board

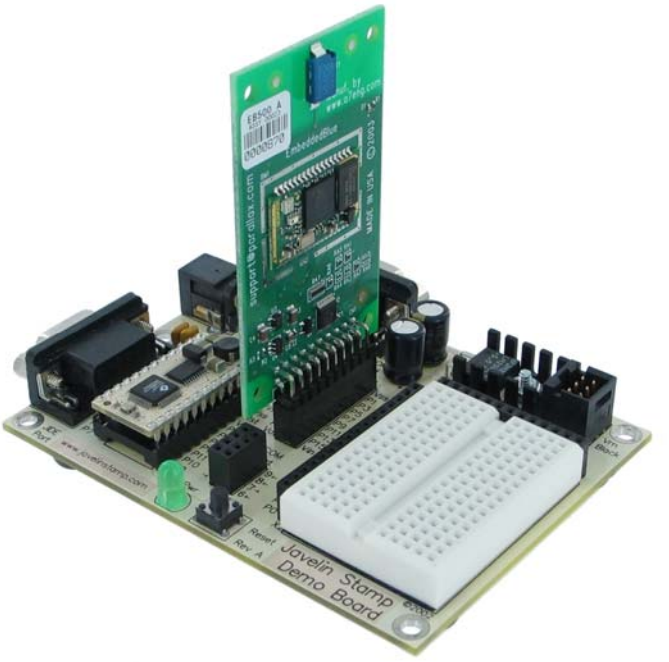


Figure 6: Javelin Stamp Demo Board

Prior to installing the eb500 module into the AppMod header, please ensure that the Stamp IO line P5 is not configured as an output. Failure to do so may result in damage to the eb500 module.

The Javelin Stamp Demo Board contains an AppMod header and supports a direct connection with the eb500 module. On the Javelin Stamp Demo Board, the AppMod header is labeled X1. When inserting the eb500 module into the Javelin Stamp Demo Board AppMod header, assure that you insert Pin 1 of the eb500 module, marked with a white dot and a square, into the VSS pin of the AppMod header as pictured in Figure 6.

SumoBoard

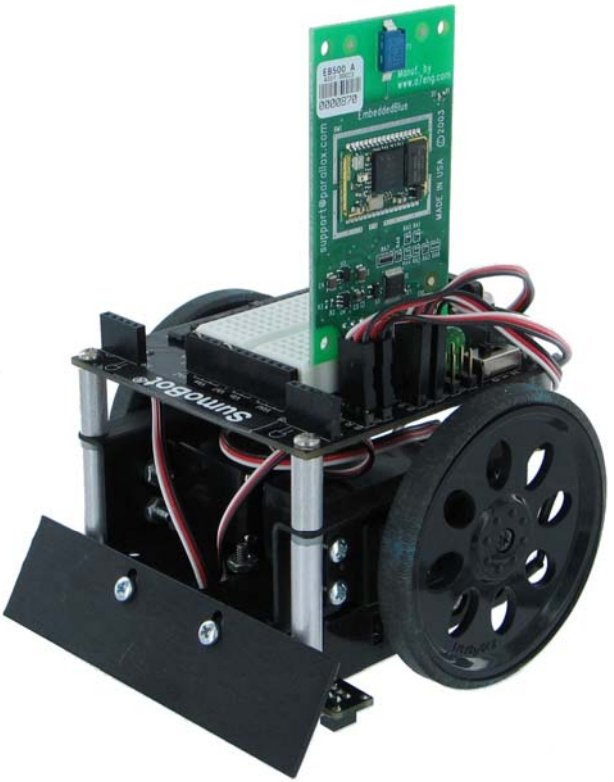


Figure 7: SumoBoard

> ⚠ **Prior to installing the eb500 module into the AppMod header, please ensure that the Stamp IO line P5 is not configured as an output. Failure to do so may result in damage to the eb500 module.**

The SumoBoard contains an AppMod header and supports a direct connection with the eb500 module. On the SumoBoard, the AppMod header is labeled X10. When inserting the eb500 module into the SumoBoard AppMod header, assure that you insert Pin 1 of the eb500 module, marked with a white dot and a square, into the VSS pin of the AppMod header as pictured in Figure 7.

Super Carrier Board

Figure 8: Super Carrier Board

> ⚠ **Prior to installing the eb500 module into the AppMod header, please ensure that the Stamp IO line P5 is not configured as an output. Failure to do so may result in damage to the eb500 module.**

The Super Carrier Board contains an AppMod header and supports a direct connection with the eb500 module. On the Super Carrier Board, the AppMod header is labeled X1. When inserting the eb500 module into the Super Carrier Board AppMod header, assure that you insert Pin 1 of the eb500 module, marked with a white dot and a square, into the VSS pin of the AppMod header as pictured in Figure 8.

Establishing a Connection

This section contains a number of exercises that demonstrate methods of establishing Bluetooth wireless connections with the eb500. The scenarios described are not meant to form an exhaustive list, but rather illustrate a number of more common and useful configurations. All source code shown in these exercises is available in electronic form on the accompanying CD, in the Samples folder, using the filename used in this manual. Additional samples will be made available on the A7 Engineering website at http://www.a7eng.com.

Connecting two eb500 Modules

In this exercise we will step through the process of establishing a connection between an eb500 inserted into a Board of Education and an eb500 inserted into a SumoBoard.

To perform this exercise, as documented, you will need a Board of Education board, a SumoBoard, and two eb500 modules. If you are using any of the other supported Parallax boards, you may need to make adjustments to this exercise.

Step 1: Insert the eb500 Modules into the BOE and SumoBoard Boards

In this step we will insert the eb500 modules into the Board of Education (BOE) and SumoBoard boards.

1. Insert an **eb500** module into the **AppMod header** of the Board of Education board; assuring that Pin 1 of the eb500 module is inserted into the VSS pin of the AppMod header.
2. Insert an **eb500** module into the **AppMod header** of the SumoBoard board; assuring that Pin 1 of the eb500 module is inserted into the VSS pin of the AppMod header.

Step 2: Write a BASIC Stamp Application to Get the eb500 Address

In this step we will write a BASIC Stamp application to interrogate an eb500 for its unique Bluetooth address.

1. Open the **Basic Stamp Editor**.

2. Enter the following **program code** into the editor. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file GetAddress.bs2.

```
'{$STAMP BS2}
szData VAR BYTE(20)
'Wait for the eb500 radio to be ready
PAUSE 1000

'Get the eb500 Bluetooth Address
SEROUT 1,84,["get address",CR]
SERIN 0,84,[WAIT("ACK",CR)]

'Read the local address from the get command
SERIN 0,84,[STR szData\17]
SERIN 0,84,[WAIT(CR,">")]
szData(17) = 0
DEBUG "Local eb500 address: ",STR szData\17,CR
```

   The BASIC Stamp application issues an eb500 Get Address command and then reads and displays the response in the debug window. The response is the Bluetooth address of the local eb500 module.

3. On the **File** menu, click **Save As**.
4. In the **File name** box, enter a file name to which to save the program just created. For example, GetAddress.bs2.
5. Click **Save**.

Step 3: Get the Address of the eb500 on the Board of Education Board

In this step we will get the Bluetooth address of the eb500 module on the Board of Education board. We will then use this address in the next step.

1. Connect the **Board of Education board serial port** to the **PC**.
2. Apply **power** to the Board of Education board.
3. On the **Run** menu, click **Run**.

   The Bluetooth address for the eb500 on the Board of Education board is shown in the debug window (Figure 9).

4. On the **Debug Terminal #1** dialog click **Close**.
5. Disconnect the **power** from the **Board of Education board**.
6. Disconnect the **Board of Education board serial port** from the **PC**.

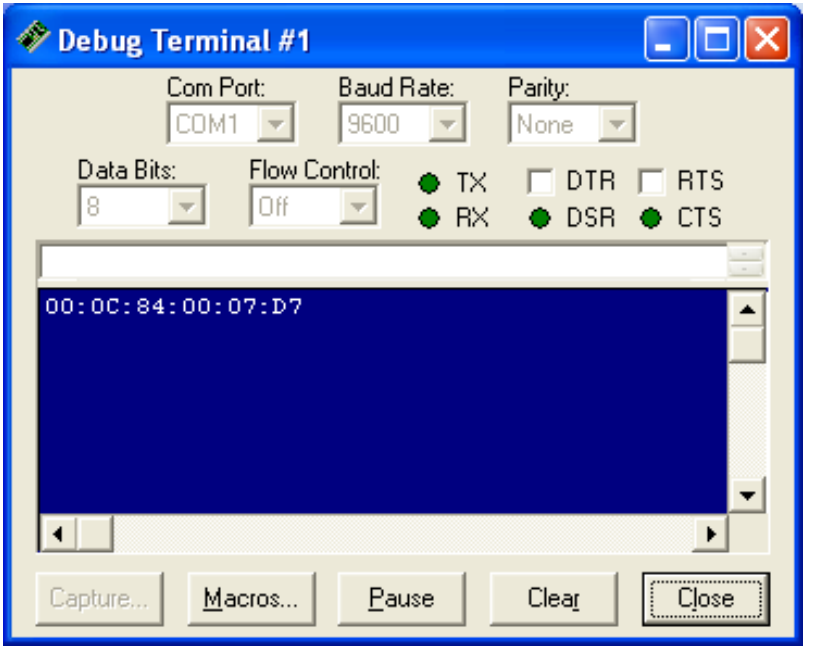


Figure 9: eb500 Bluetooth Address Output

Step 4: Connect the eb500 on the SumoBoard to the eb500 on the BOE

In this step we will develop and run a BASIC Stamp application on the SumoBoard to establish a connection with the Board of Education.

1. Using the BASIC Stamp Editor; on the **File** menu, click **New**.

   This will create a new project window within the BASIC Stamp Editor.

2. Enter the following **program code** into the editor, replacing the Bluetooth device address with the device address of the eb500 on the Board of Education board, which we obtained in the previous step. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file Connect.bs2.

```
{$STAMP BS2}
'Wait for the eb500 radio to be ready
```

```
PAUSE 1000

'Connect to the remote device
SEROUT 1,84,[“con 00:0C:84:00:05:29”,CR]
SERIN 0,84,[WAIT(“ACK”,CR)]

'Wait for the connection to be established and switch to data mode
WaitForConnection:
       IF in5 = 0 THEN WaitForConnection
HIGH 6
PAUSE 300

'Wait for 20 seconds
PAUSE 20000

'Switch to Command Mode
LOW 6
SERIN 0,84,[WAIT(CR,”>”)]

'Disconnect from the remote device
SEROUT 1,84,[“dis”,CR]
SERIN 0,84,[WAIT(CR,”>”)]
```

The BASIC Stamp application establishes a connection with the remote Bluetooth device, waits twenty seconds, switches back to command mode and then disconnects from the remote device.

3. On the **File** menu, click **Save As**.
4. In the **File name** box, enter a file name to which to save the program just created. For example, Connect.bs2.
5. Click **Save**.
6. Apply **power** to the **Board of Education board**.
7. Apply **power** to the **SumoBoard board**.

8. On the **Run** menu, click **Run**.

   The Connection Status LED (see Figure 1 on page 15) on both eb500 modules will turn on when a connection is established between the two eb500 modules.

9. Disconnect the **power** from the **Board of Education board**.

10. Disconnect **power** from the **SumoBoard board**.

Connecting a PC with an eb600 to a Board of Education

In this exercise we will step through the process of establishing a connection between a PC that has an eb600 RS232 Adapter to an eb500 inserted into a Board of Education board.

To perform this exercise, as documented, you will need an eb600 RS232 Adapter, a Board of Education board and two eb500 modules. If you are using any of the other supported Parallax boards, you may need to make adjustments to this exercise.

Step 1: eb600 RS232 Adapter Setup

In this step we will attach an eb500 module to the eb600 PC Adapter and apply power to the device.

1. Insert an **eb500** module into the **eb600 RS232 Adapter** header; assuring that Pin 1 of the eb500 module is inserted into Pin 1 of the header on the eb600 RS232 Adapter.
2. Connect the **eb600 RS232 Adapter** to a serial port on the PC using the **provided straight through serial cable**.

   The PC serial port must be available for HyperTerminal use.
3. Apply **power** to the **eb600 RS232 Adapter**.

Step 2: HyperTerminal Setup

In this step we will setup the Windows HyperTerminal application to establish a connection with the eb500 attached to the eb600 RS232 Adapter.

1. Open **HyperTerminal**.

   This will display the Connection Description dialog.
2. In the **Name** box, type the name of your connection. For example, eb600.
3. Click **OK**.

   This will display the Connect To dialog.
4. In the **Connect using** dropdown, select the **serial port** to which you have connected the eb600 RS232 Adapter.
5. Click **OK**.

   This will display the Properties dialog.
6. In the **Bits per second** dropdown, select **9600**.
7. In the **Data bits** dropdown, select **8**.

8. In the **Parity** dropdown, select **None**.
9. In the **Stop bits** dropdown, select **1**.
10. In the **Flow control** dropdown, select **None**.
11. Click **OK**.

    This will establish a connection with the serial port.
12. On the **Call** menu, click **Disconnect**.

    This will disconnect the connection just established, so that we can modify the connection properties in the following actions.
13. On the **File** menu, click **Properties**.

    This will display the Properties dialog.
14. On the **Settings** tab, click **ASCII Setup**.

    This will display the ASCII Setup dialog.
15. Check the **Send line ends with line feeds** checkbox.
16. Check the **Echo typed characters locally** checkbox.
17. Check the **Append line feeds to incoming line ends** checkbox.
18. Check the **Wrap lines that exceed terminal width** checkbox.
19. Click **OK**.

    This will return to the Properties dialog.
20. Click **OK**.
21. On the **Call** menu, click **Call**.

    This will establish a connection with the serial port.

Step 3: Board of Education – eb500 Setup

In this step we will attach an eb500 module to the Board of Education board and apply power to the device.

1. Insert an **eb500** module into the **AppMod header** of the Board of Education board; assuring that Pin 1 of the eb500 module is inserted into the VSS pin of the AppMod header.

2. Apply **power** to the **Board of Education board**.

   Power can be applied by attaching a 9 Volt battery, or the AC-Adapter provided by Parallax.

Step 4: Establish a Connection

In this step we will establish a connection between the PC and the Board of Education.

1. Using HyperTerminal, get the **address** of the **eb500** module that is connected to the Board of Education board by using the eb500 LIST VISIBLE serial command.

   By issuing the LIST VISIBLE command, the eb500 connected to the eb600 lists other Bluetooth devices that are in range and visible. Please note that this operation will take 30 seconds to complete.

   To obtain the address, type `lst visible` at the "`>`" prompt and press the return key.

   Example:

   ```
   >lst visible
   ACK
   00:0C:84:00:07:D7
   >
   ```

2. Using HyperTerminal, establish a **connection** with the **eb500** that is connected to the Board of Education board by using the eb500 CONNECT serial command.

   To establish a connection, type `con` followed by a space, followed by the address returned in the previous action, followed by a carriage-return. The Connection Status LED (Figure 1) on both eb500 modules will turn on when a connection is established.

   Example:

   ```
   >con 00:0C:84:00:07:D7
   ACK
   >
   ```

3. Disconnect **power** from both the **eb600 PC Adapter** and the **Board of Education boards**.

   The removal of power resets the eb500 so that when power is restored the eb500 will boot into command mode.

Connecting a PC with a DBT-120 to a Board of Education

In this exercise we will step through the process of establishing a connection from a PC that has a D-Link® DBT-120 Bluetooth USB Adapter to an eb500 module inserted into a Board of Education (BOE) board.

To perform this exercise, as documented, you will need a D-Link DBT-120, a Board of Education board, and an eb500 module. If you are using any of the other supported Parallax boards, you may need to make adjustments to this exercise.

On the PC, the DBT-120 Bluetooth Software associates a COM port for establishing a connection from the PC to a remote Bluetooth device and a separate COM port for connections that are established from a remote Bluetooth device to the PC. This exercise demonstrates establishing a connection from the PC to a remote eb500. The next exercise will demonstrate establishing a connection from a remote eb500 to the PC.

The D-Link DBT-120 Bluetooth USB Adapter software must be fully installed prior to establishing a connection. The PC settings shown in this exercise are based upon the software provided with the D-Link DBT-120 Bluetooth USB Adapter.

Step 1: DBT-120 Setup

In this step we will attach the DBT-120 USB Adapter to the PC. The software for the DBT-120 should already be setup.

1. Connect the **DBT-120** to an available **USB port** on the PC, following the instructions provided with the DBT-120 Bluetooth USB Adapter.

Step 2: Board of Education – eb500 Setup

In this step we will attach an eb500 module to the Board of Education board and apply power to the device.

1. Insert an **eb500** module into the **AppMod header** of the Board of Education board; assuring that Pin 1 of the eb500 module is inserted into the VSS pin of the AppMod header.
2. Apply **power** to the **Board of Education board**.

   Power can be applied by attaching a 9 Volt battery, or the AC-Adapter provided by Parallax.

Step 3: Establishing trust between the PC and the eb500 module

In this step we will establish a trusted relationship between the PC and the eb500 module. Verifying passkeys is required when performing the initial connection on two devices that require security. If you have connected these two devices once with security enabled then this step should be skipped.

The actions in this step need to be performed only once for the eb500. After performing the actions in this step, the connection security details will be stored on both the PC and the eb500 module. Therefore, future connections can be established to an eb500 by simply opening the associated COM port.

1. Open the **Bluetooth Devices** dialog by double-clicking on the Bluetooth tray icon.

   This will display the Bluetooth Devices dialog.

2. Click **Find Bluetooth Devices** to locate the **eb500 module** connected to the Board of Education board.

   Provided the eb500 on the Board of Education board is within range, eb500 will be shown in the window.

3. Right click on the eb500 and click **Connect A7 Serial Port**.

   This will bring up the Bluetooth PIN Code Request dialog since we have not connected to this device in the past.

4. Enter the Bluetooth **PIN Code** for the eb500.

   The factory default passkey is **0000**. The eb500 module enables security by default so the devices must verify passkeys to establish a trusted relationship before they can communicate.

   Note about terminology: The DBT-120 software uses the term “PIN Code” as a substitute to the eb500’s “Passkey”. These two terms refer to the same idea of using a secret code to establish a connection.

Step 4: Establish a Connection Using the DBT-120 Bluetooth Software

In this step we will establish a connection from the PC to the eb500 module inserted into the Board of Education board.

The actions in this step need to be performed only once for the eb500. After performing the actions in this step, the connection details will be stored on the PC. Therefore, future connections can be established to an eb500 by simply opening the associated COM port.

1. Open **My Bluetooth Places** by double-clicking on the desktop icon.

2. Click **Find Bluetooth Devices** to locate the **eb500 module** connected to the Board of Education.

   Provided the eb500 on the Board of Education is within range, eb500 will be shown in the window.

3. Right click on **eb500** and click **Discover Available services**.

   The A7 Serial Port service will be shown in the window.

4. Right click on **A7 Serial Port** and click **Connect to Bluetooth Serial Port**.

   This will establish a connection from the PC to the eb500 on the Board of Education board and associate this connection with a specific COM port.

5. If the **A7 Serial Port** dialog is shown, click **OK**.

6. Right click on **A7 Serial Port on eb500** and click **Properties**.

   This will display the Bluetooth Properties dialog.

7. In the **Port** dropdown, which is disabled, please note the **COM port** shown.

   The DBT-120 Bluetooth software associates a specific COM port for a connection from the PC to an eb500. Applications, such as HyperTerminal, use this COM port to establish a connection and communicate with an eb500 from the PC. Remember, this COM port is used to establish a connection from the PC to the eb500. A different COM port is used when a connection is established from the eb500 to the PC.

8. Click **OK**.

9. Select **A7 Serial Port** and click **Disconnect Bluetooth Serial Port**.

   This will disconnect the wireless connection to the eb500 on the Board of Education board.

Step 5: HyperTerminal Setup

In this step we will setup the Windows HyperTerminal application to establish a connection with the eb500 on the Board of Education board.

1. Open **HyperTerminal**.

   This will display the Connection Description dialog.

2. In the **Name** box, type the name of your connection. For example, eb500-BOE.

3. Click **OK**.

   This will display the Connect To dialog.

4. In the **Connect using** dropdown, select the **serial port** associated with the DBT-120 Bluetooth connection discovered in the previous step.
5. Click **OK**.

   This will display the Properties dialog.
6. In the **Bits per second** dropdown, select **9600**.
7. In the **Data bits** dropdown, select **8**.
8. In the **Parity** dropdown, select **None**.
9. In the **Stop bits** dropdown, select **1**.
10. In the **Flow control** dropdown, select **None**.
11. Click **OK**.

    This will establish a connection with the eb500 on the Board of Education board.
12. On the **Call** menu, click **Disconnect**.

    This will disconnect the connection just established, so that we can modify the connection properties in the following actions.
13. On the **File** menu, click **Properties**.

    This will display the Properties dialog.
14. On the **Settings** tab, click **ASCII Setup**.

    This will display the ASCII Setup dialog.
15. Check the **Send line ends with line feeds** checkbox.
16. Check the **Echo typed characters locally** checkbox.
17. Check the **Append line feeds to incoming line ends** checkbox.
18. Check the **Wrap lines that exceed terminal width** checkbox.
19. Click **OK**.

    This will return to the Properties dialog.
20. Click **OK**.

Step 6: Establish a Connection Using HyperTerminal

In this step we will establish a connection from the PC to the eb500 on the Board of Education, using HyperTerminal. This step relies on the connection information created previously.

1. On the **Call** menu, click **Call**.

   This will establish a connection with the eb500 on the Board of Education board. The Connection Status LED (see Figure 1 on page 15) on the eb500 module will turn on when a connection is established.

2. On the **Call** menu, click **Disconnect**.

   This will close the connection with the eb500 on the Board of Education.

Connecting a Board of Education to a PC with a DBT-120

In this exercise we will step through the process of establishing a connection from an eb500 module inserted into a Board of Education (BOE) board to a PC that has a D-Link® DBT-120 Bluetooth USB Adapter.

To perform this exercise, as documented, you will need a D-Link DBT-120, a Board of Education board, and an eb500 module. If you are using any of the other supported Parallax boards, you may need to make adjustments to this exercise.

On the PC, the DBT-120 Bluetooth Software associates a COM port for establishing a connection from the PC to a remote Bluetooth device and a separate COM port for connections that are established from a remote Bluetooth device to the PC. This exercise demonstrates establishing a connection from a remote eb500 to the PC. When a remote Bluetooth device establishes a connection with the PC, the connection is established with the DBT-120 Bluetooth USB Adapter software running on the PC. To gain access to the data, an application, such as HyperTerminal, must open the COM port associated with the connection established from the remote device. In the Communications section, we will step through this process.

The D-Link DBT-120 Bluetooth USB Adapter software must be fully installed prior to establishing a connection. The PC settings shown in this exercise are based upon the software provided with the D-Link DBT-120 Bluetooth USB Adapter.

Step 1: DBT-120 Setup

In this step we will attach the DBT-120 USB Adapter to the PC. The software for the DBT-120 should already be setup.

1. Connect the **DBT-120** to an available **USB port** on the PC, following the instructions provided with the DBT-120 Bluetooth USB Adapter.

Step 2: Obtain the Bluetooth Address of the PC

In this step we will obtain the Bluetooth address of the DBT-120 USB Adapter attached to the PC.

1. Open **My Bluetooth Places** by double-clicking on the desktop icon.
2. Right click on **My Device** and click **Properties**.

   This will display the Bluetooth Configuration dialog.

3. Select the **Hardware** tab and note the **Device Address** shown in the Device Properties section of the dialog.

   The device address will be used in the BASIC Stamp application developed in the next step.

4. Click **Cancel**

   This will close the Bluetooth Configuration dialog.

Step 3: Write a BASIC Stamp Application to Connect to the PC

In this step we will attach an eb500 module to the Board of Education board and develop a BASIC Stamp application to establish a connection with the PC.

1. Insert an **eb500** module into the **AppMod connector** of the Board of Education board; assuring that Pin 1 of the eb500 module is inserted into the VSS pin of the AppMod header.
2. Connect the **Board of Education board serial port** to the **PC**.
3. Open the **BASIC Stamp Editor**.
4. Enter the following **program code** into the editor, replacing the Bluetooth device address with the device address of the PC, which we obtained from the Hardware tab of the Device Properties section of the Bluetooth Configuration dialog in the previous step. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file Connect.bs2.

```
{$STAMP BS2}
'Wait for the eb500 radio to be ready
PAUSE 1000

'Connect to the remote device
SEROUT 1,84,[“con 00:0C:84:00:05:29”,CR]
SERIN 0,84,[WAIT(“ACK”,CR)]

'Wait for the connection to be established and switch to data mode
WaitForConnection:
       IF in5 = 0 THEN WaitForConnection
HIGH 6
PAUSE 300
```

```
'Wait for 20 seconds
PAUSE 20000

'Switch to Command Mode
LOW 6
SERIN 0,84,[WAIT(CR,”>”)]

'Disconnect from the remote device
SEROUT 1,84,[“dis”,CR]
SERIN 0,84,[WAIT(CR,”>”)]
```

The BASIC Stamp application establishes a connection with the PC device, waits twenty seconds, switches back to command mode and then disconnects from the PC.

5. On the **File** menu, click **Save As**.
6. In the **File name** box, enter a file name to which to save the program just created. For example, Connect.bs2.
7. Click **Save**.

Step 4: Establishing trust between the PC and the eb500 module

In this step we will establish a trusted relationship between the PC and the eb500 module. Verifying passkeys is required when performing the initial connection on two devices that require security. If you have connected these two devices once with security enabled then this step should be skipped.

The actions in this step need to be performed only once for the eb500. After performing the actions in this step, the connection security details will be stored on both the PC and the eb500 module. Therefore, future connections can be established to an eb500 by simply opening the associated COM port.

1. Open the **Bluetooth Devices** dialog by double-clicking on the Bluetooth tray icon.

   This will display the Bluetooth Devices dialog.

2. Click **Find Bluetooth Devices** to locate the **eb500 module** connected to the Board of Education board.

   Provided the eb500 on the Board of Education board is within range, eb500 will be shown in the window.

3. Right click on the eb500 and click **Connect A7 Serial Port**.

This will bring up the Bluetooth PIN Code Request dialog since we have not connected to this device in the past.

4. Enter the Bluetooth **PIN Code** for the eb500.

   The factory default passkey is **0000**. The eb500 module enables security by default so the devices must verify passkeys to establish a trusted relationship before they can communicate.

   Note about terminology: The DBT-120 software uses the term "PIN Code" as a substitute to the eb500's "Passkey". These two terms refer to the same idea of using a secret code to establish a connection.

Step 5: Connect the eb500 on the Board of Education to the PC

1. Apply **power** to the **Board of Education board**.

   Power can be applied by attaching a 9 Volt battery, or the AC-Adapter provided by Parallax.

2. On the **Run** menu, click **Run**.

   The Connection Status LED (see Figure 1 on page 15) on the eb500 module will turn on when a connection is established. Additionally, on the My Bluetooth Places window, in the Additional Information column, the text "Connected" will be shown while a connection exists between the eb500 and the PC.

Connecting a PC with XP SP2 to a Board of Education

In this exercise we will step through the process of establishing a connection from a PC with a Bluetooth USB adapter that is running Windows XP SP2 to an eb500 module in pass-through mode.

To perform this exercise, as documented, you will need a PC running Windows XP SP2, a Bluetooth USB adapter, a Parallax Board of Education, and an eb500 module. If you are using any of the other supported Parallax boards, you may need to make adjustments to this exercise.

On the PC, the Microsoft Bluetooth Software associates a COM port for establishing a connection from the PC to a remote Bluetooth device and a separate COM port for connections that are established from a remote Bluetooth device to the PC. This exercise demonstrates establishing a connection from the PC to a remote eb500. The next exercise will demonstrate establishing a connection from a remote eb500 to the PC.

Step 1: Bluetooth USB Adapter Setup

In this step we will attach the Bluetooth USB Adapter to the PC. Windows XP SP2 should automatically detect and configure the adapter for use.

1. Connect the **Bluetooth USB Adapter** to an available **USB port** on the PC.

Step 2: Board of Education – eb500 Setup

In this step we will attach an eb500 module to the Board of Education board and apply power to the device.

1. Insert an **eb500** module into the **AppMod header** of the Board of Education board; assuring that Pin 1 of the eb500 module is inserted into the VSS pin of the AppMod header.
2. Apply **power** to the **Board of Education board**.

   Power can be applied by attaching a 9 Volt battery, or the AC-Adapter provided by Parallax.

Step 3: Establishing trust between the PC and the eb500 module

In this step we will establish a trusted relationship between the PC and the eb500 module. Verifying passkeys is required when performing the initial connection on two devices that require security. If you have connected these two devices once with security enabled then this step should be skipped.

The actions in this step need to be performed only once for the eb500. After performing the actions in this step, the connection security details will be stored on both the PC and the eb500 module. Therefore, future connections can be established to an eb500 by simply opening the associated COM port.

1. Open the **Bluetooth Devices** dialog by double-clicking on the Bluetooth tray icon.

   This will display the Bluetooth Devices dialog.

2. Click **Add** to open the Add Bluetooth Device Wizard

   The Windows XP SP2 Bluetooth Software requires that devices are added before they can be used.

3. Click **My device is set up and ready to be found** and then click **Next** to locate the eb500 module connected to the Board of Education board.

   Provided the eb500 connected to the Board of Education board is within range, eb500 will be shown in the window.

4. Select **eb500** and click **Next**.

   The passkey selection dialog will be shown. The eb500 module enables security by default so the devices must verify passkeys to establish a trusted relationship before they can communicate.

5. Select **Use the passkey found in the documentation,** enter **0000** into the edit field, and click **Next**.

   This will establish a connection from the PC to the eb500 on the Board of Education and associate this connection with a specific COM port.

6. Please note both the Outgoing and Incoming **COM ports** shown.

   The Microsoft Bluetooth software associates a specific COM port for a connection from the PC to an eb500. Applications, such as HyperTerminal, use this COM port to establish a connection and communicate with an eb500 from the PC. Remember, this COM port is used to establish a connection from the PC to the eb500. A different COM port is used when a connection is established from the eb500 to the PC.

7. Click **Finish**.

   This will complete the wizard and close the Add Bluetooth Device Wizard.

Step 4: HyperTerminal Setup

In this step we will setup the Windows HyperTerminal application to establish a connection with the eb500 on the Board of Education.

1. Open **HyperTerminal**.

   This will display the Connection Description dialog.

2. In the **Name** box, type the name of your connection. For example, eb500-BOE
3. Click **OK**.

   This will display the Connect To dialog.

4. In the **Connect using** dropdown, select the **serial port** to which the Microsoft Bluetooth software associated with the connection from the PC to the eb500 on the Board of Education.

   The COM port associated with the connection was discovered in the previous step.

5. Click **OK**.

   This will display the Properties dialog.

6. In the **Bits per second** dropdown, select **9600**.
7. In the **Data bits** dropdown, select **8**.
8. In the **Parity** dropdown, select **None**.
9. In the **Stop bits** dropdown, select **1**.
10. In the **Flow control** dropdown, select **None**.
11. Click **OK**.

    This will establish a connection with the eb500 on the Board of Education.

12. On the **Call** menu, click **Disconnect**.

    This will disconnect the connection just established, so that we can modify the connection properties in the following actions.

13. On the **File** menu, click **Properties**.

    This will display the Properties dialog.

14. On the **Settings** tab, click **ASCII Setup**.

    This will display the ASCII Setup dialog.

15. Check the **Send line ends with line feeds** checkbox.

16. Check the **Echo typed characters locally** checkbox.
17. Check the **Append line feeds to incoming line ends** checkbox.
18. Check the **Wrap lines that exceed terminal width** checkbox.
19. Click **OK**.

    This will return to the Properties dialog.
20. Click **OK**.

Step 5: Establish a Connection from the PC Using HyperTerminal

In this step we will establish a connection from the PC to the eb500 on the Board of Education, using HyperTerminal. This step relies on the connection information created previously in Step 3.

1. On the **Call** menu, click **Call**.

   This will establish a connection with the eb500 on the Board of Education. The Connection Status LED (Figure 1) on the eb500 module will turn on when a connection is established. By default, the eb500 is in data mode.
2. On the **Call** menu, click **Disconnect**.

   This will close the connection with the eb500 on the Board of Education.

Connecting a Board of Education to a PC with XP SP2

In this exercise we will step through the process of establishing a connection from an eb500 module inserted into a Board of Education (BOE) to a PC that is running Windows XP SP2 and has a Bluetooth USB Adapter.

To perform this exercise, as documented, you will need a PC running Windows XP SP2, a Bluetooth USB Adapter, a Parallax Basic Stamp module, a Parallax Board of Education, and an eb500 module. If you are using any of the other supported Parallax boards, you may need to make adjustments to this exercise.

On the PC, the Microsoft Bluetooth Software associates a COM port for establishing a connection from the PC to a remote Bluetooth device and a separate COM port for connections that are established from a remote Bluetooth device to the PC. This exercise demonstrates establishing a connection from a remote eb500 to the PC. When a remote Bluetooth device establishes a connection with the PC, the connection is established with the Bluetooth USB Adapter and the software running on the PC. To gain access to the data, an application, such as HyperTerminal, must open the COM port associated with the connection established from the remote device. In the Communications section, we will step through this process.

Step 1: Bluetooth USB Adapter Setup

In this step we will attach the Bluetooth USB Adapter to the PC. Windows XP SP2 should automatically detect and configure the adapter for use.

1. Connect the **Bluetooth USB Adapter** to an available **USB port** on the PC.

Step 2: Obtain the Bluetooth Address of the PC

In this step we will obtain the Bluetooth address of the Bluetooth USB Adapter attached to the PC.

1. Open the **Bluetooth Devices** dialog by double-clicking on the Bluetooth tray icon.

   This will display the Bluetooth Devices dialog.

2. On the **Hardware** tab select **Generic Bluetooth Radio** and click on the **Properties** button.

   This will display the Generic Bluetooth Radio Properties dialog.

3. Select the **Advanced** tab and note the **Address** shown in the Radio Information section of the dialog.

   The device address will be used in the BASIC Stamp application developed in the next step.

4. Click **OK**

   This will close the Generic Bluetooth Radio Properties dialog.

5. Click **OK**

   This will close the Bluetooth Devices dialog.

Step 3: Establishing trust between the PC and the eb500 module

In this step we will establish a trusted relationship between the PC and the eb500 module. Verifying passkeys is required when performing the initial connection on two devices that require security. If you have already connected these two devices with security enabled then this step should be skipped.

The actions in this step need to be performed only once for the eb500. After performing the actions in this step, the connection security details will be stored on both the PC and the eb500 module. Therefore, future connections can be established to an eb500 by simply opening the associated COM port.

1. Open the **Bluetooth Devices** dialog by double-clicking on the Bluetooth tray icon.

   This will display the Bluetooth Devices dialog.

2. Click **Add** to open the Add Bluetooth Device Wizard

   The Windows XP SP2 Bluetooth Software requires that devices are added before they can be used.

3. Click **My device is set up and ready to be found** and then click **Next** to locate the eb500 module connected to the Board of Education.

   Provided the eb500 on the Board of Education is within range, eb500 will be shown in the window.

4. Select **eb500** and click **Next**.

   The passkey selection dialog will be shown. The eb500 module enables security by default so the devices must verify passkeys to establish a trusted relationship before they can communicate.

5. Select **Use the passkey found in the documentation,** enter **0000** into the edit field, and click **Next**.

   This will establish a connection from the PC to the eb500 on the Board of Education and associate this connection with a specific COM port.

6. Please note both the Outgoing and Incoming **COM ports** shown.

   The Microsoft Bluetooth software associates a specific COM port for a connection from the PC to an eb500. Applications, such as HyperTerminal, use this COM port to establish a connection and communicate with an eb500 from the PC. Remember, this COM port is used to establish a connection from the PC to the eb500. A different COM port is used when a connection is established from the eb500 to the PC.

7. Click **Finish**.

   This will complete the wizard and close the Add Bluetooth Device Wizard.

Step 4: Write a BASIC Stamp Application to Connect to the PC

In this step we will attach an eb500 module to the Board of Education board and develop a BASIC Stamp application to establish a connection with the PC.

1. Insert an **eb500** module into the **AppMod connector** of the Board of Education board; assuring that Pin 1 of the eb500 module is inserted into the VSS pin of the AppMod header.
2. Connect the **Board of Education board serial port** to the **PC**.
3. Open the **BASIC Stamp Editor**.
4. Enter the following **program code** into the editor, replacing the Bluetooth device address with the device address of the PC, which we obtained from the Hardware tab of the Device Properties section of the Bluetooth Configuration dialog in the previous step. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file Connect.bs2.

```
{$STAMP BS2}
'Wait for the eb500 radio to be ready
PAUSE 1000

'Connect to the remote device
SEROUT 1,84,[“con 00:0C:84:00:05:29”,CR]
SERIN 0,84,[WAIT(“ACK”,CR)]

'Wait for the connection to be established and switch to data mode
WaitForConnection:
       IF in5 = 0 THEN WaitForConnection
HIGH 6
PAUSE 300
```

```
'Wait for 20 seconds
PAUSE 20000

'Switch to Command Mode
LOW 6
SERIN 0,84,[WAIT(CR,”>”)]

'Disconnect from the remote device
SEROUT 1,84,[“dis”,CR]
SERIN 0,84,[WAIT(CR,”>”)]
```

The BASIC Stamp application establishes a connection with the PC device, waits twenty seconds, switches back to command mode and then disconnects from the PC.

5. On the **File** menu, click **Save As**.
6. In the **File name** box, enter a file name to which to save the program just created. For example, Connect.bs2.
7. Click **Save**.

Step 5: Connect the eb500 on the Board of Education to the PC

1. Apply **power** to the **Board of Education board**.

   Power can be applied by attaching a 9 Volt battery, or the AC-Adapter provided by Parallax.

2. On the **Run** menu, click **Run**.

   The Connection Status LED (see Figure 1 on page 15) on the eb500 module will turn on when a connection is established. Additionally, on the My Bluetooth Places window, in the Additional Information column, the text “Connected” will be shown while a connection exists between the eb500 and the PC.

Connecting a Pocket PC 2003 device to a Board of Education

In this exercise we will step through the process of establishing a connection from a Pocket PC 2003 device with integrated Bluetooth, to an eb500 module inserted into a Board of Education board.

To perform this exercise, as documented, you will need a Pocket PC 2003 device, a Parallax Board of Education, and an eb500 module. Depending on the specific Pocket PC model that you are using, you may need to make minor adjustments to this exercise.

Step 1: Board of Education – eb500 Setup

In this step we will attach an eb500 module to the Board of Education board and apply power to the device.

1. Insert an **eb500** module into the **AppMod connector** of the Board of Education board; assuring that Pin 1 of the eb500 module is inserted into the VSS pin of the AppMod header.
2. Apply **power** to the **Board of Education board**.

   Power can be applied by attaching a 9 Volt battery, or the AC-Adapter provided by Parallax.

Step 2: Pocket PC 2003 Setup

In this step we will setup the Pocket PC for connecting to the eb500.

1. Tap the **Bluetooth icon** in the system tray on the Today screen and select **Bluetooth Manager**.

   This will display the Bluetooth Manager dialog.
2. On the **New** menu, select **Connect**.

   This will display the first page of the Connection Wizard.
3. Select **Explore a Bluetooth device** and tap **Next**.

   This will display the next page of the Connection Wizard.
4. Tap in the **Device** box.

   This will display the Connection Wizard Bluetooth Browser dialog containing a list of found devices.
5. Tap **eb500**.

   This will display the next page of the Connection Wizard.

6. In the **Service Selection** box, select **A7 Serial Port**.
7. Tap **Next**.

   This will create a shortcut for the service.
8. Tap **Finish**.

   This will display the Bluetooth Manager dialog with the shortcut created in the window, eb500: A7 Serial Port.

Step 3: Establish a Connection

In this step we will establish a connection from the Pocket PC to the eb500.

1. Tap-and-hold the shortcut created in the previous step, **eb500: A7 Serial Port**.
2. Select **Connect**.

   This will establish a connection with the eb500 on the Board of Education. The Connection Status LED (see Figure 1 on page 15) on the eb500 module will turn on when a connection is established.
3. Tap **Active Connections**.

   This will display the Bluetooth Manager Active Connections page showing the status of your active Bluetooth connections.
4. Tap **My Shortcuts**.
5. Tap-and-hold the shortcut created in the previous step, **eb500: A7 Serial Port**.
6. Select **Disconnect**.

   This will close the connection with the eb500 on the Board of Education board. The Connection Status LED on the eb500 module will turn off.

Connecting a Board of Education to a Pocket PC 2003 device

In this exercise we will step through the process of establishing a connection from an eb500 module attached to a Board of Education board to a Pocket PC 2003 device with integrated Bluetooth.

To perform this exercise, as documented, you will need a Pocket PC 2003 device, a Parallax Board of Education, and an eb500 module. Depending on the specific Pocket PC model that you are using, you may need to make minor adjustments to this exercise.

Step 1: Obtain the Bluetooth Address of the Pocket PC

In this step we will obtain the Bluetooth address of the Pocket PC.

1. Tap the **Bluetooth icon** in the system tray on the Today screen and select **Bluetooth Settings**.

   This will display the Settings dialog.

2. Tap the **Accessibility** tab and note the **Address** shown in the Device Identification section of the dialog.

   The device address will be used in the Basic application developed in the next step.

3. Tap **OK** to close the dialog.

Step 2: Write a Basic Application to Connect to the Pocket PC

In this step we will attach an eb500 module to the Board of Education and develop a Basic application to establish a connection with the Pocket PC.

1. Insert an **eb500** module into the **AppMod header** of the Board of Education board; assuring that Pin 1 of the eb500 module is inserted into the VSS pin of the AppMod header.
2. Connect the **Board of Education board serial port** to the **PC**.
3. Open the **BASIC Stamp Editor**.
4. Enter the following **program code** into the editor, replacing the Bluetooth device address with the device address of the Pocket PC, which we obtained in the previous step. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file Connect.bs2.

```
{$STAMP BS2}
'Wait for the eb500 radio to be ready
PAUSE 1000
```

```
'Connect to the remote device
SEROUT 1,84,[“con 00:0C:84:00:05:29”,CR]
SERIN 0,84,[WAIT(“ACK”,CR)]

'Wait for the connection to be established and switch to data mode
WaitForConnection:
       IF in5 = 0 THEN WaitForConnection
HIGH 6
PAUSE 300

'Wait for 20 seconds
PAUSE 20000

'Switch to Command Mode
LOW 6
SERIN 0,84,[WAIT(CR,”>”)]

'Disconnect from the remote device
SEROUT 1,84,[“dis”,CR]
SERIN 0,84,[WAIT(CR,”>”)]
```

The BASIC Stamp application establishes a connection with the remote Bluetooth device, waits twenty seconds, switches back to command mode and then disconnects from the remote device.

On most Pocket PC 2003 devices, the Bluetooth software closes the connection after a short period of time if there is not an application running on the device to receive the data over the connection. Therefore, after the twenty second wait, the application checks to see if there is still a valid connection before switching to Command Mode. If there is no connection, the eb500 is already in Command Mode.

5. On the **File** menu, click **Save As**.
6. In the **File name** box, enter a file name to which to save the program just created. For example, Connect.bs2.
7. Click **Save**.

Step 3: Establish a Connection

In this step we will establish a connection from the Board of Education board to the Pocket PC.

1. Turn on the **Pocket PC 2003 device**.
2. Tap the **Bluetooth icon** and select **Bluetooth Manager**.

   This will display the Bluetooth Manager dialog.
3. Tap the **Active Connections** tab.
4. Apply **power** to the **Board of Education board**.

   Power can be applied by attaching a 9 Volt battery, or the AC-Adapter provided by Parallax.
5. Using the Basic Stamp Editor, on the **Run** menu, click **Run**.

   Depending on your current Pocket PC Bluetooth configuration, the Authorization Requested Dialog may appear (Figure 10). If this dialog appears, tap Accept to accept the connection. The Connection Status LED (see Figure 1 on page 15) on the eb500 module will turn on when a connection is established. On the Pocket PC the connection will be shown in the Incoming Connections section of the Active Connections tab on the Bluetooth Active Connections dialog.

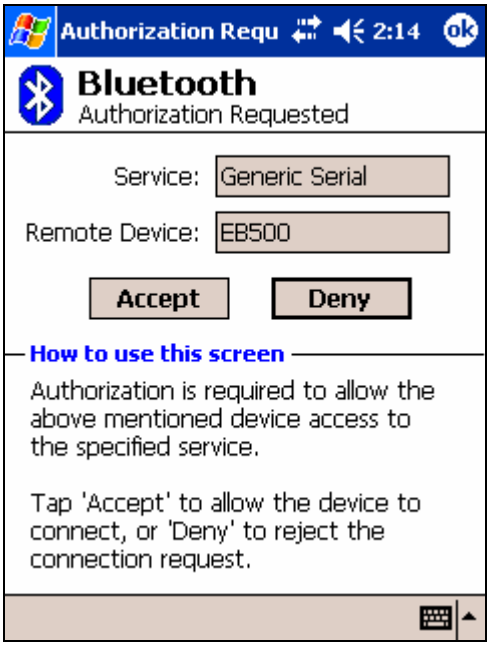


Figure 10: Pocket PC Bluetooth Authorization Request Dialog

This page intentionally left blank.

Communications

This section contains a number of exercises that demonstrate methods of communicating over a Bluetooth wireless connection with the eb500. The scenarios described are not meant to form an exhaustive list, but rather illustrate a number of more common and useful configurations. All source code shown in these exercises is available in electronic form on the accompanying CD, in the Samples folder, using the filename used in this manual.

Communicating between Two eb500 Modules

In this exercise we will step through the process of communicating wirelessly between two eb500 modules, one inserted into a Boe-Bot robot and the other inserted into a SumoBot robot. We will program the SumoBot to use its infrared sensors to follow an object and then transmit its movements to the Boe-Bot. The Boe-Bot will use the received information to mimic the movements of the SumoBot.

To perform this exercise, as documented, you will need a Boe-Bot, a SumoBot, and two eb500 modules. If you are using any of the other supported Parallax robots, you may need to make adjustments to this exercise.

Step 1: Create a Monkey-See Application for the SumoBot

In this step we will create a BASIC Stamp application that will use the infrared sensors of the SumoBot to follow an object and transmit its movements to a remote eb500.

1. Open the **BASIC Stamp Editor**.
2. Enter the following **program code** into the editor, replacing the Bluetooth device address with the device address of the eb500 inserted into the Boe-Bot robot. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file MonkeySee.bs2.

```
'{$STAMP BS2}
‘I/O Line 5 provides the connection status
INPUT 5
```

```
'-----[I/O Definitions]-------------------------------------
LMotor  CON     13
RMotor  CON     12
LfIrOut CON     4
LfIrIn  VAR     In11
RtIrOut CON     15
RtIrIn  VAR     In14

'-----[Constants]-------------------------------------------
LFwdFast        CON     1000
LRevFast        CON     500
RFwdFast        CON     500
RRevFast        CON     1000

'-----[Variables]-------------------------------------------
irBits  VAR     NIB
irLeft  VAR     irBits.Bit1
irRight VAR     irBits.Bit0
lastIr  VAR     NIB
bBuffer VAR     BYTE(4)
bErrorCode      VAR     BYTE

'-----[Initialization]--------------------------------------
'Wait for the eb500 radio to be ready
PAUSE 1000

Connect:
        'Connect to Monkey-Do
        SEROUT 1,84,["con 00:0C:84:00:07:D7",CR]
        SERIN 0,84,[WAIT(“ACK”,CR)]
        ‘Either an Err #<CR> or a ">" will be received
        SERIN 0,84,[STR bBuffer\6\”>”]
        IF bBuffer(0) = “E” THEN ErrorCode

WaitForConnection:
```

```
        IF in5 = 0 THEN WaitForConnection

'-----[Main Code]-----------------------------------------
Main:
        'Verify the connection is still up before each loop
        IF in5 = 0 THEN Connect
        GOSUB Read_IR_Sensors
        BRANCH irBits,[Hold, Turn_Right, Turn_left, Move_Fwd]

Move_Fwd:
        SEROUT 1,84,["3"]
        PULSOUT LMotor,LFwdFast
        PULSOUT RMotor,RFwdFast
        GOTO Main

Turn_Right:
        SEROUT 1,84,["1"]
        PULSOUT LMotor,LFwdFast
        PULSOUT RMotor,RRevFast
        GOTO Main

Turn_Left:
        SEROUT 1,84,["2"]
        PULSOUT LMotor,LRevFast
        PULSOUT RMotor,RFwdFast
        GOTO Main

Hold:
        GOTO Main

'-----[Subroutines]---------------------------------------
Read_IR_Sensors:
        FREQOUT LfIrOut,1,38500
        irLeft = ~LfIrIn
        FREQOUT RtIrOut,1,38500
```

```
        irRight = ~RtIrIn
        RETURN

BadCommand:
        DEBUG "A bad command was received."
        END

ErrorCode:
        bErrorCode = bBuffer(4)
        DEBUG "An error was received: ",STR bErrorCode,CR
        END
```

3. On the **File** menu, click **Save As**.
4. In the **File name** box, enter a file name to which to save the program just created. For example, MonkeySee.bs2.
5. Click **Save**.

Step 2: Create a Monkey-Do Application for the Boe-Bot

In this step we will create a BASIC Stamp application that will receive information from the remote SumoBot and perform movements based on that information.

1. On the **File** menu, click **New**.

   This will create a new project window within the BASIC Stamp Editor.

2. Enter the following **program code** into the editor. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file MonkeyDo.bs2.

```
'{$STAMP BS2}
'-----[I/O Definitions]-------------------------------------
LMotor          CON     15
RMotor          CON     14

'-----[Constants]-------------------------------------------
LFwdFast        CON     1000
LRevFast        CON     500
RFwdFast        CON     500
```

```
RRevFast       CON     1000

'-----[Variables]------------------------------------------
CmdData VAR     BYTE

'-----[Initialization]--------------------------------------
Initialize:
        'Wait for the eb500 radio to be ready
        PAUSE 1000
        'Set the initial state to hold
        CmdData = 3

'-----[Main Code]-------------------------------------------
Main:
        'Wait for a command
        SERIN 0,84,[DEC1 CmdData]

        'Process the command
        BRANCH CmdData,[Hold, Turn_Right, Turn_left, Move_Fwd]

        'If the command was invalid just loop again
        GOTO Main

Move_Fwd:
        PULSOUT LMotor,LFwdFast
        PULSOUT RMotor,RFwdFast
        GOTO Main

Turn_Right:
        PULSOUT LMotor,LFwdFast
        PULSOUT RMotor,RRevFast
        GOTO Main

Turn_Left:
        PULSOUT LMotor,LRevFast
```

```
        PULSOUT RMotor,RFwdFast
        GOTO Main

Hold:
        GOTO Main
```

3. On the **File** menu, click **Save As**.
4. In the **File name** box, enter a file name to which to save the program just created. For example, MonkeyDo.bs2.
5. Click **Save**.

Step 3: Download the Applications to the Robots

In this step we will download the applications we just created to the respective robots.

1. Click the **MonkeySee.bs2** tab in the BASIC Stamp Editor.
2. Connect the **SumoBoard board serial port** to the **PC**.
3. Apply **power** to the **SumoBoard board**.
4. On the **Run** menu, click **Run**.
5. On the **Debug Terminal #1** dialog click **Close**.
6. Disconnect the **power** from the **SumoBoard board**.
7. Disconnect the **SumoBoard board serial port** from the **PC**.
8. Click the **MonkeyDo.bs2 tab** in the BASIC Stamp Editor.
9. Connect the **Board of Education board serial port** to the **PC**.
10. Apply **power** to the **Board of Education board**.
11. On the **Run** menu, click **Run**.
12. Disconnect the **Board of Education board serial port** from the **PC**.

Step 4: Run the Monkey-See / Monkey-Do Applications

In this step we will run the Monkey-See / Monkey-Do applications.

1. Apply **power** to the **SumoBoard board**.
2. Make the **Boe-Bot** robot **mimic the movements** of the SumoBot by putting your hand in front of the SumoBot IR sensors.

   As you move your hand left, right and forward, the SumoBot will follow your hand and the Boe-Bot will mimic the same movements.

Communicating between a PC with an eb600 and a BOE

In this exercise we will step through the process of communicating between a PC that has a an eb600 RS232 Adapter and an eb500 module inserted into a Board of Education (BOE) board.

To perform this exercise, as documented, you will need to have two serial ports available on your PC, an eb600 PC Adapter, a Board of Education board, and two eb500 modules. One serial port will be used to connect the PC to the Board of Education serial port. The other serial port will be used to connect to the eb600 PC Adapter. If you are using any of the other supported Parallax boards, you may need to make adjustments to this exercise.

Step 1: Transmit Data from the PC to the BASIC Stamp

In this step we will create a BASIC Stamp application to read data from the eb500 and display the data in the BASIC Stamp Editor Debug window. We will then download and run the application.

1. Connect the **Board of Education board serial port** to the **PC**.
2. Open the **BASIC Stamp Editor**.
3. Enter the following **program code** into the editor. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file Receive.bs2.

```
‘{$STAMP BS2}
bData VAR BYTE
‘Wait for the eb500 radio to be ready
PAUSE 1000
Main:
        SERIN 0,84,[STR bData\1]
        DEBUG STR bData\1
        GOTO Main
```

The application waits for an individual byte of data to arrive and then displays the byte in the debug window and then repeats this process.

4. On the **File** menu, click **Save As**.
5. In the **File name** box, enter a file name to which to save the program just created. For example, Receive.bs2.
6. Click **Save**.

7. Apply **power** to the **Board of Education board**.
8. On the **Run** menu, click **Run**.
9. Establish a connection from the **PC** to the **Board of Education**.

   Please see the section titled Connecting a PC with an eb600 to a Board of Education on page 28 for information on establishing the connection.

10. Using HyperTerminal, type a series of **characters**.

    These characters will be transmitted over the wireless link, read by the BASIC Stamp application, and then displayed in the debug window (Figure 11).

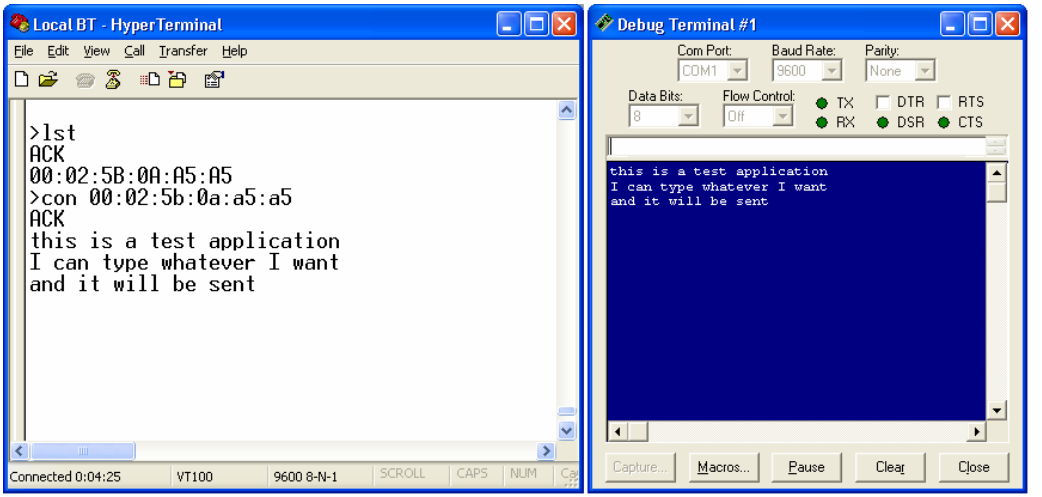


Figure 11: HyperTerminal Input and Debug Output

Step 2: Transmit Data from the BASIC Stamp to the PC

In this step we will create a BASIC Stamp application to send data out the eb500 to the PC where we will use HyperTerminal to display the data received by the eb500 module attached to the eb600 on the PC.

1. Reset the **eb500** attached to the **eb600 PC Adapter** to place the eb500 into command mode.

   To reset the eb500 attached to the eb600 PC Adapter, disconnect the power, wait a couple of seconds, and then reconnect the power.

2. Reset the **eb500** attached to the **Board of Education board** to place the eb500 into command mode.

   To reset the eb500 attached to the Board of Education board, disconnect the power, wait a couple of seconds, and then reconnect the power. The Reset push button on the Board of Education board will NOT reset the eb500.

3. Using HyperTerminal, acquire the **device address** of the **eb500** connected to the eb600 PC Adapter by using the eb500 GET ADDRESS serial command.

   Please note the device address as it will be used in the BASIC Stamp application developed in the following actions.

   By issuing the GET ADDRESS command, the eb500 connected to the eb600 will return its own device address.

   To obtain the device address, type `get address` at the "`>`" prompt and press the return key.

   Example:

   ```
   >get address
   ACK
   00:0C:84:00:07:D8
   >
   ```

4. Using the BASIC Stamp Editor, on the **File** menu, click **New**.

   This will create a new project window within the BASIC Stamp Editor.

5. Enter the following **program code** into the editor, replacing the device address with the device address obtained from the GET ADDRESS command issued above. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file HelloWorld.bs2.

```
'{$STAMP BS2}
nCount VAR BYTE
'Wait for the eb500 radio to be ready
PAUSE 1000

'Connect to the remote device
SEROUT 1,84,["con 00:0C:84:00:05:29",CR]
SERIN 0,84,[WAIT("ACK",CR)]
```

```
'Wait for the connection to be established and switch into data mode.
'When switching into data mode, a 300ms timeout is required to give the
'module enough time to make the change.
WaitForConnection:
        IF in5 = 0 THEN WaitForConnection
HIGH 6
PAUSE 300

DEBUG "Connection established",CR

'Send "Hello World" ten times
FOR nCount = 1 to 10
        SEROUT 1,84,["Hello World",CR]
        PAUSE 1000
NEXT

'Switch to Command Mode
LOW 6
SERIN 0,84,[WAIT(CR,”>”)]

'Disconnect from the remote device
SEROUT 1,84,[“dis”,CR]
SERIN 0,84,[wait(CR,”>”)]

DEBUG “Disconnected”,CR
```

The application establishes a connection with the remote eb500 device, transmits “Hello World” ten times, switches back to command mode and then disconnects from the remote device.

The first call to SEROUT is used when the eb500 is in command mode and instructs the eb500 to establish a connection with the device specified. Once a connection is established the eb500 is in data mode, which causes further calls to SEROUT to be sent to the remote device.

6. On the **File** menu, click **Save As**.

   This will display the Save As dialog.

7. In the **File name** box, enter a file name to which to save the program just created. For example, HelloWorld.bs2.
8. Click **Save**.
9. On the **Run** menu, click **Run**.

   This will display the Download Program dialog while downloading the program to the BASIC Stamp. After the download is complete the BASIC Stamp application will transmit “Hello World” over the wireless link, and HyperTerminal will display the received data (Figure 12).

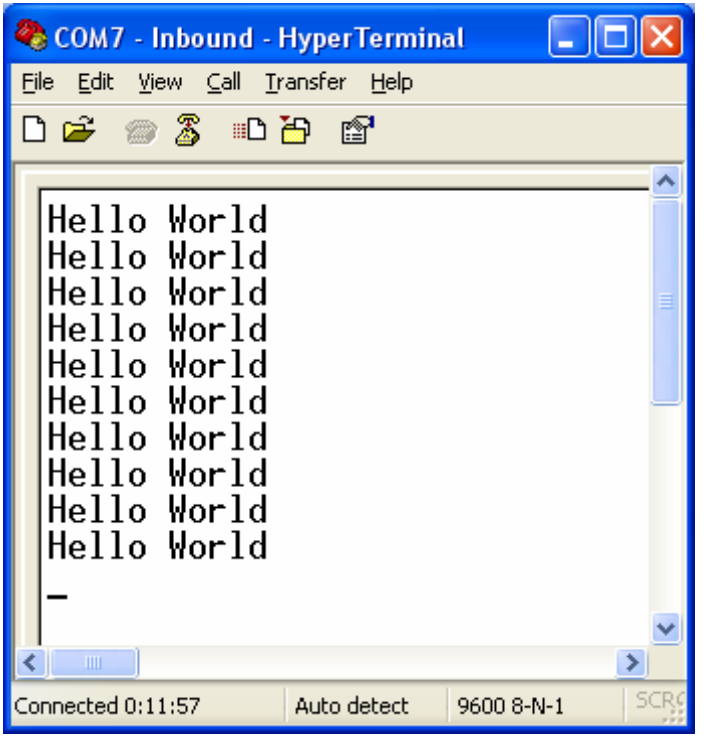


Figure 12: HyperTerminal Output - Hello World

Communicating between a PC with DBT-120 and a BOE

In this exercise we will step through the process of communicating between a PC that has a D-Link® DBT-120 Bluetooth USB Adapter and an eb500 module inserted into a Board of Education.

To perform this exercise, as documented, you will need a D-Link DBT-120, a Board of Education board, and an eb500 module. If you are using any of the other supported Parallax boards, you may need to make adjustments to this exercise.

On the PC, the DBT-120 Bluetooth Software associates a COM port for establishing a connection from the PC to a remote Bluetooth device and a separate COM port for connections that are established from a remote Bluetooth device to the PC. The Establishing a Connection section of this manual describes how to establish the connection between devices. This exercise demonstrates how to communicate data between the PC and a remote eb500.

The D-Link DBT-120 Bluetooth USB Adapter software must be fully installed prior to establishing a connection. The PC settings shown in this exercise are based upon the software provided with the D-Link DBT-120 Bluetooth USB Adapter.

Step 1: Transmit Data from the PC to the BASIC Stamp

In this step we will create a BASIC Stamp application to read data from the eb500 and display the data in the BASIC Stamp Editor Debug window. We will then download and run the application.

1. Connect the **Board of Education board serial port** to the **PC**.
2. Open the **BASIC Stamp Editor**.
3. Enter the following **program code** into the editor. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file Receive.bs2.

```
‘{$STAMP BS2}
bData VAR BYTE
‘Wait for the eb500 radio to be ready
PAUSE 1000
Main:
        SERIN 0,84,[STR bData\1]
        DEBUG STR bData\1
        GOTO Main
```

The application waits for an individual byte of data to arrive and then displays the byte in the debug window and then repeats this process.

4. On the **File** menu, click **Save As**.
5. In the **File name** box, enter a file name to which to save the program just created. For example, Receive.bs2.
6. Click **Save**.
7. Apply **power** to the **Board of Education board**.
8. On the **Run** menu, click **Run**.

   This will display the Download Progress dialog while downloading the program to the BASIC Stamp. After the download is complete, the Debug Terminal #1 dialog will be shown.
9. Establish a connection from the **PC** to the **Board of Education**.

   Please see the section titled Connecting a PC with a DBT-120 to a Board of Education on page 31 for information on establishing a connection.
10. Using HyperTerminal, type a series of **characters**.

    These characters will be transmitted over the wireless link, read by the BASIC Stamp application, and then displayed in the debug window.

Step 2: Transmit Data from the BASIC Stamp to the PC

In this step we will create a BASIC Stamp application to send data out the eb500 to the PC where we will use HyperTerminal to display the data received by the DBT-120 Bluetooth USB Adapter.

1. Reset the **eb500** attached to the **Board of Education board** to place the eb500 into command mode.

   To reset the eb500 attached to the Board of Education board, disconnect the power, wait a couple of seconds, and then reconnect the power. The Reset push button on the Board of Education board will NOT reset the eb500.
2. Close **HyperTerminal**.
3. Close the **BASIC Stamp Editor Debug** dialog.
4. Open **My Bluetooth Places** by double-clicking on the desktop icon.

   This will display the My Bluetooth Places dialog.

5. Click **View or modify configuration**.

   This will display the Bluetooth Configuration dialog.

6. Select the **Local Services** tab and note the **COM Port** for the Bluetooth Serial Port service. You may have to scroll to the right to see the COM Port column of the table.

   This COM port is the serial communications port that the DBT-120 Bluetooth software has associated for connections that are established from a remote Bluetooth device. This COM port can be used to communicate with the eb500 from applications, such as HyperTerminal, when connections are established from remote Bluetooth devices to the PC.

7. Select the **Hardware** tab and note the **Device Address** shown in the Device Properties section of the dialog.

   The device address will be used in the BASIC Stamp application developed in later actions.

8. On the **Bluetooth Configuration** dialog, click **Cancel**.

   This will close the Bluetooth Configuration dialog.

9. Close the **My Bluetooth Places** window.

10. Open **HyperTerminal**.

    This will display the Connection Description dialog.

11. In the **Name** box, type the name of your connection. For example, eb500.

12. Click **OK**.

    This will display the Connect To dialog.

13. In the **Connect using** dropdown, select the **serial port** to which the DBT-120 Bluetooth software associated with the connection from the eb500 on the Board of Education board to the PC.

    This is the COM port that we previously noted as being the COM port that is used to communicate with the eb500 when connections are established from remote Bluetooth devices to the PC.

14. Click **OK**.

    This will display the Properties dialog.

15. In the **Bits per second** dropdown, select **9600**.

16. In the **Data bits** dropdown, select **8**.

17. In the **Parity** dropdown, select **None**.

18. In the **Stop bits** dropdown, select **1**.
19. In the **Flow control** dropdown, select **None**.
20. Click **OK**.

    This will establish a connection with the DBT-120 Bluetooth USB Adapter software. This does NOT establish a connection with the remote Bluetooth device. In the next steps, we will write a BASIC Stamp application to run on the Board of Education Board which will connect the eb500 to the DBT-120 and provide the serial connection over Bluetooth.
21. On the **Call** menu, click **Disconnect**.

    This will disconnect the connection just established, so that we can modify the connection properties in the following actions.
22. On the **File** menu, click **Properties**.

    This will display the Properties dialog.
23. On the **Settings** tab, click **ASCII Setup**.

    This will display the ASCII Setup dialog.
24. Check the **Send line ends with line feeds** checkbox.
25. Check the **Echo typed characters locally** checkbox.
26. Check the **Append line feeds to incoming line ends** checkbox.
27. Check the **Wrap lines that exceed terminal width** checkbox.
28. Click **OK**.

    This will return to the Properties dialog.
29. Click **OK**.
30. On the **Call** menu, click **Call**.

    This will establish a connection with the DBT-120 Bluetooth USB Adapter Software.
31. Using the BASIC Stamp Editor, on the **File** menu, click **New**.

    This will create a new project window within the BASIC Stamp Editor.
32. Enter the following **program code** into the editor, replacing the device Bluetooth address with the device address obtained from the Hardware tab of the Device Properties section of the Bluetooth Configuration dialog on the PC. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file HelloWorld.bs2.

```
'{$STAMP BS2}
nCount VAR BYTE
'Wait for the eb500 radio to be ready
PAUSE 1000

'Connect to the remote device
SEROUT 1,84,[“con 00:0C:84:00:05:29”,CR]
SERIN 0,84,[WAIT(“ACK”,CR)]

'Wait for the connection to be established and switch into data mode.
'When switching into data mode, a 300ms timeout is required to give the
'module enough time to make the change.
WaitForConnection:
        IF in5 = 0 THEN WaitForConnection
HIGH 6
PAUSE 300

DEBUG "Connection established",CR

'Send "Hello World" ten times
FOR nCount = 1 to 10
        SEROUT 1,84,["Hello World",CR]
        PAUSE 1000
NEXT

'Switch to Command Mode
LOW 6
SERIN 0,84,[WAIT(CR,”>”)]

'Disconnect from the remote device
SEROUT 1,84,[“dis”,CR]
SERIN 0,84,[wait(CR,”>”)]

DEBUG “Disconnected”,CR
```

The BASIC Stamp application establishes a Bluetooth connection with the PC, transmits “Hello World” ten times, switches back to command mode, and then disconnects from the remote device.

The first call to SEROUT is used when the eb500 is in command mode and instructs the eb500 to establish a connection with the device specified. Once a connection is established the eb500 is in data mode, which causes further calls to SEROUT to be sent to the remote device.

33. On the **File** menu, click **Save As**.

    This will display the Save As dialog.

34. In the **File name** box, enter a file name to which to save the program just created. For example, HelloWorld.bs2.

35. Click **Save**.

36. On the **Run** menu, click **Run**.

    This will display the Download Program dialog while downloading the program to the BASIC Stamp. After the download is complete the BASIC Stamp application will transmit “Hello World” over the wireless link, and HyperTerminal will display the received data (Figure 12 on page 66).

Communicating between a PC with XP SP2 and a BOE

In this exercise we will step through the process of communicating between a PC that has a Bluetooth USB adapter that is running Windows XP SP2 and an eb500 module connected to a Board of Education.

To perform this exercise, as documented, you will need a PC running Windows XP SP2, a Bluetooth USB adapter, a Board of Education (BOE), and an eb500 module. If you are using any of the other supported Parallax boards, you may need to make adjustments to this exercise.

On the PC, the Microsoft Bluetooth Software associates a COM port for establishing a connection from the PC to a remote Bluetooth device and a separate COM port for connections that are established from a remote Bluetooth device to the PC. The Establishing a Connection section of this manual describes how to establish the connection between devices. This exercise demonstrates how to communicate data between the PC and a remote eb500.

Step 1: Transmit Data from the PC to the eb500

In this step we will create a BASIC Stamp application to read data from the eb500 and display the data in the BASIC Stamp Editor Debug window. We will then download and run the application.

1. Connect the **Board of Education board serial port** to the **PC**.
2. Open the **BASIC Stamp Editor**.
3. Enter the following **program code** into the editor. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file Receive.bs2.

```
‘{$STAMP BS2}
bData VAR BYTE
‘Wait for the eb500 radio to be ready
PAUSE 1000
Main:
        SERIN 0,84,[STR bData\1]
        DEBUG STR bData\1
        GOTO Main
```

The application waits for an individual byte of data to arrive and then displays the byte in the debug window and then repeats this process.

4. On the **File** menu, click **Save As**.
5. In the **File name** box, enter a file name to which to save the program just created. For example, Receive.bs2.
6. Click **Save**.
7. Apply **power** to the **Board of Education board**.
8. On the **Run** menu, click **Run**.

   This will display the Download Progress dialog while downloading the program to the BASIC Stamp. After the download is complete, the Debug Terminal #1 dialog will be shown.
9. Establish a connection from the **PC** to the **Board of Education**.

   Please see the section titled Connecting a PC with XP SP2 to a Board of Education for information on establishing a connection.
10. Using HyperTerminal, type a series of **characters**.

    These characters will be transmitted over the wireless link, read by the BASIC Stamp application, and then displayed in the debug window.

Step 2: Transmit Data from the eb500 to the PC

In this step we will create a BASIC Stamp application to transmit data from the eb500 to the PC where we will use HyperTerminal to display the data received.

1. Open **HyperTerminal** on the port for inbound Bluetooth connections.

   Please see the section titled Communicating between a PC with DBT-120 and a BOE for information on configuring HyperTerminal for the inbound Bluetooth port.
2. Using the BASIC Stamp Editor, on the **File** menu, click **New**.

   This will create a new project window within the BASIC Stamp Editor.
3. Enter the following **program code** into the editor, replacing the device Bluetooth address with the device address obtained from the Hardware tab of the Device Properties section of the Bluetooth Configuration dialog on the PC. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file HelloWorld.bs2.

```
'{$STAMP BS2}
nCount VAR BYTE
'Wait for the eb500 radio to be ready
PAUSE 1000
```

```
'Connect to the remote device
SEROUT 1,84,[“con 00:0C:84:00:05:29”,CR]
SERIN 0,84,[WAIT(“ACK”,CR)]

'Wait for the connection to be established and switch into data mode.
'When switching into data mode, a 300ms timeout is required to give the
'module enough time to make the change.
WaitForConnection:
        IF in5 = 0 THEN WaitForConnection
HIGH 6
PAUSE 300

DEBUG "Connection established",CR

'Send "Hello World" ten times
FOR nCount = 1 to 10
        SEROUT 1,84,["Hello World",CR]
        PAUSE 1000
NEXT

'Switch to Command Mode
LOW 6
SERIN 0,84,[WAIT(CR,”>”)]

'Disconnect from the remote device
SEROUT 1,84,[“dis”,CR]
SERIN 0,84,[wait(CR,”>”)]

DEBUG “Disconnected”,CR
```

The BASIC Stamp application establishes a Bluetooth connection with the PC, transmits “Hello World” ten times, switches back to command mode, and then disconnects from the remote device.

The first call to SEROUT is used when the eb500 is in command mode and instructs the eb500 to establish a connection with the device specified. Once a connection is

established the eb500 is in data mode, which causes further calls to SEROUT to be sent to the remote device.

4. On the **File** menu, click **Save As**.

   This will display the Save As dialog.

5. In the **File name** box, enter a file name to which to save the program just created. For example, HelloWorld.bs2.

6. Click **Save**.

7. On the **Run** menu, click **Run**.

   This will display the Download Program dialog while downloading the program to the BASIC Stamp. After the download is complete the BASIC Stamp application will transmit "Hello World" over the wireless link, and HyperTerminal will display the received data (Figure 12 on page 66).

Communicating between a Pocket PC 2003 and a BOE

In this exercise we will step through the process of communicating between a Pocket PC 2003 device with integrated Bluetooth, and an eb500 module inserted into a Board of Education.

To perform this exercise, as documented, you will need a Pocket PC 2003 device, a Board of Education, and an eb500 module. Depending on the specific Pocket PC model that you are using, you may need to make minor adjustments to this exercise.

Step 1: Transmit Data from the Pocket PC to the BASIC Stamp

In this step we will create a BASIC Stamp application to read data from the eb500 and display the data in the BASIC Stamp Editor Debug window. We will then download and run the application. The application for the Pocket PC is too verbose to include in this manual; therefore, the application, along with the source code, is available on the accompanying CD, in the Samples folder. To modify the Pocket PC application you will need eMbedded Visual C++ 4.0 with Service Pack 2 and the SDK for Windows Mobile™ 2003-based Pocket PCs.

1. Connect the **Board of Education serial port** to the **PC**.
2. Open the **BASIC Stamp Editor**.
3. Enter the following **program code** into the editor. The application is available in electronic form on the accompanying CD, in the Samples folder, in the file ReceivePPC.bs2.

```
‘{$STAMP BS2}
szData VAR BYTE(20)
‘Wait for the eb500 radio to be ready
PAUSE 1000
Main:
        SERIN 0,84,[STR szData\20\CR]
        DEBUG STR szData,CR
        GOTO Main
```

4. On the **File** menu, click **Save As**.
5. In the **File name** box, enter a file name to which to save the program just created. For example, ReceivePPC.bs2.
6. Click **Save**.

7. Apply **power** to the **Board of Education board**.
8. On the **Run** menu, click **Run**.

   This will display the Download Progress dialog while downloading the program to the BASIC Stamp. After the download is complete, the Debug Terminal #1 dialog will be shown.
9. On the Pocket PC, tap the **Bluetooth icon** in the system tray on the Today screen and select **Bluetooth Settings**.

   This will display the Settings dialog.
10. Scroll to the right, tap the **Serial Port** tab and note the **Outbound COM port**.

    The Outbound COM port will be used in the Pocket PC application later in this step.
11. Download the **PPCTxToEB** Pocket PC application to the **Pocket PC 2003 device**.
12. Run the **PPCTxToEB** Application (Figure 13).

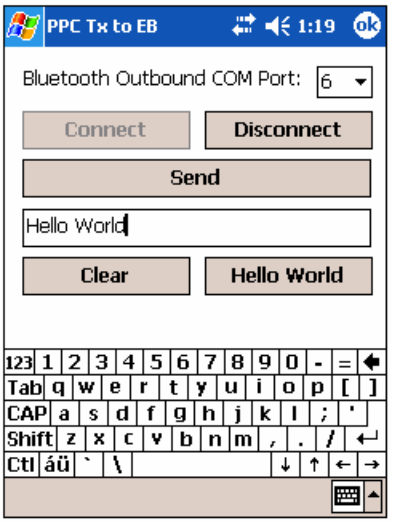


Figure 13: PPCTxToEB Pocket PC Application

13. In the **Bluetooth Outbound COM Port** dropdown, select the **COM port number** that matches the Bluetooth Outbound COM Port, which we previously discovered.
14. Tap the **Connect** button.

    This will display the Bluetooth Browser dialog (Figure 14).

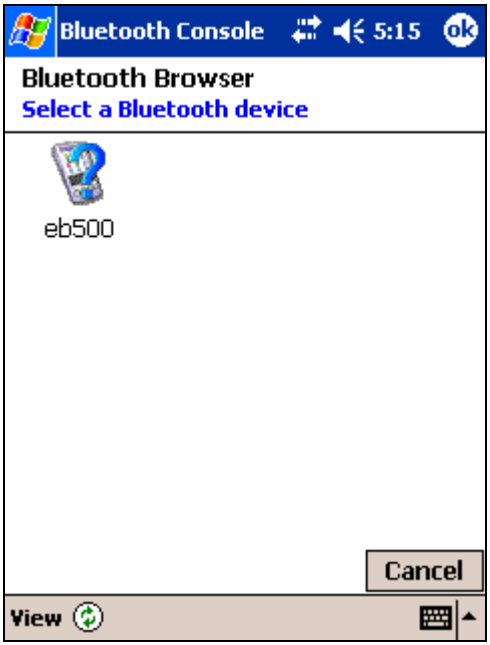


Figure 14: Pocket PC Bluetooth Browser Dialog

15. Tap **eb500** in the Bluetooth Browser dialog to establish a connection with the eb500 on the Board of Education.

    If there are no devices shown in the Bluetooth Browser dialog, tap the refresh icon to search for your Bluetooth device.

    A connection will be established with the device and the PPCTxToEB application will be shown again.

16. Enter some text and tap **Send**.

    This will transmit the ASCII text over the wireless link. The BASIC Stamp application will then receive these characters and display them in the BASIC Stamp Editor Debug window.

    You can also tap the **Hello World** button to have the application fill the edit box with the text “Hello World”.

17. Tap the **Disconnect** button to close the Bluetooth connection.

Step 2: Transmit Data from the BASIC Stamp to the Pocket PC

In this step we will create a BASIC Stamp application to send data out the eb500 to the Pocket PC. We will then download and run the application. The application for the Pocket PC is too verbose to include in this manual; therefore, the application, along with the source code, is available on the accompanying CD in the Samples folder. To modify the Pocket PC application you will need eMbedded Visual C++ 4.0 with Service Pack 2 and the SDK for Windows Mobile™ 2003-based Pocket PCs.

1. Using the BASIC Stamp Editor, on the **File** menu, click **New**.

   This will create a new project window within the BASIC Stamp Editor.

2. Enter the following **program code** into the editor, replacing the device address with the device address obtained from the Pocket PC. This application is available in electronic form on the accompanying CD, in the Samples folder, in the file HelloWorld.bs2.

```
'{$STAMP BS2}
nCount VAR BYTE
'Wait for the eb500 radio to be ready
PAUSE 1000

'Connect to the remote device
SEROUT 1,84,[“con 00:0C:84:00:05:29”,CR]
SERIN 0,84,[WAIT(“ACK”,CR)]

'Wait for the connection to be established and switch into data mode.
'When switching into data mode, a 300ms timeout is required to give the
'module enough time to make the change.
WaitForConnection:
        IF in5 = 0 THEN WaitForConnection
HIGH 6
PAUSE 300

DEBUG "Connection established",CR

'Send "Hello World" ten times
FOR nCount = 1 to 10
        SEROUT 1,84,["Hello World",CR]
```

```
        PAUSE 1000
NEXT

'Switch to Command Mode
LOW 6
SERIN 0,84,[WAIT(CR,”>”)]

'Disconnect from the remote device
SEROUT 1,84,[“dis”,CR]
SERIN 0,84,[wait(CR,”>”)]

DEBUG “Disconnected”,CR
```

The BASIC Stamp application establishes a connection with the Pocket PC, transmits “Hello World” ten times, switches back to command mode, and then disconnects from the remote device.

The first call to SEROUT is used when the eb500 is in command mode and instructs the 500 module to establish a connection with the device specified. Once a connection is established the eb500 is in data mode, which causes further calls to SEROUT to be sent to the remote device.

3. On the **File** menu, click **Save As**.
4. In the **File name** box, enter a file name to which to save the program just created. For Example, HelloWorld.bs2.
5. Click **Save**.
6. On the Pocket PC, tap the **Bluetooth icon** in the system tray on the Today screen and select **Bluetooth Settings**.

   This will display the Settings dialog.
7. Scroll to the right, tap the **Serial Port** tab and note the **Inbound COM port**.

   The Inbound COM port will be used in the Pocket PC application later in this step.
8. Download the **PPCRxFromEB** application to the **Pocket PC**.
9. Run the **PPCRxFromEB** application (Figure 15).

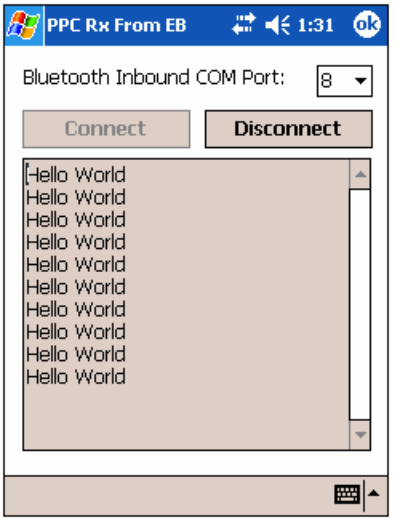


Figure 15: PPCRxFromEB Pocket PC Application

10. In the **Bluetooth Inbound COM Port** dropdown, select the **COM port number** that matches the Bluetooth Inbound COM Port, which we discovered in a previous action.
11. Tap the **Connect** button.
12. Apply **power** to the Board of Education board.
13. Using the BASIC Stamp Editor, on the **Run** menu, click **Run**.

    This will display the Download Progress dialog while downloading the program to the BASIC Stamp. After the download is complete the BASIC Stamp application will establish a connection with the Pocket PC. Depending on your current Pocket PC Bluetooth configuration, the Authorization Request Dialog may appear (see Figure 10 on page 53). If this dialog appears, tap Accept to accept the connection. Once the connection is established, the BASIC Stamp application will transmit “Hello World” over the wireless link, and the Pocket PC application will display the received data.
14. On the Pocket PC, tap the **Disconnect** button to close the connection.

Security

This section contains a number of exercises that demonstrate various security scenarios that can be implemented when using the eb500 module. The scenarios described are not meant to form an exhaustive list, but rather illustrate a number of more common and useful configurations. All source code shown in these exercises is available in electronic form on the accompanying CD, in the Samples folder, using the filename used in this manual.

Strong Security on a Board of Education

Cellular phones are typically configured for strong Bluetooth security because not only do they allow you to make phone calls, but they often contain sensitive contact information as well. In this exercise we will demonstrate how to use the eb500 module to implement the strong Bluetooth security model used on cellular phones.

To perform this exercise, as documented, you will need a PC running Windows XP SP2, a Bluetooth USB adapter, a Board of Education, two 470Ω resistors (yellow-violet-brown), one LED, and one eb500 module. If you are using any of the other supported Parallax boards, you may need to make adjustments to this exercise.

Step 1: Setup the circuit on the Board of Education

In this step we will assemble a circuit on the Board of Education breadboard that will visibly display the current security mode, connection status, and allow the security mode to be changed.

1. **Disconnect** the **power** from the Board of Education board.

   If you are using a Board of Education Rev C, you can simply set the 3-position switch to position-0. For all other boards you should remove the power source from the board.

2. **Assemble** the **circuit** shown in Figure 16 on the Board of Education breadboard.

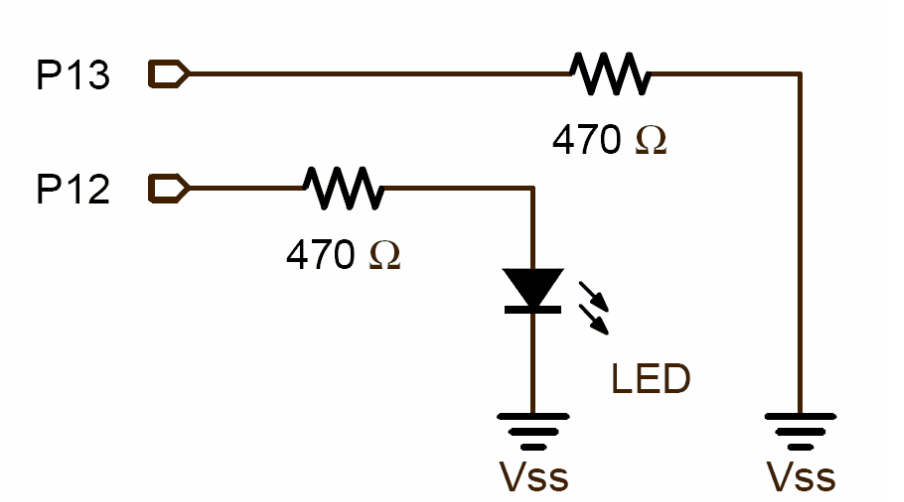


Figure 16: Closed Security Mode Circuit Diagram

Step 2: Configure Bluetooth security through a BASIC Stamp application

In this step we will write a BASIC Stamp application that allows you to change the current Bluetooth security settings by pulling I/O line P13 on the BASIC Stamp either high or low. We will then download and run the application.

Note about passkeys: Passkeys should be created with the same idea in mind as creating a password for your own personal use. Some tips include using at least eight characters, including a combination of uppercase letters, lowercase letters, symbols, and numbers. The more characters and different types of characters you use, the more secure your passkey will be. Additionally, it's recommended not to use words that can be found in the dictionary as this allows for a common dictionary attack to easily crack your passkey. This BASIC Stamp application changes the passkey of the eb500 module. You will need to make note of the passkey as it will be needed in the next step when we connect the PC to the eb500.

1. Connect the **Board of Education serial port** to the **PC**.
2. Open the **BASIC Stamp Editor**.
3. Enter the following **program code** into the editor. The application is available in electronic form on the accompanying CD, in the Samples folder, in the file Security.bs2.

```
'{$STAMP BS2}
'{$PBASIC 2.5}
nSecurityMode VAR Byte

'Wait for the eb500 radio to be ready
PAUSE 1000

'Use a secure passkey
SEROUT 1,84,["set passkey a7blue*500",CR]
SERIN 0,84,[WAIT(CR,">")]

'Use encryption
SEROUT 1,84,["set encrypt on",CR]
SERIN 0,84,[WAIT(CR,">")]

'Start with security closed
GOSUB SetSecurityClosed

DO
        'Check for changes in the security mode line. Only make
        'a change if not currently connected.
        IF nSecurityMode ^ IN13 THEN
                IF  IN5 = 0 THEN GOSUB ChangeSecurityMode
        ENDIF

        'If there is an active connection turn on the LED
        IF IN5 = 1 THEN
                HIGH 12
        ELSE
                IF (nSecurityMode = 1) THEN
                        'The module is in open mode so blink the LED
                        HIGH 12
                        PAUSE 500
                        LOW 12
                        PAUSE 500
```

```
                ELSE
                        'The module is in closed mode so turn off the LED
                        LOW 12
                ENDIF
        ENDIF
LOOP

SetSecurityClosed:
        'Set visible off, security closed
        SEROUT 1,84,["set visible off",CR]
        SERIN 0,84,[WAIT(CR,">")]
        SEROUT 1,84,["set security closed",CR]
        SERIN 0,84,[WAIT(CR,">")]

        'Record the new mode and return
        nSecurityMode = 0
        RETURN

SetSecurityOpen:
        'Set visible on, security open
        SEROUT 1,84,["set visible on",CR]
        SERIN 0,84,[WAIT(CR,">")]
        SEROUT 1,84,["set security open",CR]
        SERIN 0,84,[WAIT(CR,">")]

        'Record the new mode and return
        nSecurityMode = 1
        RETURN

ChangeSecurityMode:
        IF nSecurityMode = 0 THEN
                GOSUB SetSecurityOpen
        ELSE
                GOSUB SetSecurityClosed
        ENDIF
```

```
        RETURN
END
```

The application waits for the Stamp I/O line P13 to be pulled high as shown in Figure 17. Once line P13 is high, the eb500 module goes into a more vulnerable mode; it becomes visible and switches to open security enabling new trusted relationships. When in open security mode, the LED will blink. By pulling P13 low again as shown in Figure 16, the eb500 will return to closed security mode and visibility will be turned off.

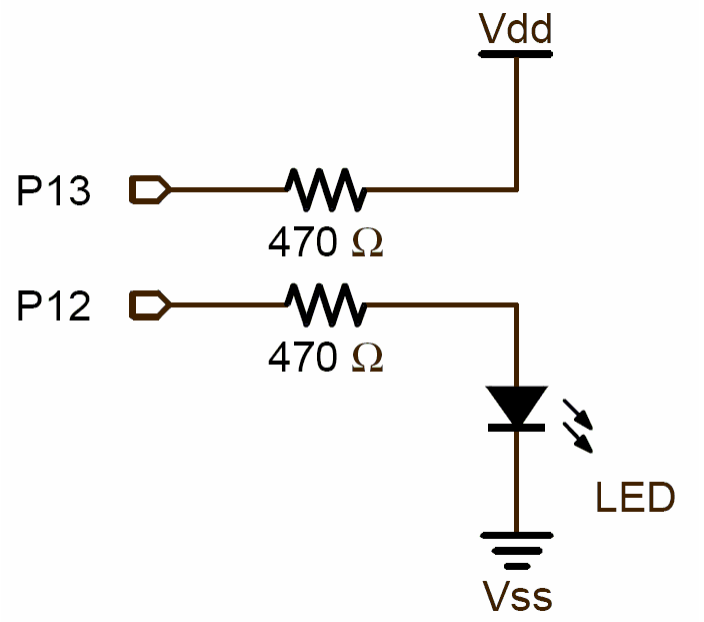


Figure 17: Open Security Mode Circuit Diagram

4. On the **File** menu, click **Save As**.
5. In the **File name** box, enter a file name to which to save the program just created. For example, Security.bs2.
6. Click **Save**.
7. Apply **power** to the **Board of Education board**.

8. On the **Run** menu, click **Run**.

   This will display the Download Progress dialog while downloading the program to the BASIC Stamp.

   A device that implements strong security is most vulnerable when allowing devices that have not yet been trusted to connect, verify their passkey, and become trusted. There is nothing inherently unsafe about this process; however it is a time when intruders have the most opportunity to compromise the device.

   There are two main concerns when security is set to open. The first is that if a weak passkey is set, an intruder may be able to guess the passkey and therefore gain access to the device. This can be largely avoided however, by first choosing a strong passkey and second by only placing the device into open security mode temporarily in a safe environment and then returning to closed security mode.

   The second concern is that an intruder pulls sensitive information out of the air that either is directly useful or allows trusted access to the device. This is only possible during a small window of time when an untrusted device is first connected and becomes trusted, so this process should be carried out in a safe environment if possible. Once a device is considered trusted, no information that could be used by an intruder to gain access to the device is transmitted over the air. If encryption is enabled as well, then no information including the data sent over the air is vulnerable to an intruder. By default EmbeddedBlue modules use 56-bit encryption for all data when enabled.

Step 3: Connecting a PC with XP SP2 to a BOE in open security

In this step, we will run the application we wrote in the previous section, which initially runs in closed security mode with visibility set to off. Stamp I/O line P13 can be pulled high to change into open security mode with visibility on. At this point, connect your PC with XP SP2 to the remote Bluetooth device and become "trusted." After returning to closed security mode with visibility turned off, only trusted devices (the PC) can connect to the remote Bluetooth device.

1. Disconnect the **serial cable** from the **Board of Education** board.
2. Recycle the power on the **Board of Education** by unplugging the **power cable** and plugging the cable in.
3. Pull Stamp I/O line **P13 high** on the **Board of Education** board by implementing the circuit shown in Figure 17.

   This puts the eb500 module in open security mode with visibility on. Note the blinking LED which informs us we are vulnerable to other devices becoming trusted with the eb500 if they have the correct passkey.

4. Connect your **PC** with XP SP2 to the remote **Bluetooth device**.

   See the section named Connecting a PC with XP SP2 to a Board of Education for more information. Upon successful connection, the PC and eb500 will establish a trusted relationship. You will need the passkey from the BASIC Stamp application written in the previous section. The application changed the passkey to a more secure passkey from the default of 0000 by using the command set passkey.

5. **Disconnect** the PC and the remote Bluetooth device.

6. **Pull** Stamp line **P13 low** to return to closed security with visibility off.

   Notice the security status LED on the breadboard will be off.

7. **Connect** to the **eb500** module from the PC with XP SP2, referring to the section named Connecting a PC with XP SP2 to a Board of Education for more information.

   The PC can now connect and disconnect with the remote Bluetooth device in a secure manner since visibility is now off and security is closed. Remember that when security is closed, no other devices are allowed to become "trusted."

   The LED on the Board of Education breadboard will mimic the connection LED on the eb500 module. When there is a connection between the PC and the remote Bluetooth device, the LED will be on. Likewise, it will be off when there is no connection.

   By pulling Stamp I/O line P13 high, the eb500 module will go back into open security mode with visibility set to on. At this point, other devices can become trusted.

   Note that when the eb500 is connected, changing the voltage on line P13 does NOT toggle between the security settings.

This page intentionally left blank.

Command Set

The EmbeddedBlue command set is comprised of visible ASCII characters. Therefore, a command can be issued from a terminal application, such as HyperTerminal, or directly from a custom application program, written in a programming language such as C++ or Visual Basic, running on a PC, using the eb600 PC adapter. From a BASIC Stamp application, these commands can be issued by using the PBASIC™ SERIN and SEROUT commands.

Command Basics

Commands may only be sent to the module when it is in Command Mode. White spaces are used to separate parameters of the command and a carriage-return is used to mark the end of the command. Upon receipt of a command the eb500 begins to parse the parameters. If the syntax of the command is correct the eb500 returns an ACK string, not the ACK character (0x06); otherwise, a NAK string is returned. Following the ACK or NAK string is a carriage-return (0x0D) character. If an error occurs while processing the command an error string is returned followed by a carriage-return followed by the prompt (>) character. If the command executed successfully the module will issue the prompt (>) character. Please see the Error Codes section for a description of the error codes.

The following example shows the basic structure of a command. A prompt (>) is issued by the EmbeddedBlue module. A command followed by a carriage-return is sent to the module. The module responds with either an ACK or NAK string followed by a carriage-return. If an error occurs, the module responds with an Err string followed by a space followed by an ASCII string numeric value followed by a carriage-return. A prompt (>) is then issued by the module.

```
>command<CR>
ACK | NAK<CR>
Err number<CR>
>
```

Command Error Handling in BASIC Stamp Applications

The BASIC Stamp has a software based UART; meaning it does not buffer incoming serial data. Therefore, the checking of errors from the issuing of an eb500 command must be performed immediately after the issuing of the command; otherwise, the data may be lost. Below is a sample of BASIC Stamp code that issues an eb500 Connect command, waits for the `ACK<CR>` response from the eb500, then waits for the error string or the prompt (`>`) to be returned from the eb500. It then checks the first bye of the data returned to determine if an error has occurred. If an error has occurred, the code jumps to the error handler code, where an error string along with the error number is shown in the debug window of the Basic Stamp Editor.

```
        'Connect to remote Bluetooth device
        SEROUT 1,84,["con 00:0C:84:00:07:D8",CR]
        SERIN 0,84,[WAIT(“ACK”,CR)]
        ‘Either an Err #<CR> or a ">" will be received
        SERIN 0,84,[STR bBuffer\6\”>”]
        IF bBuffer(0) = “E” THEN ErrorCode
        … Progam Logic …
ErrorCode:
        bErrorCode = bBuffer(4)
        DEBUG “Error: “,STR bErrorCode,CR
        END
```

Connect

The *connect* command establishes a connection to another Bluetooth device. The *connect* command may be canceled before a connection is established by issuing a carriage-return to the EmbeddedBlue device. It can take up to four seconds to cancel the connection request.

Syntax

con *address* [*timeout*]<CR>

Parameters

address — The Bluetooth address of the remote device. The Bluetooth device address is the 48-bit IEEE address which is unique for each Bluetooth unit. The format of a Bluetooth device address is a series of six hexadecimal byte values separated by colons, i.e., 00:0C:84:00:05:29.

timeout — An optional parameter used to abort the connection request after the specified number of seconds. The maximum value is 120 seconds.

Example

```
>con 00:0C:84:00:05:29<CR>
ACK<CR>
>
```

Delete Trusted Device

The *delete trusted device* command removes the remote device from trusted status and prevents it from being able to connect with the EmbeddedBlue device when security is set to closed. A delete can be performed for either a single device by passing its device address or for all trusted devices by specifying the keyword all.

Syntax

del trusted all | *address*<CR >

Parameters

all	This parameter is used to remove all devices from trusted status.
address	The Bluetooth address of the device that should be removed from trusted status. The Bluetooth device address is the 48-bit IEEE address which is unique for each Bluetooth unit. The format of a Bluetooth device address is a series of six hexadecimal byte values separated by colons, i.e., 00:0C:84:00:05:29.

Example

```
>del trusted 00:0C:84:00:05:29<CR>
ACK<CR>
>
```

Disconnect

The *disconnect* command closes the connection with the remote Bluetooth device.

Syntax

dis<CR>

Example

```
>dis<CR>
ACK<CR>
>
```

Get Address

The *get address* command returns the address of the local EmbeddedBlue device.

Syntax

get address<CR>

Returns

The unique address of the local EmbeddedBlue device used to identify the module when making connections. In Bluetooth terminology this is the Bluetooth Device Address.

Example

```
>get address<CR>
ACK<CR>
00:0C:84:00:05:29<CR>
>
```

Get Connectable Mode

The *get connectable mode* command returns the connectable mode setting of the local EmbeddedBlue device.

Syntax

get connectable<CR>

Returns

The current connectable mode setting of the local EmbeddedBlue device. In Bluetooth terminology, the returned value reflects the current setting for page scan.

on	The device will accept connections.
off	The device will NOT accept connections.

Example

```
>get connectable<CR>
ACK<CR>
on<CR>
>
```

Get Encrypt Mode

The *get encrypt mode* command returns the current encryption setting of the module. This setting controls whether the module encrypts transmitted data when security is set to either open or closed. By default EmbeddedBlue modules use 56-bit encryption, but 128-bit encryption is available upon request. Contact A7 Engineering for more information about getting 128-bit encryption.

Syntax

get encrypt<CR>

Returns

The current encrypt mode setting of the module.

on	If security is set to open or closed, transmitted data will be encrypted.
off	Transmitted data will NOT be encrypted.

Example

```
>get encrypt<CR>
ACK<CR>
on<CR>
>
```

Get Escape Character

The *get escape character* command returns the current character used in the Switch to Command Mode command to instruct the EmbeddedBlue device to leave Data Mode and enter Command Mode.

Syntax

get escchar<CR>

Example

```
>get escchar<CR>
ACK<CR>
+<CR>
>
```

Get Flow Control

The *get flow control* command returns the flow control setting of the local EmbeddedBlue device.

Syntax

get flow<CR>

Returns

none	The EmbeddedBlue device is configured for no flow control and both the RTS and CTS lines are configured as high Z inputs. An application is free to use these lines as normal I/O.
hardware	The EmbeddedBlue device is configured for hardware flow control and the RTS line is used for receive flow control and the CTS line is used for transmit flow control.

Example

```
>get flow<CR>
ACK<CR>
none<CR>
>
```

Get Link Timeout

The *get link timeout* command returns the amount of time, in seconds, it takes for the local EmbeddedBlue device to notice that the connection has been broken if the remote device disappears. This timeout also has an effect on how robust the communications link is to interference. If this value is set very low, the link may be lost if interference picks up for several seconds, such as when a heavy burst of 802.11 traffic is encountered.

Syntax

get linktimeout<CR>

Example

```
>get linktimeout<CR>
ACK<CR>
5<CR>
>
```

Get Name

The *get name* command returns the name of the local device. This is the value that is transmitted when a remote device performs an Inquiry and then requests the device name. If you look for local Bluetooth devices from a PC or PDA, this is the value that will be displayed to the user.

Syntax

get name<CR>

Example

```
>get name<CR>
ACK<CR>
eb500<CR>
>
```

Get Security Mode

The *get security mode* command returns the modules current security mode setting. When security is turned off, the module will allow connections to be established by any Bluetooth device. When security is set to open, the remote Bluetooth device is required to provide a valid passkey before a connection can be established. When security is set to closed, only existing trusted devices are allowed to establish connections.

For maximum security it is recommended that the module be operated in closed mode whenever possible.

Note: The Security mode is not applicable if connectable mode is set to off.

Syntax

get security<CR>

Returns

The current security mode setting of the module.

off	The module will allow any Bluetooth device to establish a connection.
open	The module will allow any Bluetooth device that provides the correct passkey to establish a connection.
closed	The module will only allow trusted devices to establish a connection.

Example

```
>get security<CR>
ACK<CR>
open<CR>
>
```

Get Visible Mode

The *get visible mode* command returns the modules current visibility setting. This setting controls whether the module can be seen by other Bluetooth devices.

Syntax

get visible<CR>

Returns

The current visible mode setting of the module. In Bluetooth terminology, the returned value reflects the current setting for inquiry scan.

on	The module is visible to other devices.
off	The module is NOT visible to other devices.

Example

```
>get visible<CR>
ACK<CR>
on<CR>
>
```

Help

The *help* command returns a listing of the EmbeddedBlue commands and a brief description of each command.

Syntax

hlp [*command*]<CR>

Parameters

command The EmbeddedBlue command name (con, del, dis, get, lst, rst, set, and ver) for which to return help.

Examples

```
>hlp<CR>
ACK<CR>
… Help Information …
<CR>
>

>hlp con<CR>
ACK<CR>
… Help Information on the Connect Command …
<CR>
>
```

List Trusted Devices

The *list trusted devices* command returns a list of all the devices that are allowed to connect when security is set to closed. The maximum number of devices that can be trusted at any given time is twenty five, so this command will return a list of between zero and twenty five addresses. When security is set to open, new devices can be added to this list by presenting the proper passkey while establishing a new connection.

Syntax

lst trusted<CR>

Returns

List of the trusted device addresses. These devices are the only ones that are allowed to connect with this module when security is set to closed.

Example

```
>lst trusted<CR>
ACK<CR>
00:0C:84:00:05:29<CR>
00:80:C8:35:2C:B8<CR>
>
```

List Visible Devices

The *list visible devices* command returns a listing of all the devices that are currently in range and visible. The command may be canceled before the timeout is reached by sending an additional carriage-return to the module.

Syntax

lst visible [*timeout*]<CR>

Parameters

timeout — An optional parameter used to abort the list request after the specified number of seconds. The default value is 30. The maximum value is 120 seconds.

Returns

The addresses of the Bluetooth devices that are in range and visible.

Example

```
>lst visible<CR>
ACK<CR>
00:0C:84:00:05:29<CR>
00:80:C8:35:2C:B8<CR>
>
```

Reset Factory Defaults

The *reset factory defaults* command restores all module settings to factory defaults. This includes the baud rate parameter which may cause serial communications to be lost after the command is issued. To reestablish communications with the module, simply adjust the baud rate on the microprocessor serial port to match the module default rate of 9600bps.

Syntax

rst factory<CR >

Parameters

factory Resets all settings to factory defaults.

Example

```
>rst factory<CR>
ACK<CR>
>
```

Return to Data Mode

The *return to data mode* command instructs the module to enter Data Mode when there is an active connection.

Syntax

ret<CR>

Example

```
>ret<CR>
ACK<CR>
>
```

Set Baud Rate

The *set baud rate* command sets the baud rate for communications with the local EmbeddedBlue module.

Syntax

set baud *rate* [*]<CR>

Parameters

rate	The baud rate value. Valid baud rates are 9600 (default), 19200, 38400, 57600, 115200, and 230400. Once the baud rate has been set, applications, such as HyperTerminal, must also be configured to the same baud rate to continue communicating with the eb500.
*	An optional parameter used to persist the new setting when the module is powered down.

Example

```
>set baud 19200<CR>
ACK<CR>
>
```

Set Connectable Mode

The *set connectable mode* command provides control over whether the local EmbeddedBlue module will accept connections from other Bluetooth devices. In Bluetooth terminology, this command controls the setting for page scan.

Syntax

set connectable on | off [*]<CR>

Parameters

on	Configures the module so that other Bluetooth devices may establish a connection.
off	Configures the module so that other Bluetooth devices may not establish a connection.
*	An optional parameter used to persist the new setting when the module is powered down.

Example

```
>set connectable off<CR>
ACK<CR>
>
```

Set Encrypt Mode

The *set encrypt mode* command provides control over whether transmitted data is encrypted or sent in the clear. This setting is only in effect when security is set to either open or closed. When security is turned off, the transmitted data is never encrypted. By default EmbeddedBlue modules use 56-bit encryption, but 128-bit encryption is available upon request. Contact A7 Engineering for more information about getting 128-bit encryption.

Syntax

set encrypt on | off [*]<CR>

Parameters

on	Configures the module so that transmitted data will be encrypted when security is set to either open or closed.
off	Configures the module so that transmitted data will NOT be encrypted.
*	An optional parameter used to persist the new setting when the module is powered down.

Example

```
>set encrypt on<CR>
ACK<CR>
>
```

Set Escape Character

The *set escape character* command provides control over the character used in the Switch to Command Mode command to instruct the module to leave Data Mode and enter Command Mode. The factory default escape character is the plus sign (+).

Syntax

set escchar *character* [*]<CR>

Parameters

character	The character the module should recognize as the escape character used in the Switch to Command Mode command.
*	An optional parameter used to persist the new setting when the module is powered down.

Example

```
>set escchar & *<CR>
ACK<CR>
>
```

Set Flow Control

The *set flow control* command provides control over the flow control setting of the local EmbeddedBlue module.

Syntax

set flow none | hardware [*] <CR>

Parameters

none	Configures the module for no flow control and both the RTS and CTS lines are configured as high Z inputs. This allows an application to use these lines a normal I/O.
hardware	Configures the module for hardware flow control. The RTS line is used for receive flow control and the CTS line is used for transmit flow control.
*	An optional parameter used to persist the new setting when the module is powered down.

Example

```
>set flow none<CR>
ACK<CR>
>
```

Set Link Timeout

The *set link timeout* command sets the amount of time it takes for the local EmbeddedBlue module to notice that the connection has been broken, if the remote device disappears. This timeout also has an effect on how robust the communications link is to interference. If this value is set very low, the link may be lost if interference picks up for several seconds, such as when a heavy burst of 802.11 traffic is encountered. In Bluetooth terminology, this command controls the setting for link supervisor timeout.

Syntax

set linktimeout *timeout* [*]<CR>

Parameters

timeout	The time, in seconds, it takes for the module to notice that a connection has been broken. The default value is 5. The maximum value is 40 seconds.
*	An optional parameter used to persist the new setting when the module is powered down.

Example

```
>set linktimeout 10<CR>
ACK<CR>
>
```

Set Name

The *set name* command sets the name of the local device. This is the value that is transmitted when a remote device performs an Inquiry and then requests the device name. If you look for local Bluetooth devices from a PC or PDA, this is the value that will be displayed to the user.

Syntax

set name *value* [*]<CR>

Parameters

value	A new device name. This value can be up to 32 characters in length and may contain any valid ASCII character.
*	An optional parameter used to persist the new setting when the module is powered down.

Example

```
>set name eb500<CR>
ACK<CR>
>
```

Set Passkey

The *set passkey* command sets the passkey that is used when establishing a connection with security set to open. The passkey is set to 0000 by default, but this value should be changed to enhance security. It is recommended that you use a passkey that is 8 to 16 digits long.

Syntax

set passkey *value* [*]<CR>

Parameters

value	A new passkey value that is between 1 and 16 digits long.
*	An optional parameter used to persist the new setting when the module is powered down.

Example

```
>set passkey MyNewKey<CR>
ACK<CR>
>
```

Set Security Mode

The *set security mode* command sets the module's current security mode setting. When security is turned off, the module will allow connections to be established by any Bluetooth device. When security is set to open, the remote Bluetooth device is required to provide a valid passkey before a connection can be established. When security is set to closed, only existing trusted devices are allowed to establish connections.

For maximum security it is recommended that the module be set to closed mode whenever possible.

Syntax

set security off | open | closed [*]<CR>

Parameters

off	Turns security off allowing any Bluetooth device to establish a connection.
open	Configures the module to require other devices to provide the correct passkey before establishing a connection.
closed	Configures the module to only allow trusted devices to establish a connection.
*	An optional parameter used to persist the new setting when the module is powered down.

Example

```
>set security open<CR>
ACK<CR>
>
```

Set Visible Mode

The *set visible mode* command provides control over whether the module can be seen by other Bluetooth devices. In Bluetooth terminology, this command controls the setting for inquiry scan.

Syntax

set visible on | off [*]<CR>

Parameters

on	Configures the module so that other Bluetooth devices can detect its presence.
off	Configures the module so that other Bluetooth devices can NOT detect its presence.
*	An optional parameter used to persist the new setting when the module is powered down.

Example

```
>set visible on<CR>
ACK<CR>
>
```

Switch to Command Mode

The *switch to command mode* command instructs the EmbeddedBlue module to enter Command Mode.

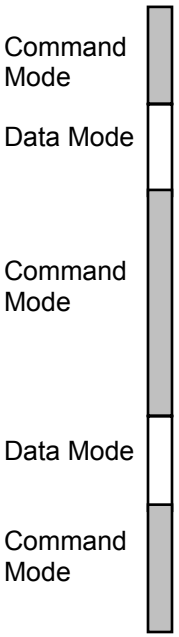
Syntax

<2 second pause>*esc sequence*<2 second pause>

Parameters

esc sequence Three consecutive instances of the escape character. The factory default escape character is the plus sign (+). A different escape character can be set by using the Set Escape Character command.

Example




```
>con 00:0C:84:00:07:D7<CR>
ACK<CR>
>This text is sent in data mode<CR>
<2 second pause>+++<2 second pause><CR>
>get addr<CR>
ACK<CR>
00:0C:84:00:05:29<CR>
>ret<CR>
ACK<CR>
>This text is sent in data mode<CR>
<2 second pause>+++<2 second pause><CR>
>dis<CR>
ACK<CR>
>
```

Version

The *version* command returns the current firmware version of the EmbeddedBlue module.

Syntax

ver [all] <CR>

Parameters

all	An optional parameter used to return the build number, model number, serial number, and manufacturer.

Example

```
>ver all<CR>
ACK<CR>
Firmware Version: 2.0<CR>
Firmware Build: 247<CR>
Model Number: eb500<CR>
Serial Number: 1008<CR>
Manufacturer: A7 Engineering<CR>
>
```

This page intentionally left blank.

Firmware Upgrade

From time-to-time A7 Engineering provides new versions of firmware that provide enhancements to the product. A7 Engineering also provides an EmbeddedBlue DFU utility which provides the mechanism to upgrade the firmware in the eb500 module. The latest version of the EmbeddedBlue DFU utility and firmware for the eb500 module may be obtained from the A7 Engineering web site at www.a7eng.com.

Upgrading the eb500 Firmware

This procedure will step you through the process of upgrading the firmware of the eb500 module. To upgrade the firmware of your eb500 you will need an eb600, a PC running Windows XP with Service Pack 2, the EmbeddedBlue DFU utility and the firmware upgrade file (a .DFU file obtained from the A7 Engineering web site).

Step 1: Install the EmbeddedBlue DFU Utility

In this step we will install the EmbeddedBlue DFU utility onto your PC. To execute this step you must have previously downloaded the EmbeddedBlue DFU utility from the A7 Engineering web site. If you have previously installed the EmbeddedBlue DFU utility you may proceed to the next step.

1. Navigate to the **folder** where you downloaded the EmbeddedBlue DFU utility and double-click on the **EmbeddedBlueDFU.exe**.
2. Step through the **install wizard** supplying the requested information on each of the dialogs of the wizard.

   It is recommended that you use the default settings of the install wizard.

Step 2: eb600 RS232 Adapter Setup

In this step we will attach an eb500 module to the eb600 PC Adapter and apply power to the device.

1. Insert an **eb500** module into the **eb600 RS232 Adapter** header; assuring that Pin 1 of the eb500 module is inserted into Pin 1 of the header on the eb600 RS232 Adapter.
2. Connect the **eb600 RS232 Adapter** to a serial port on the PC using the **provided straight through serial cable**.
3. Apply **power** to the **eb600 RS232 Adapter**.

Step 3: eb500 Module Setup

In this step we will configure the eb500 module to communicate with the EmbeddedBlue DFU Wizard. If your eb500 module is already configured to communicate at 9600 baud you may proceed to the next step.

1. Using a **terminal emulator**, such as HyperTerminal, establish a **connection** with the **eb500** attached to the eb600 RS232 Adapter.

   If you need assistance in the setup of HyperTerminal to communicate with your eb500 module, please refer to the *Establishing a Connection* section of this manual under the topic *Connecting a PC with an eb600 to a Board of Education*, *Step 2: HyperTerminal Setup*.

2. Set the **baud rate** of the **eb500** module to **9600** baud by issuing the set baud rate command.

   To set the baud rate, type `set baud 9600` at the "`>`" prompt and press the return key.

   Example:

   ```
   >set baud 9600
   ACK
   >
   ```

3. Close the **terminal emulator**.

Step 4: Run the EmbeddedBlue DFU Wizard

In this step we will step through the EmbeddedBlue DFU Wizard and upgrade the firmware of the eb500 module.

1. Launch the **EmbeddedBlue DFU** Wizard.

   From the Start menu select All Programs then A7 Engineering then EmbeddedBlue DFU.

2. The first page of the wizard is the introduction page; click **Next** to continue.
3. On the **Connection Type** dialog, select **COM port (RS-232)**.
4. Click **Next**.

   The wizard will now search for available COM ports.

5. Select the **COM port** to which the **eb600 RS232 Adapter** is connected.
6. Click **Next**.
7. On the **Upgrade File** dialog, click **Browse** to navigate to the file containing the firmware upgrade.
8. On the **Select EmbeddedBlue Firmware File** dialog, select the **file containing the firmware upgrade** (a .DFU file).
9. Click **Select**.
10. On the **Upgrade File** dialog, click **Next**.

    This will display the Ready to Upgrade dialog.

11. Review the **information** shown on the **Ready to Upgrade** dialog.
12. Click **Next**.

    This will display the Upgrade in Progress dialog and the upgrade process will begin. When the upgrade is complete the Successful Upgrade dialog will appear.

> ⚠ **You must not stop the upgrade process or remove power from the eb500 module until the upgrade is complete. If the upgrade process is interrupted the eb500 module may become non-functional.**

13. Review the **information** shown on the **Successful Upgrade** dialog.

    The firmware of your eb500 module has been upgraded.

    If the **Upgrade Failed** dialog appears click **Details…** to get additional information about the failure. Ensure that the eb500 is correctly inserted into the eb600 RS232 Adapter, that the eb600 RS232 Adapter is connected to the PC using the provided straight through serial cable and that you have selected the appropriate COM port in the EmbeddedBlue DFU wizard.

14. Click **Finish**.

Step 5: Check the Firmware Version

While it is not necessary to check the firmware version to complete the firmware upgrade process, this step demonstrates how to check the version of firmware on your EmbeddedBlue module.

1. Using a **terminal emulator**, such as HyperTerminal, establish a **connection** with the **eb500** attached to the eb600 RS232 Adapter.

   If you need assistance in the setup of HyperTerminal to communicate with your eb500 module, please refer to the *Establishing a Connection* section of this manual under the topic *Connecting a PC with an eb600 to a Board of Education*, *Step 2: HyperTerminal Setup*.

2. Check the **version** information by issuing the version command.

   To view the version information, type `ver all` at the "`>`" prompt and press the return key.

   Example:

```
>ver all
ACK
Firmware Version: 2.0
Firmware Build: 247
Model Number: eb500
Serial Number: 1008
Manufacturer: A7 Engineering
>
```

3. Close the **terminal emulator**.

Error Codes

While using the eb500 you may encounter an error. Below is a listing of all eb500 error codes with a description of what causes the error to occur.

Error Code	Description
1	General connection failure.
2	Connection attempt failed. This error occurs when attempting to connect with an invalid Bluetooth address or a device that is not available.
3	Command not valid while active. This error occurs when there is an active connection and a command is issued that is not valid while connected with a remote device.
4	Command only valid while active. This error occurs when there is not an active connection and a command is issued that is only valid while connected with a remote device.
5	An unexpected request occurred. This error occurs when the remote device makes an invalid request. This is typically seen with older Bluetooth devices that may have errors in their firmware.
6	Connection attempt failed due to a timeout.
7	Connection attempt was refused by the remote device. This error typically occurs when the security settings of the remote and local device are incompatible. It can also occur when establishing a connection with security set to open if the remote and local passkeys do not match.
8	Connection attempt failed because the remote device does not support the Serial Port Profile.
9	An unexpected error occurred when deleting trusted devices.
10	Unable to add a new trusted device. This error will occur if you attempt to have more than twenty five simultaneously trusted devices.
11	Trusted device not found. This error occurs when the trusted device address is not recognized.
12	Command not valid during startup. This error occurs when a command has been issued before the EmbeddedBlue module is fully powered up and initialized.

Table 1: eb500 Error Codes

This page intentionally left blank.

Technical Specifications

Operating Parameters

The operating parameters of the eb500 are shown below in Table 2.

Transmit Power	4dBm (max) class 2 operation	
Open Field Range	eb500-AHC-IN (surface mount antenna) – 100 meters (328 feet) *(Actual range is dependent upon location and environment.)*	
Receiver Sensitivity	-85dBm	
Operating Temp.	-15° to 70°C	
Supply Power	5 to 12VDC	
Current Consumption	115.2kbps data transfer: 35mA 38.4kbps data transfer: 30mA 9.6kbps data transfer: 25mA	connected and idle: 8mA no connection: 3mA
Interfaces	5V logic level UART or RS232 with optional eb600 adapter Baud rate 9.6k – 230.4k Flow control: RTS/CTS or none	
Connector	One 10x2 AppMod compatible 20 pin 0.1” header	
Antenna	Matched internal surface mount	
Bluetooth Support	Version 1.2 compliant with profiles L2CAP, RFCOMM, SDP, SPP	
Firmware	Upgradeable via PC application with eb600 adapter	

Table 2: eb500 Operating Parameters

Dimensions

The dimensions of the eb500 are shown below in Table 3. Please reference Figure 18 to locate the referenced dimension on the eb500.

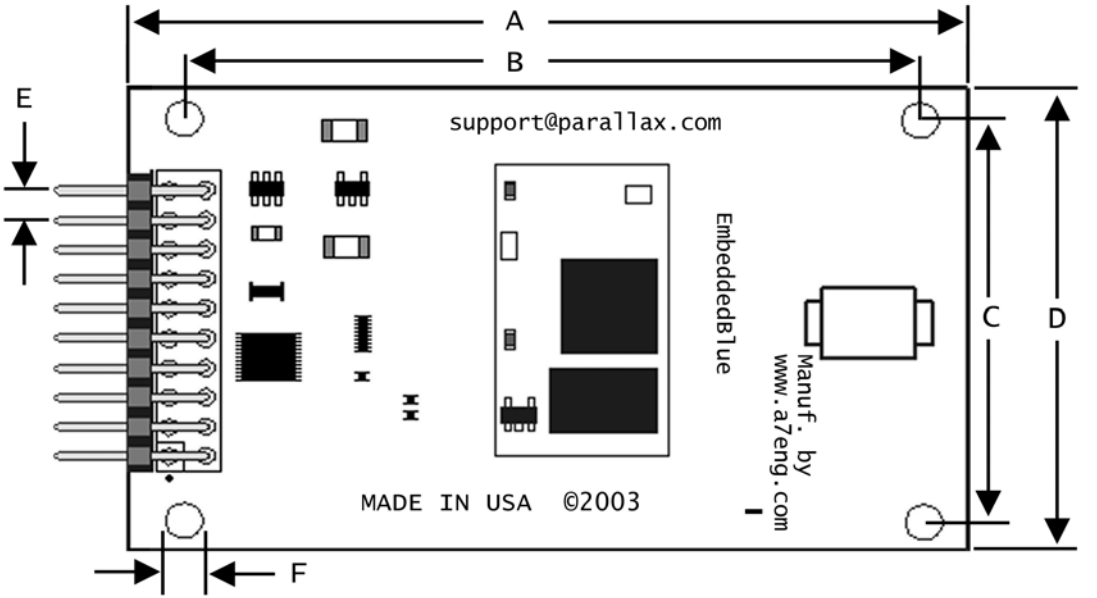


Figure 18: eb500 Dimensions

Dimension	inches	mm
A	2.75	69.85
B	2.40	60.96
C	1.30	33.02
D	1.60	40.64
E	0.10	2.54
F	0.125	3.20

Table 3: eb500 Dimensions

Pin out

The eb500 module features a 20 pin connector with 0.1" spacing for direct connection to a Parallax AppMod header. Currently, nine of the pins are in use (seven when flow control is set to none). The other pins are reserved for future use.

Pin	Parallax Pin	Function	Description	Usage
CN1 - 1	GND	GND	Ground	Required
CN1 - 2	GND	GND	Ground	Required
CN1 - 3	P0	TX	Serial Transmit line from eb500	Required
CN1 - 4	P1	RX	Serial Receive line to eb500	Required
CN1 - 5	P2	RTS	Request-to-Send on the serial port interface between the eb500 and the BASIC Stamp	Optional
CN1 - 6	P3	CTS	Clear-to-Send on the serial port interface between the eb500 and the BASIC Stamp	Optional
CN1 - 8	P5	Status	Bluetooth connection status (0 = not connected, 1 = connected)	Required
CN1 – 9	P6	Mode	Command/data mode toggle (0 = command, 1 = data)	Required
CN1 - 20	VCC	VCC	Power	Required

Table 4: eb500 Pin out Description

This page intentionally left blank.

Frequently Asked Questions

Question: How do I obtain eMbedded Visual C++ 4.0 to develop Pocket PC applications?

Answer: The eMbedded Visual C++ 4.0 development tool is available from Microsoft. In addition, you will need eMbedded Visual C++ 4.0 SP2 and the SDK for Windows Mobile™ 2003-based Pocket PCs. These tools can be downloaded free of charge from the Microsoft Windows Mobile web site: http://www.microsoft.com/windowsmobile.

Question: Why is my eb500 not displayed when I try to discover it from my PC or Pocket PC?

Answer: Verify that the eb500 module is properly powered. It is likely you will discover the eb500 on the first attempt; however, because Bluetooth discovery is not deterministic, discovery on the first attempt is not guaranteed. On the PC or Pocket PC, use the refresh option to search for devices again. Verify that the visible mode setting in the eb500 is set to on.

Question: I can discover my eb500, but why am I unable to establish a connection?

Answer: Verify that the connectable mode setting in the eb500 is set to on and that security is set either to off or open. In closed security mode only devices that have already established a trusted relationship will be allowed to connect.

Question: When I try to connect from an EmbeddedBlue device with 1.0 firmware to one with 2.0 firmware the connection attempt times out and then fails with Error 2. Why?

Answer: Version 1.0 firmware did not support passkey security and trusted relationships, which is enabled as the default in version 2.0 firmware. To connect from a version 1.0 device you will need to disable security on the version 2.0 device with the “set security off” command.

Question: I am transmitting large packets of data between two Parallax BASIC Stamp Modules using two eb500’s. Now and then I notice that some data seems to be lost. What is going on?

Answer: Bluetooth is a reliable point to point protocol much like TCP\IP. If transmitted data is lost or corrupted over the air it will automatically and seamlessly be retransmitted. As long as the eb500 status line tells you that there is a valid connection, you can be confident that all data will be delivered properly.

The most likely cause of this data loss involves the way that serial data is handled in the BASIC Stamp application. The BASIC Stamp devices implement a UART in software and therefore will miss data that arrives while not executing a SERIN command. Refer to the BASIC Stamp User's Guide for more details.

Question: I used the *set visible* command to make the eb500 module not visible to other devices, but when I perform a scan from my PC I still see the device. Why?

Answer: Most of the PC Bluetooth implementations cache device scan results to save time. If you located the eb500 module before making it invisible, the PC will remember the device even though it can no longer be seen. These results are typically only cached until the Bluetooth stack is reset, so if you reboot the PC or remove and reinsert the dongle you should no longer see the device.

Contact Information

Parallax provides technical support through email, an online group, and by telephone. It is recommended that you use email as the first line of questioning because common questions can be answered quickly and in greater detail in this manner.

Website: www.parallax.com

Support Email: support@parallax.com

Online Group: http://groups.yahoo.com/group/basicstamps/

Sales Email: sales@parallax.com

Parallax, Inc.

599 Menlo Drive, Suite 100
Rocklin, CA 95765
888.512.1024 main
916.624.8003 fax

A7 Engineering has created the EmbeddedBlue product line of easy to use wireless solutions for 8 and 16 bit embedded systems. In addition, A7 provides several levels of support for OEM product integration, certification, and even custom solutions.

Website: www.a7eng.com

Online Forum: http://www.a7eng.com/support/forum/forum.htm

Sales Email: sales@a7eng.com

A7 Engineering, Inc.

12860 C Danielson Court
Poway, CA 92064
858.679.7708 main
858.391.5616 fax