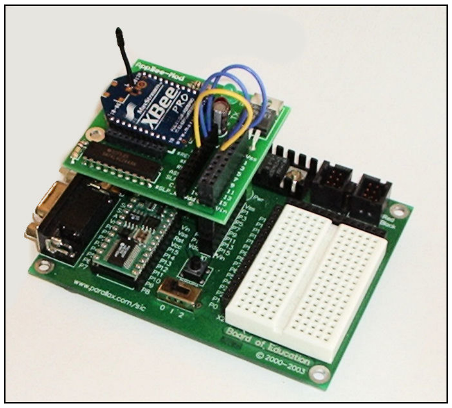
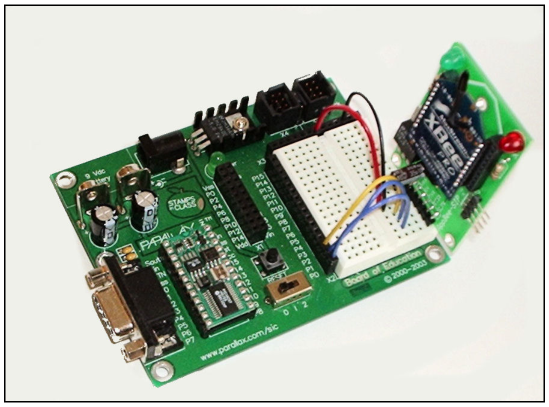
AppBee-Mod & AppBee-SIP

Documentation for use with the BASIC Stamp®

Application Interface Boards for the
ZigBee / IEEE 802.15.4 XBee® and XBee-PRO®
Wireless Network Transceivers

Document Revision 4.
Device Version AppBee-MOD Rev 0
AppBee-SIP Rev A
May 22, 2007

http://www.selmaware.com/appbee

Table of Contents

Revisions

October 9, 2006
The AppBee-SIP was redesigned for connecting to a PC using Parallax's Prop Plug as that may be the more support product. Due to having a large quantity of AppBee-Mods on hand, that board will still use the USB2SER. Connections on the USB2SER and Prop Plug are different! We recommend the USB Base Station for PC USB communications.

We now have available a USB base station for the XBee which may be used for PC interfacing and firmware programming.

Small modifications were made to this document for layout changes and corrections.

November 5, 2006
A simple TX/RX samples was added.
Warnings about board voltage regulator's heat was added.

May 22, 2007
Add CR to basic examples and added new example using flow control:
SEROUT TX, 84,[DEC X, CR]
Put RX HIGH on simple receive example to turn off LED.
Correct MY and DL for example.

Overview

The XBee is a wireless transceiver (modem) using the IEEE 802.15.4 Low-Rate Wireless Personal Area Network protocol (LR-WPAN) for Wireless Sensor Networks (WSN). This allows addressable communications between nodes. Data may be sent to individual nodes (point-to-point), or to all nodes in range (point-to-multipoint) using a broadcast address. The devices use clear channel assessment (CCA) on a CSMA/CA network which helps ensure devices do not talk-over one another. In point-to-point communications, error checking, acknowledgements and retries are used to ensure data delivery. The use of flow control (RTS) helps ensure devices, such as the BASIC Stamp, do not miss incoming data.

AppBee-Mod is an application interface board that connects to the AppMod dual row 20-pin header of many Parallax BASIC Stamp boards for power and quick communications to the BASIC Stamp. This board has connections to many I/O features of the XBee to provide greater flexibility.

AppBee-SIP is an application interface board with a 6-pin single row header for easy connection to breadboard and custom printed circuit boards. This board has only essential connection for power, communications and accessing sleep modes of the XBee, though additional jumpers and circuitry may be used to expand on the provided I/O.

Both boards provide a carrier for the XBee modules, 3.3V regulator, and 5V to 3.3V logic converters for safe interfacing. Both boards also support direct interfacing to a computer terminal program using Parallax's **USB2SER** USB to serial interface device (AppBee-MOD) or **Prop Plug** (AppBee-SIP). *All features of Maxstream's X-CTU software may be used with the USB2SER except reprogramming of the firmware.*

The **USB Base Station**, from Surveyor Corp and available though us, may be used for a USB PC interface for configuration, terminal use and reprogramming.

This document provides guidance on the use of the XBee with the BASIC Stamp but relies on the user to access the full XBee documentation to use the full features of this exciting device. Many other controllers may also use these interfaces boards, but this document will focus on BASIC Stamp use.

The following is summary of key XBee features from Maxstream's documentation for firmware version 1.083.

XBee Features

	XBee	**XBee-PRO**
* Range - Indoor	100 ft (30m)	300 ft (100m)
* Range – Outdoor	300 ft (100m)	1 mile (1500m)
Transmit Power	0 dBm (1mW)	20 dBm (100mW)
Receiver Sensitivity	-92 dBm	-100 dBm
TX Current	45mA	214mA
RX Current	50mA	55mA
Power-Down (Sleep) Current	<10uA	<10uA

* Ranges are line of sight and height dependent.

- IEEE 802.15.4 compliant, Low-Rate Personal Area Networking.
- 2.4 GHz DSSS (Direct Sequence Spread Spectrum).
- 250,000 bits per second.
- Acknowledgement and reties.
- Addressable, > 65,000 addresses available.
- Point-to-Point and Point-to-Multipoint (broadcast) messaging.
- Channel and Network ID selectable for cluster separation.
- Fully configurable via serial commands.
- Transparent transmission and reception.
- Receiver Strength indication.
- Free X-CTU interface software.
- Free and unlimited technical from MaxStream.

Firmware version 802.15.4 v1.083 is the recommended and discussed version for this document. Examples are not guaranteed to work with other firmware versions of the XBee. Please use the XBee documentation from MaxStream for full discussions on device use.

AppBee Features

- On-Board 3.3V regulator and 5V conditioning.
- Supplied power range from 5V to 12V
- Send and Receive LEDs
- Communications from serial devices (BASIC Stamp, etc), or from a PC using Parallax's USB2SER USB to serial converter for direct PC data acquisition and control or using MaxStreams X-CTU software.

NOTE: Depending on supply voltage and whether the lower power XBee or higher power XBee Pro is used, the 3.3V on the AppBee's can become very hot. Care should be taken. The AppBee-SIP can be supplied power from 5V. The AppBee-MOD is powered from Vin of the Parallax Board. If the regulator becomes excessively hot, a heat sink is recommended.

AppBee-MOD to XBee Interfacing Header

- 10 Pin header for jumpers to P0-P15
- TX, RX
- XBee Reset
- RTS and CTS for flow control
- Receiver Signal Strength Indicator output (RSSI)
- Association Indicator I/O output (ASSOC)
- Sleep Indicator I/O output (SLP_I)
- Sleep Request Input (SLP_R)
- XBee Header allows direct connection to all available XBee I/O if desired. Signal conditioning may be required.

AppBee-SIP to XBee Interfacing Header

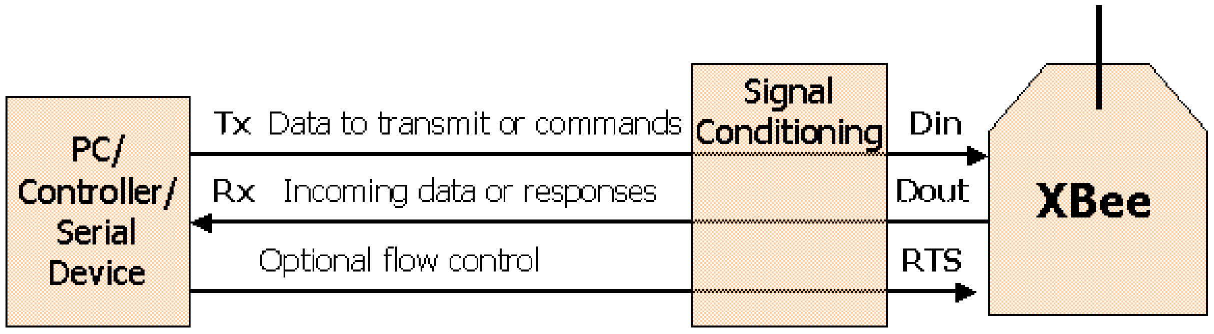
- 6 Pin header for power and I/O
- TX, RX
- RTS input for flow control
- SLP input for sleep request
- Additional 4-pin header for Parallax Prop Plug connection.
- XBee Header allows direct connection to all available XBee I/O if desired. Signal conditioning may be required.

XBee Base Station from Surveyor Corporation

- Direct XBee to a PC via USB for interfacing, configuration and firmware updates.

Device Connections and Data Communications

The XBee unit handles the packaging of data for transmission and error checking and acknowledgements on reception. All that is required is to send the device serial asynchronous data (9600, 8-N-1 by default) for transmission and accepting data addressed to that particular node. The XBee is a 3.3V device using non-inverted data (3.3V = Logic 1, 0V = Logic 0). Signal conditioning from RS-232 ports or from the 5V BASIC Stamp are required. For simple data flow all that is required is data sent to Din and accepted from Dout. RTS may be enabled so that the XBee does not send data it received via RF before the interfaced device is ready to accept the data (Firmware version 1.083 required).

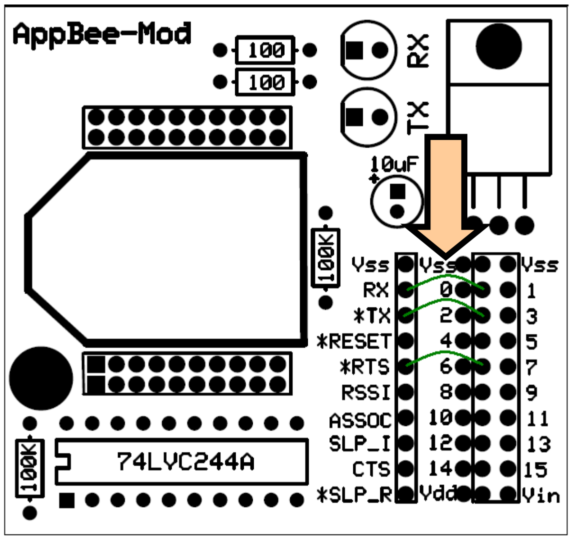

PC/ Controller/ Serial Device
Tx Data to transmit or commands
Rx Incoming data or responses
Optional flow control
Signal Conditioning
Din
Dout
RTS
XBee

Simple XBee data communications with interfaced device

By default, all devices are at address 0 and send to address 0. With a little coding addresses may be changed and flow control may be added.

BASIC Stamp to XBee Using the AppBee Mod Board

The AppBee Mod board plugs into the AppMod header on the Parallax Board of Education (BOE) and other common boards. To provide flexibility in pin assignments, jumper wires must be placed between the AppMod header 20-pin connector and XBee communications header as shown where P0 (0) of the BASIC Stamp is connected to RX, 2 to *TX and 6 to *RTS.



XBee header connections starting with * denote it is an input to the XBee and are signal conditioned to 3.3V through the 74LVC244A buffer. The BASIC Stamp header also has an additional row of solder connectors in the event the user wishes to have soldered connections between the XBee header pads and the BASIC Stamp header (Even P-numbers only).

A summary of XBee header I/O is as follows:

XBee Header Pin	Description
Vss	Vss or Ground of the BASIC Stamp and system.
RX	Received data from the XBee's Dout to the BASIC Stamp.
***TX**	Data from the BASIC Stamp to the XBee's Din, buffered to 3.3V.
***Reset**	Optional RESET input for the XBee. Connecting and bringing this line LOW (0V) will reset the XBee. Use of this line is recommended if the BASIC Stamp modifies baud rates or guard times so that on a BASIC Stamp reset the XBee can be reset by code prior to sending configuration information.
***RTS**	Ready to Send – Buffered input to the XBee to indicate the BASIC Stamp is ready to receive data. Use SERIN with flow control to accept data. Highly recommended to ensure data is not missed. (XBee firmware 1.083 required)
***RSSI**	Receiver signal strength indicator. A PWM output of the XBee, which can be configured to indicate the strength of the last, received RF data. This may be accomplished in command mode with serial data also.
ASSOC	XBee output indicating it has associated with a network under certain configurations. Please see XBee documentation. This can be read by the

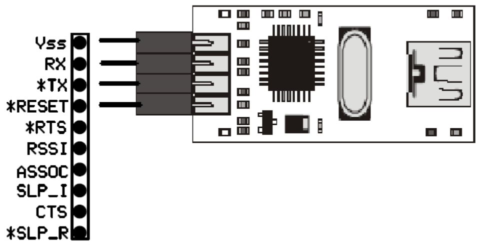
	BASIC Stamp or used to drive an LED through a 100-ohm resistor.
SLP_I	XBee output, Sleep Indicate. When sleep modes are used, this output can be used to indicate the status of the device. This may be read by the BASIC Stamp or used to drive an LED through a 100-ohm resistor.
CTS	Output of the XBee. May be used for flow control to ensure the XBee is ready to receive new data.
***SLP_R**	Buffered input to the XBee, Sleep Request. When configured, a HIGH on the pin will place the XBee in a low-power sleep mode (<10uA) conserving battery life. The XBee cannot send nor receive serial or RF data when sleeping.

Be sure NOT to configure BASIC Stamp I/O as outputs to the outputs from the XBee. A 5V output of the BASIC Stamp to a 3.3V output of the XBee may be damaging to either or both devices. Prior to connecting the XBee, download a "DEBUG 1" program to the BASIC Stamp to ensure all I/O are set to Inputs.

PC to XBee Using AppBee Mod Board with the USB2SER

The XBee may communicate to a personal computer (PC) via USB using Parallax's USB2SER device. This allows full interface features using Maxstream's X-CTU features except the ability to download new firmware.

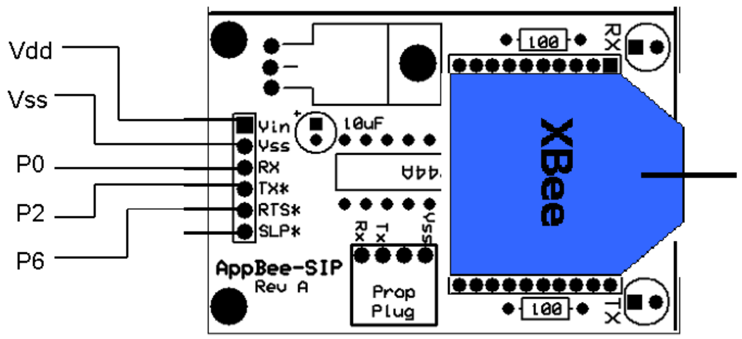
The XBee board should not be connected to the BASIC Stamp on RX/TX/RESET while the USB2SER is connected.



By supplying power to Vss and Vin (5V to 18V) of the AppBee Mod board header and using the USB2SER device, stand-alone use of the XBee with PC may be accomplished.

BASIC Stamp to XBee Using the AppBee SIP Board

The AppBee SIP board allows use the XBee Modem with a breadboard, or other 0.1" (2.54mm) spaced holes with a single 6-pin header. It provides 3.3V regulation and buffering from 5V devices. Vin should be in the range of 5V to 12V. With the BASIC Stamp, Vin of the supply is the preferred source due to XBee current draw.



The AppBee SIP provides connections for TX/RX/RTS and a Sleep request input to the XBee as discussed in the AppBee header discussion.

Ensure voltages are not supplied to I/O pins without power applied: 5V to 12V on Vin, and ground or 0V on Vss.

While the I/O is limited, connections to the breadboard may be made directly from the XBee module headers, for outputs such as Sleep indicate or inputs such as RESET. Please see the XBee Documentation for pin numbering.

Be sure to condition any 5V inputs to the XBee's I/O to 3.3V or less through the use of voltage dividers or other devices.

PC to XBee Using AppBee SIP Board with the Prop Plug

The XBee may communicate to a PC via USB using Parallax's Prop Plug device. This allows full interface features using MaxStream X-CTU features, except the ability to download new firmware.

By supplying power to Vss and Vin (5V to 18V) of the AppBee SIP board header and using the Prop Plug device, stand-alone use of the XBee with PC may be accomplished.

The XBee should not be connected to the BASIC Stamp on RX/TX while the Prop Plug is connected.

Simple BS2 to BS2 Example

Simple code to transmit values between units. Data is sent/received as decimal values.

```
' {$STAMP BS2}
' {$PBASIC 2.5}
'***********************
' Simple_tx.bs2
' Example to transmit decimal
' values 0 to 255
' ***********************
RX              PIN 0                   ' Receive Pin
TX              PIN 2                   ' Transmit Pin
X               VAR Byte

HIGH TX                                 ' Idle transmit pin
DO
  X = X + 1
  SEROUT TX,84, [DEC X,CR,CR]           ' Send value of X as decimal
                                        ' Second CR is added byte buffer for flow control example
  PAUSE 500
LOOP
```

To receive:

```
' {$STAMP BS2}
' {$PBASIC 2.5}
' ***************************
' simple_rx.bs2
' Example to receive decimal value
' and display in DEBUG Window
' ***************************
RX              PIN 0                   ' Receive Pin
TX              PIN 2                   ' Transmit Pin
X               VAR Byte
HIGH TX                                 ' Idle transmit pin
DO
 SERIN RX, 84, [DEC x]                  ' Receive data
 DEBUG DEC X, CR
LOOP
```

Simple BS2 to BS2 Receiver with Flow Control (RTS)

On the receiver, the XBee is placed in command mode, and a command of ATD6 1 is sent to enable flow control. This command mode is covered in more detail shortly. This example will demonstrate that data is buffered by waiting until a key is pressed to retrieve and display the data. Note in the transmit code, two CR's are used to prevent loss of significant data from the next packet in the buffer.

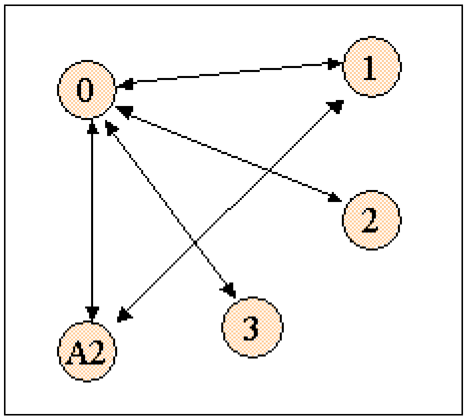
```
' {$STAMP BS2}
' {$PBASIC 2.5}
'*****************************************
' rx_flow_control.bs2
' Simple data with flow control
' Example to receive decimal value and
' display in DEBUG Window using flow control
```

```
'*****************************************
' Example to receive decimal value and display in DEBUG Window using flow control
RX                  PIN 0                  ' Receive Pin
TX                  PIN 2                  ' Transmit Pin
RTS                 PIN 6                  ' RTS flow control pin
X      VAR Byte
HIGH TX                                    ' Idle transmit pin


DEBUG CLS, "Configuring XBee for flow control", CR
PAUSE 2000                                 ' 2 second guard time
SEROUT TX, 84,["+++"]                      ' Enter command mode
PAUSE 2000                                 ' 2 second guard time
SEROUT TX, 84,["ATD6 1",CR]                ' Enable flow control
SEROUT TX, 84,["ATCN",CR]                  ' Exit command mode

DO
  DEBUG CR,"Press a key to retrieve data in buffer", CR
  DEBUGIN X                                ' Wait for key press
GetData:
  SERIN RX\RTS, 84,100,Timeout,[DEC x]   ' Receive data with 100mS timeout/flow control
  DEBUG CR,DEC X                           ' Display data
  GOTO GetData                             ' Loop back for more data
Timeout:                                   ' No data, repeat
LOOP
```

Network Communications and Configuration

Point-to-Point Messages

Each node on the network is identified by a unique 16-bit address allowing over 65,000 devices on a single network. As illustrated, a device at address 0 may communicate with various addresses, or device node 1 may communicate directly with another node. This forms a point-to-point network where anyone node may communicate with another (range dependent).



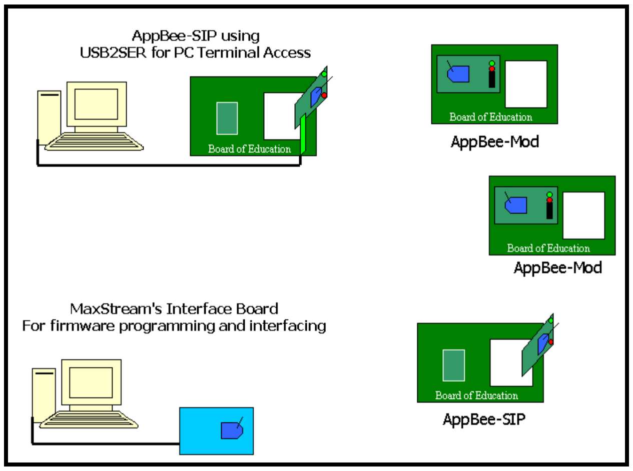
The XBee units have numerous settings for configuration, two of which are:
MY – The node's own address (16-bit)
DL – Destination Low (16-bit)

By configuring these, a node is given an address, and a node to communicate to. For example, for node 0 to send data to node 1:
MY = 0
DL = 1

The data will be delivered with error checking, an acknowledgement, and up to 3 retries if needed. For node 2 to send data to node 0:
MY = 2
DL = 0

Of course, node 1 may also send data to node A2 (hexadecimal) by setting:
DL = A2

The following figure shows a network of 5 XBee's using various monitoring and communications choices. Two are connected to the PC's to act as terminal programs, or to accept data into specialized software, such as StampPlot or StampDAQ.



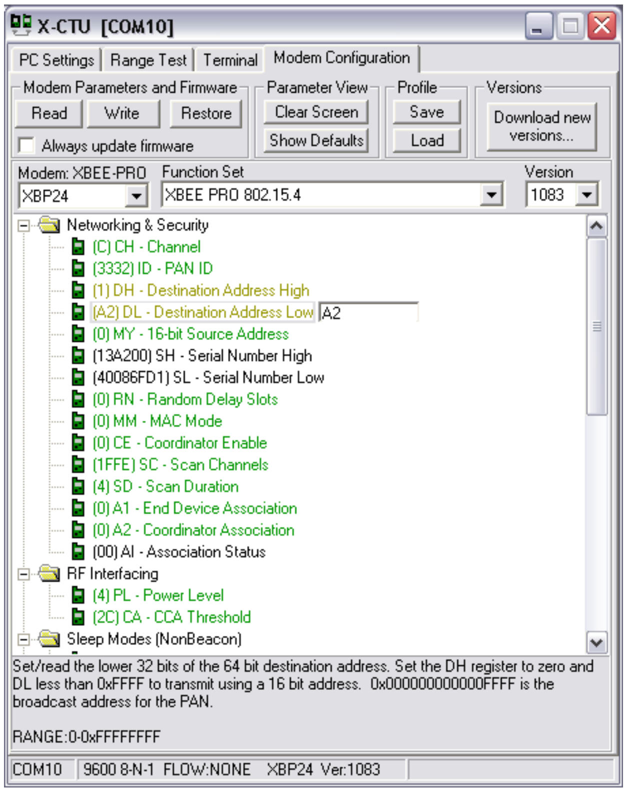
Point-to-Multipoint Broadcast Messages

A broadcast message is one sent from a node to all listening nodes. This is performed without acknowledgements or retries. To send a broadcast message, DL is set to FFFF. This is useful if you wish to poll all devices or send a control message to all. When devices send data they use clear channel assessment (CCA) to help ensure they do not talk over one another.
DL = FFFF

Configuring the XBee from X-CTU Software

By default, all devices are assigned an address of 0 on power-up. This may be changed through either the X-CTU software Modem Configuration tab or using AT commands from a terminal window, or serially from a device, such as the BASIC Stamp.

This figure shows configuring from the X-CTU software using the USB2SER (AppBee-MOD), Prop Plug (AppBee-SIP) or USB Base device. Once the device is read (READ button), settings are modified and new settings downloaded using the WRITE button.



```
Modem: XBEE-PRO   Function Set             Version
XBP24             XBEE PRO 802.15.4        1083

Networking & Security
  (C) CH - Channel
  (3332) ID - PAN ID
  (1) DH - Destination Address High
  (A2) DL - Destination Address Low  A2
  (0) MY - 16-bit Source Address
  (13A200) SH - Serial Number High
  (40086FD1) SL - Serial Number Low
  (0) RN - Random Delay Slots
  (0) MM - MAC Mode
  (0) CE - Coordinator Enable
  (1FFE) SC - Scan Channels
  (4) SD - Scan Duration
  (0) A1 - End Device Association
  (0) A2 - Coordinator Association
  (00) AI - Association Status
RF Interfacing
  (4) PL - Power Level
  (2C) CA - CCA Threshold
Sleep Modes (NonBeacon)

Set/read the lower 32 bits of the 64 bit destination address. Set the DH register to zero and
DL less than 0xFFFF to transmit using a 16 bit address.  0x000000000000FFFF is the
broadcast address for the PAN.

RANGE:0-0xFFFFFFFF

COM10 | 9600 8-N-1  FLOW:NONE   XBP24  Ver:1083
```

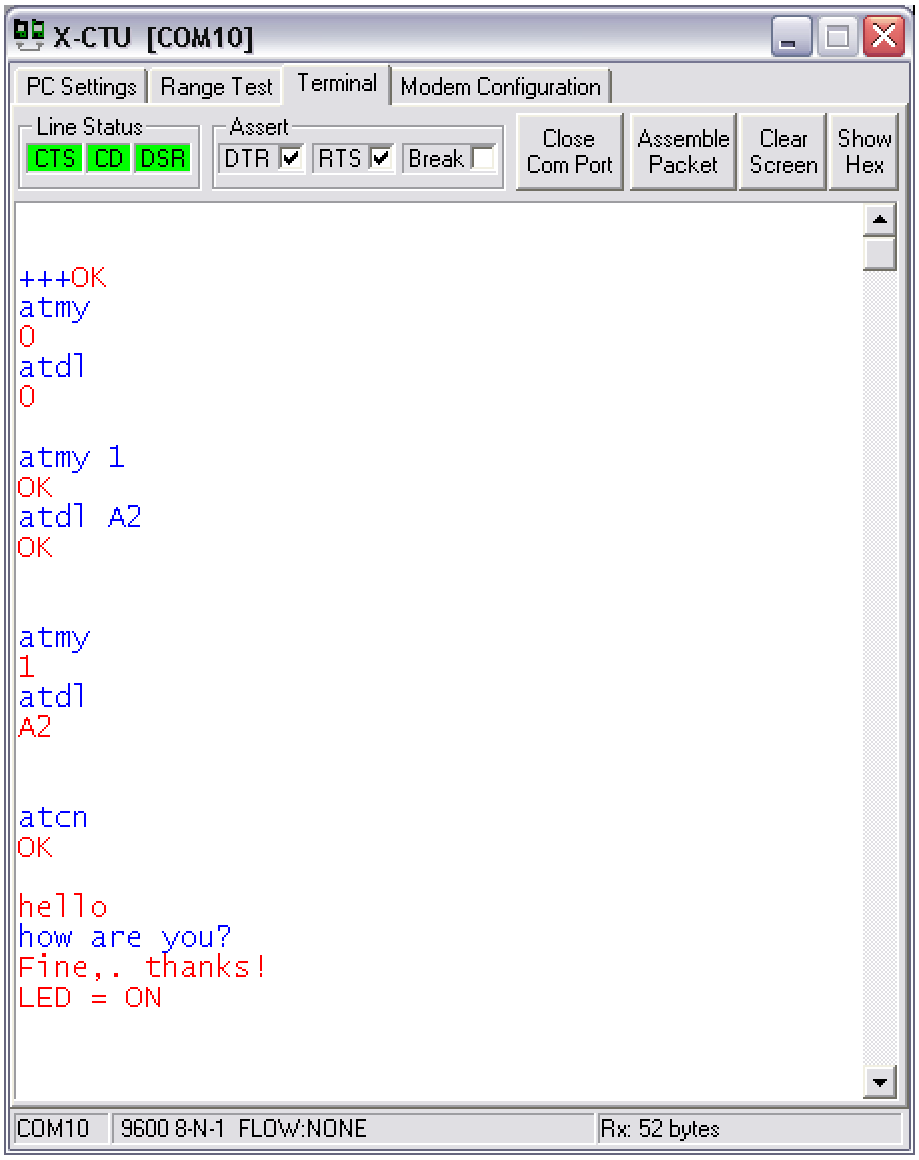
MaxStream's X-CTU Modem Configuration Software

Configuring the XBee from Terminal

The next figure shows the configuration using a terminal window set for 9600, 8-N-1. The device is placed in command mode by issuing a "+++". To ensure data that contains this does not shift the unit into command mode, a guard time before and after is required – that is, no other data can be sent for a time period before or after the +++, this includes carriage returns.

Once in command mode, AT commands are sent to configure the unit. ATCN is used to exit command mode. Sending the command with no value will cause the unit to return the current value or setting as is shown.

In this example, once in command mode the values of DL and MY are requested, then modified, then requested once again. Command mode is exited and data is sent between terminals.



```
+++OK
atmy
0
atdl
0

atmy 1
OK
atdl A2
OK


atmy
1
atdl
A2


atcn
OK

hello
how are you?
Fine,. thanks!
LED = ON
```

```
COM10 | 9600 8-N-1  FLOW:NONE | Rx: 52 bytes
```

Even though the X-CTU software is used for the example, HyperTerminal® or any terminal program running at 9600 Baud, 8-N-1, may be used. Data may be sent and received through the terminal window as well.

Summary of Common Configuration Commands

A summary of common configuration commands is listed. Again, please see the XBee documentation for full discussions and many more options for use.

Command	Description
MY	Sets/ Read the current node address. Address are in hexadecimal, 0-FFFE ATMY ATMY A2
DL	Sets/ Read the destination node address. Addresses are in hexadecimal, 0-FFFE ATDL ATDL 1 Use FFFF to broadcast to all nodes.
CH	Sets/Reads the current RF channel in the 2.4GHz range. This can be used for frequency separation between different 802.15.4 networks or to limit interference from other 2.4GHz devices. ATCH ATCH A Please see XBee documentation for allowed values.
ED	Energy Detect – Performs a scan of background energy. Useful in deciding on a channel to use. Use from the terminal window.
ID	Sets/Reads the current Personal Area Network ID (PAN ID). Allows network separation between devices in same location. ATID ATID 3350 Please see XBee documentation for allowed values.
NI	Sets/Reads the node descriptive identification, 20 characters. ATNI ATNI Temperature Sensor
BD	Sets/Reads the value for the current serial baud rate. 0 - 7 (standard baud rates) 0 = 1200 bps 1 = 2400 2 = 4800 3 = 9600 4 = 19200 5 = 38400 6 = 57600 7 = 115200 ATBD 2
D6	Sets/Reads the status of RTS flow control. Set to 1 to use RTS flow control. ATD6 ATD6 1
GT	Sets/Reads the value of guard times required before and after issuing +++ to enter command mode. Decrease this value to 3 to allow fast configuration changes, such as the need to quickly change destination addresses. Values are in hexadecimal, please see XBee documentation for a full discussion. ATGT ATGT 3
RO	Sets/Reads how long to wait for characters before packetizing and sending. Value is 3 by default, FF maximum. This is good to ensure a line of data from the BASIC Stamp is sent as a single packet instead of being broken up where data from other units may be interposed. ATRO FF
ND	Performs a node discovery – Allows viewing of all nodes, including addresses, MAC addresses, name identification, and RSSI level (dBm). ATND
DB	Reads the dBm level of the last reception in hexadecimal, please see XBee documentation for a full discussion. ATDB

SM	Sets/Reads the sleep mode of the XBee. While sleeping, power consumption is <10uA. When set to 1, the SLP_R input of the AppBee Mod board may be used to place the device in an idle state. ATSM ATSM 1
WR	Writes the current configuration to XBee non-volatile memory so that on power up the settings will persist. ATWR
RE	Restores the default settings of the XBee. ATRE
PL	Sets/Reads the current output power level of XBee-PRO Modems. Please see XBee documentation. ATPL 3
CN	Exits command mode. A timeout period will also exit from command mode. ATCN

Command instructions may also be issued in shorthand, such as:
ATMY 0, DL 0, D6 1, CN (and a carriage return)

Configuring the XBee from Code

The BASIC Stamp can use the SEROUT instruction to place the XBee in command mode and send configuration information. A guard time, sending +++, followed by a guard time is required. Once configured, ATCN is issued to exit command mode.

```
' {$STAMP BS2}
' {$PBASIC 2.5}
' *****************************************
' * Config_XBee_Example.BS2                *
' * Illustrates configuring the XBee       *
' * from the BASIC Stamp.                          *
' *****************************************
myAddr          CON $1          ' Node Address in hex
DestAddr        CON $0          ' Destination address in hex

Baud            CON 84          ' Baud rate, 9600, 8-N-1, non-inverted, on BS2.

RX              PIN 0           ' Receive Pin
TX      PIN 2           ' Transmit Pin
RTS             PIN 6           ' Flow control Pin

HIGH TX                         ' Place TX pin High (Idle state)

DEBUG CLS, "Configuring XBee..."
PAUSE 2000                                      ' Guard time for command sequence
SEROUT TX, Baud, ["+++"]                        ' Enter command mode
PAUSE 2000                                      ' Guard time for command sequence
SEROUT TX, Baud, ["ATNI BS2 Test Node", CR,     ' Set description
                  "ATMY ", HEX myAddr, CR,      ' Set node address
                  "ATDL ", HEX DestAddr, CR,    ' Set destination address
                  "ATD6 1", CR,                 ' Use RTS for flow control
                  "ATCN", CR]                   ' Exit command mode

PAUSE 1000
DEBUG "Configuration Complete!", CR
```

```
DO                                              ' Send message forever at 1S intervals
  SEROUT TX, Baud,["Hello Node!", CR]
  PAUSE 1000
Loop
```

Communication Strategies

These examples illustrate different communication and control strategies with the BASIC Stamp. The code assigns pin connections using PIN instructions and are configured for:
TX on P0
RX on P2
RTS on P6

Accepting Data to the BASIC Stamp

The BASIC Stamp can accept byte data very easily from the XBee using the SERIN instruction:
SERIN RX\RTS, Baud, [ByteVariable]

Whether accepting data from another XBee equipped BASIC Stamp or from a terminal window, flow control should be used with the BASIC Stamp. Ensure the XBee is configured for RTS flow control by setting ATD6 1.

The following code will accept a byte of data to be displayed as a character. The node is set to address 1 (MY = 1), and the destination is set to 0. Data from the transmitting unit should have DL = 1 or DL = FFFF. Note that with a rapid string of characters, every other character will be missed due to RTS being slow in stopping the next byte from being sent from the XBee.

```
' {$STAMP BS2}
' {$PBASIC 2.5}
' ****************************************
' * Accept_Character.bs2                          *
' * Illustrates accepting a Byte                  *
' * received as a character                       *
' ****************************************
myAddr          CON $1          ' Node Address
DestAddr        CON $0          ' Destination address

Baud            CON 84          ' Baud rate, 9600, 8-N-1, non-inverted, on BS2.

RX              PIN 0           ' Receive Pin
TX      PIN 2           ' Transmit Pin
RTS             PIN 6           ' Flow control Pin

RFin            VAR Byte

HIGH TX                         ' Set TX pin to idle state

DEBUG CLS, "Configuring XBee..."
PAUSE 2000                                 ' Guard time for command sequence
SEROUT TX,Baud,["+++"]                     ' Enter command mode
PAUSE 2000                                          ' Guard time for command sequence
SEROUT TX,Baud,["ATNI BS2 Test Node",CR,            ' Set description
           "ATMY ", HEX myAddr,CR,         ' Set node address
           "ATDL ", HEX DestAddr,CR,              ' Set destination node address
           "ATD6 1",CR,                                   ' Use RTS for flow control
           "ATCN",CR]                             ' Exit command mode

PAUSE 1000
DEBUG "Configuration Complete!", CR
DO
 SERIN RX\RTS,Baud,[RFin]                         ' Accept and view as character
 DEBUG RFin
LOOP
```

A timeout on SERIN may also be used:

```
DO
 SERIN RX\RTS,Baud,100,Timeout,[RFin]      ' Accept and view as character
 DEBUG Rfin
Timeout:
  ' Other processing code
LOOP
```

Decimal values may also be accepted using the DEC Modifier. Data sent must end with a carriage return.

```
 SERIN RX\RTS,Baud,[DEC RFin]                         ' Accept and view as decimal value
 DEBUG DEC RFin, CR
```

Controlling and Polling Nodes with Codes

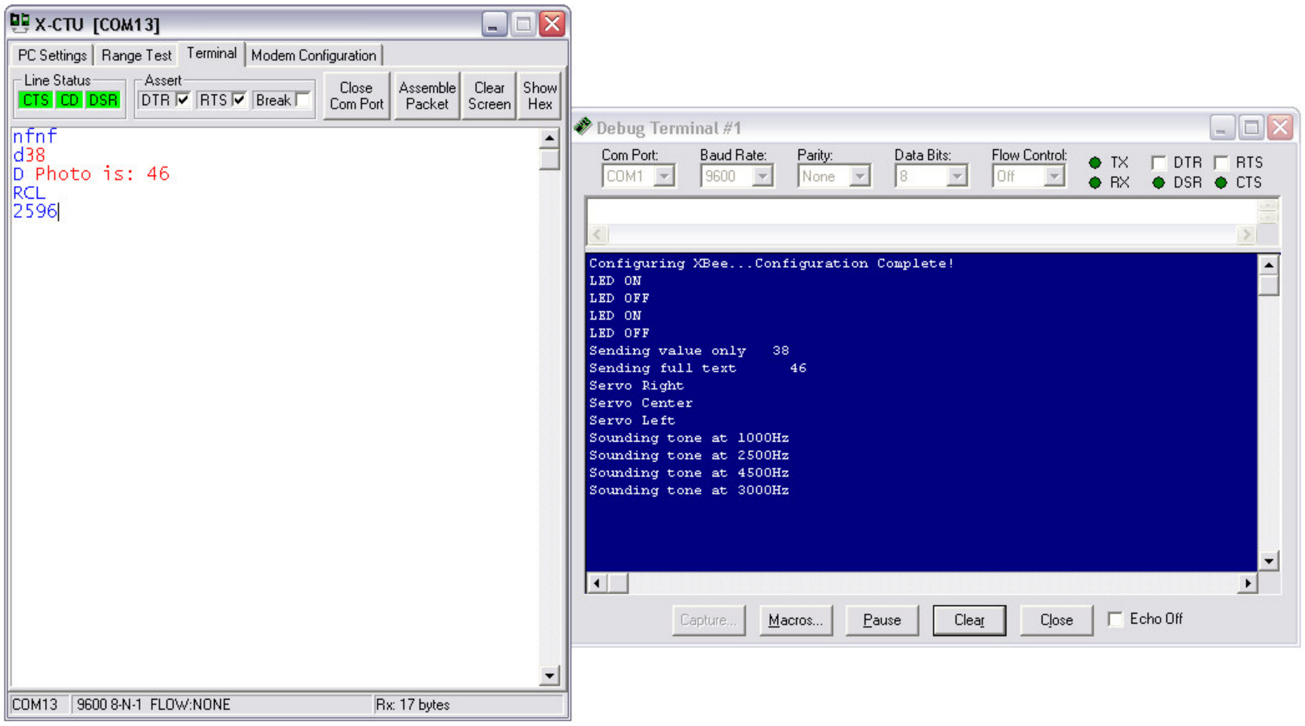
Simple control action and data polling can be performed. For example, the following hardware control and reading of data can be performed:

- Servo on P12
- LED on P7
- Speaker on P1
- Photo Resistor and Capacitor (RC Network) on P10

From a terminal window connected via a USB2SER to an XBee configured to send data to a node at address 1 or FFFF and a MY of 0:

- Send **d** to read just data
- Send **D** to read text + data (Requires string processing by BASIC Stamp)
- Send **0** to **9** to sound tones
- Send **n** (as in On) and **f** (as in off) to control the LED
- Send **L, R, C** to position the servo

With an array of such boards, each can be read and controlled individually by setting the DL to that boards address, or all at once using a broadcast
(DL = FFFF).



```
nfnf
d38
D Photo is: 46
RCL
2596
```

```
Configuring XBee...Configuration Complete!
LED ON
LED OFF
LED ON
LED OFF
Sending value only   38
Sending full text      46
Servo Right
Servo Center
Servo Left
Sounding tone at 1000Hz
Sounding tone at 2500Hz
Sounding tone at 4500Hz
Sounding tone at 3000Hz
```

```
' {$STAMP BS2}
' {$PBASIC 2.5}
' ******************************************
' * Node_Control.bs2                        *
' * Illustrates control and polling        *
' * of nodes                               *
' ******************************************
myAddr      CON $1          ' Node Address
DestAddr    CON $0          ' Destination address

Baud        CON 84               ' Baud rate, 9600, 8-N-1, non-inverted, on BS2.

RX          PIN 0                ' Receive Pin
TX          PIN 2                ' Transmit Pin
RTS         PIN 6                ' Flow control Pin
LED         PIN 7
Servo       PIN 12
Buzzer      PIN 1

RC          PIN 10

RFin        VAR Byte
DataOut     VAR Byte
x           VAR Byte

HIGH TX                                   ' Set TX pin to idle state

DEBUG CLS, "Configuring XBee..."
PAUSE 2000                                ' Guard time for command sequence
SEROUT TX,Baud,["+++"]                    ' Enter command mode
PAUSE 2000                                ' Guard time for command sequence
SEROUT TX,Baud,["ATNI Photo Node",CR,  ' Set description
```

```
        "ATMY ", HEX myAddr,CR,              ' Set node address
        "ATDL ", HEX DestAddr,CR,            ' Set destination node address
        "ATD6 1",CR,                         ' Use RTS for flow control
        "ATCN",CR]                           ' Exit command mode
PAUSE 1000
DEBUG "Configuration Complete!",CR

DO
 GOSUB READ_Data
 GOSUB ReadnControl
LOOP

DEBUG CLS,"Ready",CR

READ_Data:                                   ' Read RC Network
 HIGH RC
 PAUSE 5
 RCTIME RC,1,DataOut
RETURN

Send_Full:                                   ' Tx data with text
 DEBUG "Sending full text      ", DEC DataOut,"  ", CR
 SEROUT TX, Baud, [" Photo is: ", DEC DataOut,CR]
RETURN

Send_Value:                                  ' TX data only
 DEBUG "Sending value only   ", DEC DataOut,"  ", CR
 SEROUT TX, Baud, [DEC DataOut,CR]
RETURN

ReadnControl:                                ' Read incoming data and control
 SERIN RX\RTS,Baud,250,Timeout2,[RFin]
 SELECT RFin
 CASE "D"
   GOSUB Send_Full
 CASE "d"
   GOSUB Send_Value
 CASE "N","n"
  DEBUG "LED ON                        ",CR
  HIGH LED
 CASE "F","f"
  DEBUG "LED OFF                       ",CR
  LOW LED
 CASE "L","l"
  DEBUG "Servo Left                    ",CR
  FOR x = 1 TO 50
    PULSOUT Servo, 500
    PAUSE 20
  NEXT
 CASE "C","c"
  DEBUG "Servo Center                  ",CR
  FOR x = 1 TO 50
    PULSOUT Servo, 750
    PAUSE 20
  NEXT
 CASE "R","r"
  DEBUG "Servo Right                   ",CR
  FOR x = 1 TO 50
```

```
    PULSOUT Servo, 1000
    PAUSE 20
   NEXT
 CASE "1" TO "9"
   DEBUG "Sounding tone at ", DEC RFin-48 * 500,"Hz        ",CR
   FREQOUT buzzer, 500, RFin-48 * 500
 ENDSELECT

 TimeOut2:
RETURN
```

Using the BASIC Stamp for Fast Polling a Network of Devices

Given a number of boards with the previous template for control, a 'coordinator' BASIC Stamp may poll the devices. Ensure each node is given a unique address, such as 1 through 10 for a value of MyAddr, and that they all have node 0 as the destination address (DestAddr). Notice that the guard time (ATGT 3) is reduced in the configuration to allow the DL address to be quickly updated to poll devices quickly in a round-robin fashion. With a change in guard time, the XBee may need to be reset with either a power cycling or use of the RESET line to accept new configuration changes.

```
' {$STAMP BS2}
' {$PBASIC 2.5}
' ****************************************
' * Polling_Coordinator.bs2                        *
' * Illustrates polling a number              *
' * of BS2-XBee Nodes                              *
' ****************************************
myAddr          CON $0
DestAddr        VAR Word

LowAddr         CON 1                   ' first to poll
HighAddr        CON 10                  ' last to poll, A hex
Baud            CON 84

RX              PIN 0
TX      PIN 2
RTS             PIN 6

X               VAR Byte
DataIn          VAR Word

HIGH TX
DEBUG CLS, "Configuring XBee..."
PAUSE 2000
SEROUT TX,Baud,["+++"]
PAUSE 2000
SEROUT TX,Baud,["ATGT 3",CR]                    ' **** minimal guard time
SEROUT TX,Baud,["ATNI Coordinator", CR]
SEROUT TX,Baud,["ATDL ", HEX DestAddr,CR]
SEROUT TX,Baud,["ATMY ", HEX myAddr,CR]
SEROUT TX,Baud,["ATD6 1",CR]
SEROUT TX,Baud,["ATCN",CR]
PAUSE 1000
DO
```

```
  FOR DestAddr = lowAddr TO highAddr              ' Cycle through addresses
   GOSUB SetDest                                  ' Go set address
   SEROUT Tx, Baud, ["d"]                         ' Send d for raw data
   DEBUG CR, "Polling ", DEC DestAddr                          ' show status to DEBUG
   SERIN Rx\RTS, Baud,1000, timeout, [DEC DataIn]   ' Accept returned data with timeout
   DEBUG "   Value:", DEC DataIn                  ' If data arrives, show it
   Timeout:
  NEXT                                            ' Check next
PAUSE 2000
LOOP                                              ' Repeat

SetDest:
   PAUSE 20
   SEROUT TX,Baud,["+++"]                   ' Enter command mode with minimal guard time
   PAUSE 2
   SEROUT TX,Baud,["ATDL", HEX DestAddr, ",CN",CR]        ' Update DL for polling
RETURN
```

Reading XBee Information

The BASIC Stamp may also request information from the XBee, such as current address, Receiver Signal Strength Indicator (RSSI) dBm levels, and a host of other information discussed in the XBee Documentation. Reducing the guard time will increase the speed at which this information is received. All data is returned as hexadecimal values and displayed in decimal. Valid ranges are –40dBm to nearly –100dBm.

A routine, WaitOK, is used to accept the XBee's "OK" so that it does not interfere with actual data being received.

```
' {$STAMP BS2}
' {$PBASIC 2.5}
' ***************************************
' * Get_RSSI.bs2                      *
' * Illustrates accepting a Byte       *
' * received as a character           *
' * and reads/displays RSSI dBm        *
' ***************************************
myAddr          CON $1          ' Node Address
DestAddr        CON $0          ' Destination address

Baud            CON 84          ' Baud rate, 9600, 8-N-1, non-inverted, on BS2.

RX              PIN 0           ' Receive Pin
TX       PIN 2          ' Transmit Pin
RTS             PIN 6           ' Flow control Pin

RFin            VAR Byte
dBm             VAR Byte

HIGH TX

DEBUG CLS, "Configuring XBee..."
PAUSE 2000                                      ' Guard time for command sequence
SEROUT TX,Baud,["+++"]                  ' Enter command mode
PAUSE 2000                                      ' Guard time for command sequence
SEROUT TX,Baud,["ATNI BS2 Test Node",CR,           ' Set description
```

```
            "ATMY ", HEX myAddr,CR,    ' Set node address
            "ATDL ", HEX DestAddr,CR,  ' Set destination node address
            "ATD6 1",CR,                        ' Use RTS for flow control
            "ATGT 3",CR,                        ' Reduce Guard Time
            "ATCN",CR]                 ' Exit command mode

PAUSE 1000
GOSUB WaitOK                                    ' Accept OK data
DEBUG "Configuration Complete!", CR
DO                                              ' Accept and view a character
  SERIN RX\RTS,Baud,1000,Timeout,[RFin]
  DEBUG RFin
  GOSUB GetRSSI
Timeout:
LOOP

GetRSSI:                                        ' Read RSSI data
  PAUSE 20                                      ' Guard time
  SEROUT TX, Baud, ["+++"]                      ' Command mode
  PAUSE 2
  SEROUT TX, Baud, ["ATDB",CR]                          ' Request level
  SERIN RX\RTS, Baud, 1000, Timeout2, [HEX dBm]   ' Accept data
  DEBUG "    dbm: -", DEC dBm                           ' Display data
Timeout2:
  SEROUT TX, Baud,["ATCN",CR]                           ' Exit command mode
  GOSUB WaitOK                                          ' Accept OK from XBee
  DEBUG CR
RETURN

WaitOK:
  SERIN RX\RTS, Baud, 100,Timeout3, [RFin]              ' Accept OK data
  GOTO WaitOK
Timeout3:
Return
```

Sending Direct Data with Sleeping

Of course, the unit does not need to be polled to send data. A simple SEROUT will transmit the data to the intended recipient at the DL address. Since the XBee uses the IEEE 802.15.4 protocol and CSMA/CA (like home wireless networking routers), data delivery without conflict from other units is nearly guaranteed.

This sample program will send data (a value of X), while putting both the XBee and BASIC Stamp asleep between transmissions. The SLP_R pin of the AppBee board should be connected to P8. SLP_I on the AppBee-MOD, or going directly to the XBee connector header (pin 13), the user may add a sleep indicator LED via a resistor to their breadboard.

```
'{$STAMP BS2}
' {$PBASIC 2.5}
' ******************************************************
' * Send_Sleep.bs2                                      *
' * Illustrates sending a byte every few seconds       *
' * while sleeping between transmits.                   *
' ******************************************************
```

```
myAddr          CON $1          ' Node Address
DestAddr        CON $0          ' Destination address

Baud            CON 84          ' Baud rate, 9600, 8-N-1, non-inverted, on BS2.

RX              PIN 0           ' Receive Pin
TX      PIN 2           ' Transmit Pin
RTS             PIN 6           ' Flow control Pin
SLP_R           PIN 8           ' Connect to SLP_R

X VAR byte
HIGH TX

DEBUG CLS, "Configuring XBee..."
PAUSE 2000                                      ' Guard time for command sequence
SEROUT TX,Baud,["+++"]                  ' Enter command mode
PAUSE 2000                                      ' Guard time for command sequence
SEROUT TX,Baud,["ATNI BS2 Test Node",CR,           ' Set description
          "ATMY ", HEX myAddr,CR,       ' Set node address
          "ATDL ", HEX DestAddr,CR,     ' Set destination node address
          "ATD6 1",CR,                          ' Use RTS for flow control
          "ATSM 1",CR,                          ' ***** Sleep mode of 1
          "ATCN",CR]                    ' Exit command mode
PAUSE 1000
DEBUG "Ready!",Cr

DO
 HIGH SLP_R                                     ' Put XBee to sleep
 SLEEP 5                                        ' Put BS2 to sleep for approx 5 seconds
 LOW SLP_R                                      ' Wake XBee
 PAUSE 10                                       ' Give XBee time to stretch and yawn
 x = x + 1                                      ' change a value
 SEROUT TX, Baud, [DEC x,CR]                    ' Transmit value
 DEBUG DEC x,CR                                 ' show in DEBUG
 PAUSE 10
Loop
```

Sending Data to StampDAQ

Data from multiple units may also send data to a base using (connected to a PC via a USB2SER) and collect that data in StampDAQ from Parallax (developed by SelmaWare Solutions), a serial to Microsoft Excel program. This example shows using a column to denote the node address from which the data is arriving.

```
' {$STAMP BS2}
' {$PBASIC 2.5}
' ***************************************************
' * XBee_StampDAQ.bs2                                          *
' * Illustrates sending a data to StampDAQ                              *
' ***************************************************
myAddr          CON $1          ' Node Address
DestAddr        CON $0          ' Destination address

Baud            CON 84          ' Baud rate, 9600, 8-N-1, non-inverted, on BS2.

RX              PIN 0           ' Receive Pin
TX      PIN 2           ' Transmit Pin
RTS             PIN 6           ' Flow control Pin

X       VAR Byte

HIGH TX

DEBUG CLS, "Configuring XBee..."
PAUSE 2000                                      ' Guard time for command sequence
SEROUT TX,Baud,["+++"]                  ' Enter command mode
PAUSE 2000                                      ' Guard time for command sequence
SEROUT TX,Baud,["ATNI BS2 Test Node",CR,                ' Set description
           "ATMY ", HEX myAddr,CR,      ' Set node address
           "ATDL ", HEX DestAddr,CR,    ' Set destination node address
           "ATD6 1",CR,                         ' Use RTS for flow control
           "ATCN",CR]                   ' Exit command mode
PAUSE 1000
DEBUG "Ready!",CR
SEROUT TX,Baud,[CR,"LABEL,TIME,ADDR,X",CR]      ' Label 3 TIME, Unit address, X

SEROUT TX,Baud,["CLEARDATA",CR]                 ' Clear all data columns (A-J) in Excel

DO
 FOR X = 0 TO 255                               ' Count from 0 to 255
                                                        ' Send String with data for Excel
   SEROUT TX,Baud,["DATA,TIME,", DEC myAddr, ",", DEC X, CR]
 PAUSE 500                                              ' 500mS wait before next data
 NEXT
LOOP
```

Sending Data to StampPlot

StampPlot may receive data from a base unit using a USB2SER and selecting the correct port in the Configuration window. In this example, the unit's address is used to select the plot channel

(0-9) and the color of the plot with the !ACHN instruction of StampPlot:
!ACHN channel, value, color

```
' {$STAMP BS2}
' {$PBASIC 2.5}
' ******************************************************
' * XBee_StampPlot.bs2                                   *
' * Illustrates sending data to StampPlot               *
' * via a base unit                                   *
' ******************************************************
myAddr          CON $1          ' Node Address
DestAddr        CON $0          ' Destination address

Baud            CON 84          ' Baud rate, 9600, 8-N-1, non-inverted, on BS2.

RX              PIN 0           ' Receive Pin
TX      PIN 2           ' Transmit Pin
RTS             PIN 6           ' Flow control Pin

X VAR Byte
HIGH TX

DEBUG CLS, "Configuring XBee..."
PAUSE 2000                                  ' Guard time for command sequence
SEROUT TX,Baud,["+++"]              ' Enter command mode
PAUSE 2000                           ' Guard time for command sequence
SEROUT TX,Baud,["ATNI BS2 Test Node",CR,          ' Set description
           "ATMY ", HEX myAddr,CR,    ' Set node address
           "ATDL ", HEX DestAddr,CR,  ' Set destination node address
           "ATD6 1",CR,                       ' Use RTS for flow control
           "ATCN",CR]                 ' Exit command mode
PAUSE 1000
DEBUG "Ready!",CR

DO
 FOR X = 0 TO 255                   'Count from 0 to 255
                      'Send String to StampPlot using channel and color of address (0-9)
   SEROUT TX,Baud,["!ACHN ", DEC myAddr,",", DEC X, ",", DEC myAddr, CR]

 PAUSE 500          '500 mS wait before next data
 NEXT
LOOP
```

Board Construction and Mounting

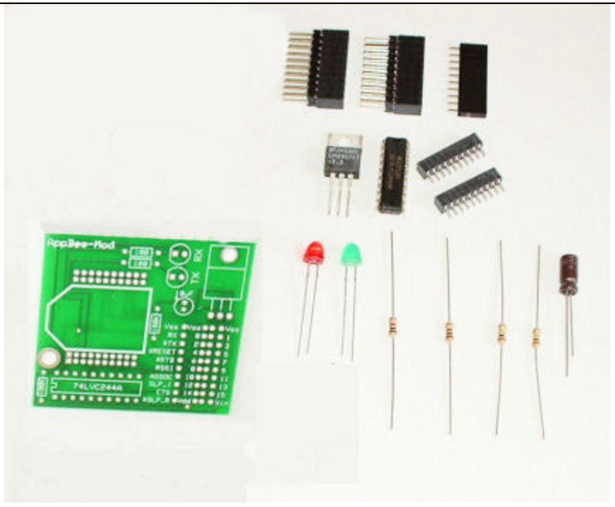
It is assumed the customer has experience soldering if the kit was purchased. If not, please research soldering online or ask someone else for assistance. We assume no liability for damage due to soldering errors or burns to the customer!

AppBee-MOD Construction

Parts Supplied:

Printed Circuit Board	1
20-pin 2mm headers for XBee mounting	2
20-pin 2.54mm headers for AppMod	2
10-pin 2.54mm header for XBee I/O	1
74LVC244A 5V to 3.3V buffer	1
100 ohms resistors (brown-black-brown)	2
100K ohm resistors (brown-black-yellow)	2
Red LED (Transmit - TX)	1
Green LED (Receiver – RX)	1

XBee – NOT INCLUDED
Jumper wires – NOT INCLUDED
Optional AppMod Standoff – NOT INCLUDED

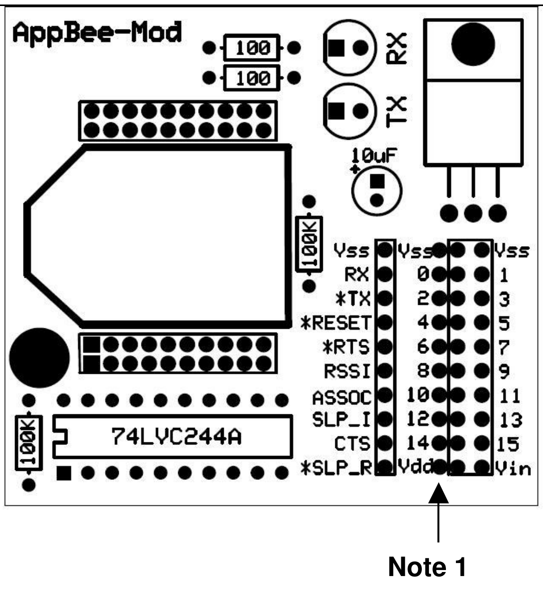


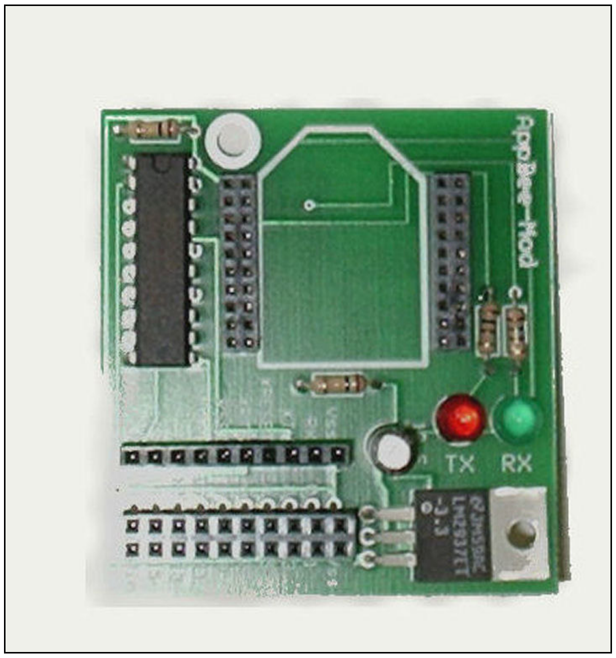
Be sure to place components in the correct directions, such as the 74LVC244A, capacitor (stripe NOT to +), and LEDs.

If you with to permanently attach wires between IO, leave off the 10-pin header and solder between it's pads and the spare pads for P0-P14 denoted by **Note 1**.

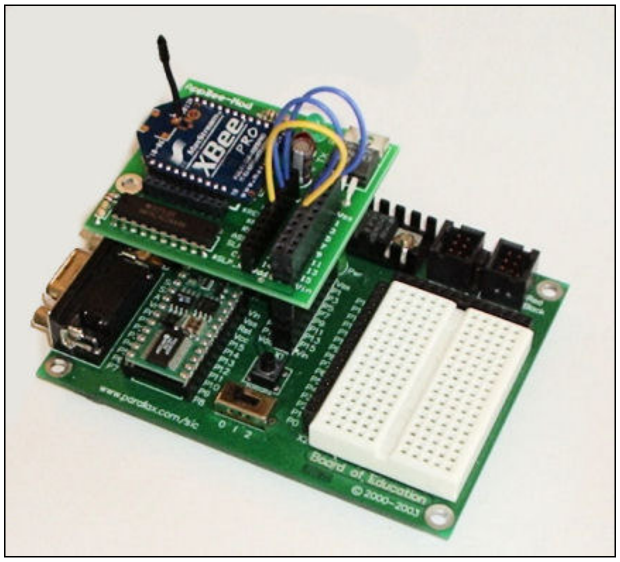
Ensure the XBee is in the direction shown when installing. The center-most row of pins on the headers will be used.

Connect jumper wires from desired BASIC Stamp I/O to XBee I/O.

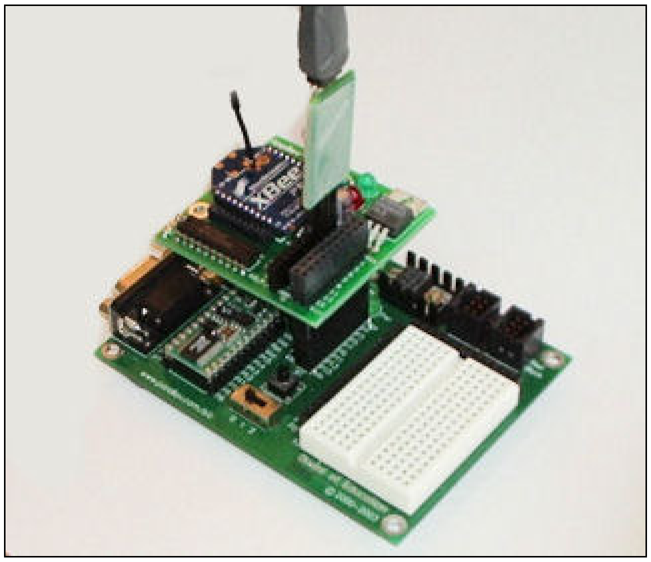




Assembled AppBee-MOD



XBee-PRO mounted on AppBee-MOD with jumpers for BASIC Stamp Communications.



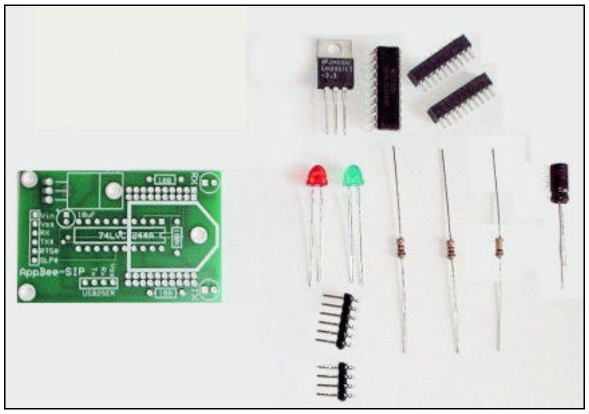
XBee-PRO mounted on AppBee-MOD using USB2SER for PC communications.

AppBee-SIP Construction

Parts Supplied:

Printed Circuit Board	1
20-pin 2mm headers for XBee mounting	2
74LVC244A 5V to 3.3V buffer	1
100 ohms resistors (brown-black-brown)	2
100K ohm resistor (brown-black-yellow)	1
Red LED (Transmit - TX)	1
Green LED (Receiver – RX)	1

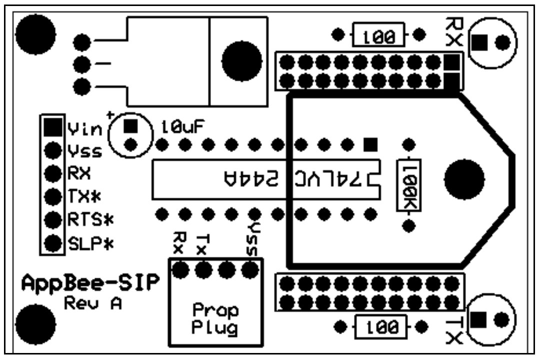
XBee – NOT INCLUDED
Jumper wires – NOT INCLUDED

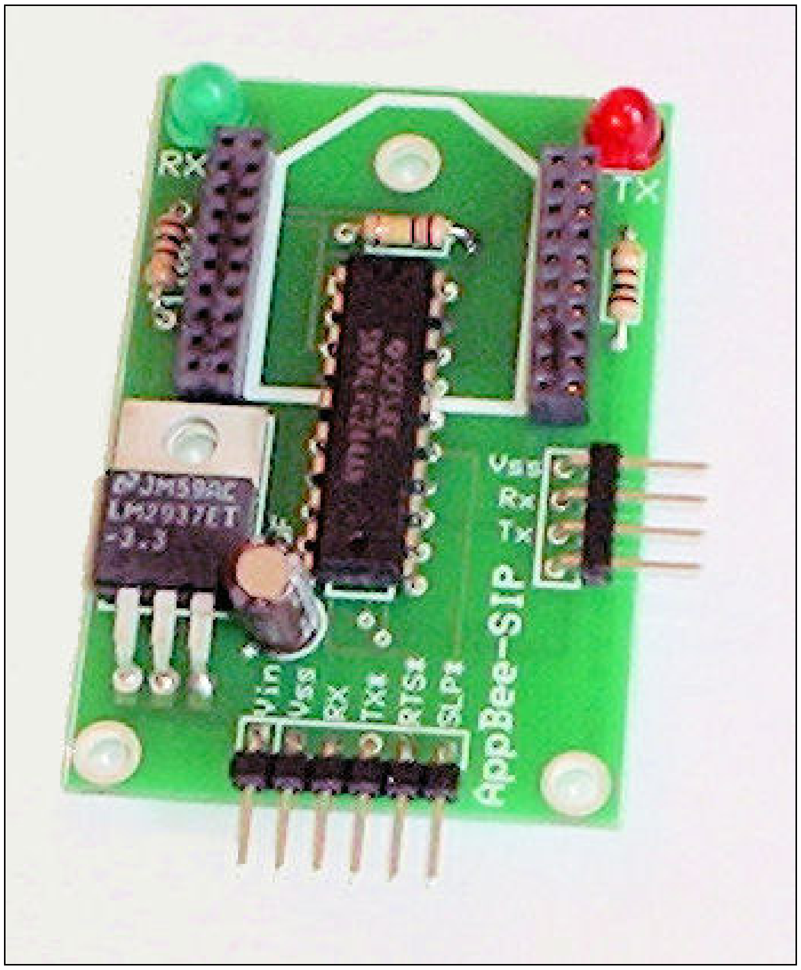


Be sure to place components in the correct directions, such as the 74LVC244A, capacitor (stripe NOT to +), and LEDs.

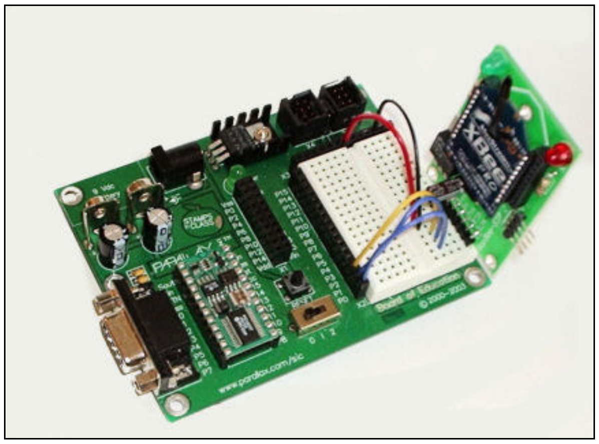
Ensure the XBee is in the direction shown when installing. The center-most row of pins on the headers will be used.

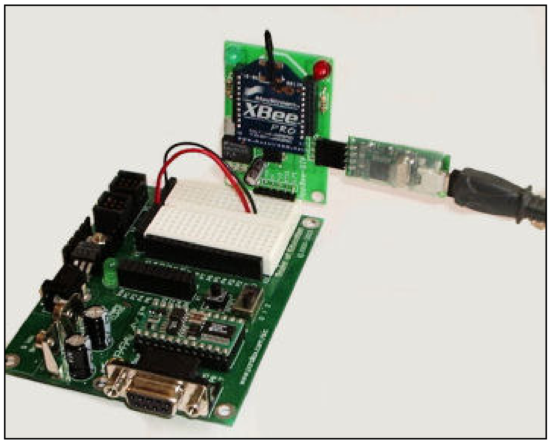
Connect jumper wires from desired BASIC Stamp I/O to XBee I/O.





AppBee-SIP Assembled


XBee-PRO mounted on AppBee-SIP with jumpers for BASIC Stamp Communications.


XBee-PRO mounted on AppBee-SIP using USB2SER for PC communications.
NOTE: REV A USES A PROP PLUG INSTEAD

FAQs

Why do I need an adapter board for the XBee?

The XBee is a 20 pin device with 2mm spacing (breadboards are typically 2.54mm or 0.1 in). Even if the pins did fit, it would gobble up most of the real estate on smaller breadboards. The XBee is also a 3.3V device. The adapter is regulated for 3.3V and has a 5V to 3.3V buffer. Voltages from the XBee to the controller do not need to be conditioned since they are higher than the BASIC Stamp's threshold voltage (1.8V).

What is IEEE 802.15.4 and ZigBee?

IEEE 802.15.4 is a low-rate communications protocol for wireless networks. The protocol is responsible for moving data between 2 addressable devices, much like Ethernet moves data (IEEE 802.3). This protocol uses the 900MHz or 2.4GHz (XBee) bands, and uses Direct Spread Spectrum Sequence to use very low power for transmission.

ZigBee is a trademark of a consortium of companies in developing compatible applications on top of 802.15.4. ZigBee can be compared to TCP/IP, which uses 802.3 to move data. ZigBee includes the ability to route data between points.

Can I communicate with my WiFi network?

Not directly. IEEE 802.15.4 operates on a different protocol (and purpose!) than WiFi (IEEE 802.11).

What is different about these devices from other RF devices, such as 433MHz ones?

Many RF devices simple send data freely without regard to errors or collisions with data from other devices. IEEE 802.15.4 is a fully implemented protocol ensuring as much as possible that data between devices do not collide (media access control), and the packets arrive without errors. Of course, addressing is a major benefit also.

Does the XBee support routing?

The version of the firmware discussed here, 1.083, is mainly 802.15.4. Other versions are in beta, which may include ZigBee functions of routing, though they may not be compatible with the examples illustrated here. With a little coding, routing may be designed into the BASIC Stamp to assist in these needs.

Can I use other I/O on the XBee that are not pinned out to the header?

Yes, for outputs you may simply run a jumper from the XBee header to your breadboard, keeping in mind it is 3.3V. For inputs to the XBee, your signal must be conditioned using a voltage or other means to 3.3V.

Can I use this board with controllers other than the BASIC Stamp?

Yes, any controller or device, 5V logic or 3.3V logic may be used, such as the Propeller or SX series from Parallax. The 3.3V regulator on the boards requires at least 5V to Vin. If this is unavailable on your 3.3V board, you may customize your board and keep off and jumper the regulator holes pin 1 to 3.

How much current does the AppBee board draw?

With an XBee, current draw is around 50mA. With an XBee-PRO, current draw can exceed 180mA when transmitting. Sleep mode may be used to lower current draw to <2mA, but the unit cannot send nor receive while sleeping. Lower power can be obtained by using a LT1521CST-3.3-ND surface mount voltage regulator instead of the T0-220 style provided. The boards have pads for them, but they may have solder mask over the top that may need to be scraped off.

How can I reprogram the firmware on my XBee?

A serial connection providing DTR and other hardware handshaking lines is required. The USB Base Station may be used to update the firmware on the XBee.

Is the unit RoHS compliant?

All parts and soldered used are rated as RoHS, though older stock of part (resistors, capacitors) from supplies may have been sent which was not compliant. So… It should be compliant, but not guaranteed at this time. Recent versions of the XBee itself are compliant and compliant solder is used in assembled boards.

Why can't I reach 1500 meters with the XBee-PRO or 300 meters with the XBee?

RF can be a tricky thing due to reflection, scattering and absorption. Even out doors, line of sight means more than the units are in sight of each other. Think of the signal between the 2 antennae as a football shaped field. The further the antennae are apart, the wider this football will be. If the field hits a surface, energy is reflected and will interfere with received data. For example, at 100 meters, the antennae should be 1.5 meters high for optimal transmission. At further distances, the higher the units need to be. With the AppBee SIP, the antenna being bent upward is not the best configuration, but should provide adequate transmission paths.

Are there other antenna styles available?

Yes, XBee's are available with a chip antenna and connectors for larger dipole antenna. These are available from vendors listed in the next section, but we plan on keeping a small stock of Whip antennas only.

Resource Links

Parallax, Inc:	
Starter kits:	http://www.parallax.com/html_pages/products/kits/starter_kits.asp
USB2SER Device:	http://www.parallax.com/detail.asp?product_id=28024
Prop Plug	http://www.parallax.com/detail.asp?product_id=32201
StampDAQ Software:	http://www.parallax.com/html_pages/downloads/software/software_stampDAQ.asp

Other controllers available such as the SX and the Propeller Chip multi-core.

MaxStream:	
XBee information:	http://www.maxstream.net/products/oem-rf-modules.php
X-CTU Software:	http://www.maxstream.net/support/downloads.php
White Papers:	http://www.maxstream.net/support/
DigiKey	http://www.digikey.com Search for 'XBee'
SelmaWare Solutions:	
AppBee information:	http://www.selmaware.com/appbee
StampPlot:	http://www.stampplot.com
Other XBee Solutions:	
Surveyor Corporation	http://www.surveyorcorp.com/

Warrantee Information:

AppBee-SIP and AppBee-MOD (the "Product") are warranted against defects in materials and workmanship under normal use, for a period of 90-days from the date of purchase. In the event of a product failure due to materials or workmanship for pre-assembled boards, SelmaWare will repair or replace the defective product. For warranty service, return the defective product to SelmaWare, shipping prepaid, for prompt repair or replacement. The foregoing sets forth the full extent of SelmaWare's warranties regarding the Product. Repair or replacement at SelmaWare's option is the exclusive remedy. THIS WARRANTY IS GIVEN IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, AND SELMAWARE SPECIFICALLY DISCLAIMS ALL WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL SELMAWARE, ITS SUPPLIERS OR LICENSORS BE LIABLE FOR DAMAGES IN EXCESS OF THE PURCHASE PRICE OF THE PRODUCT, FOR ANY LOSS OF USE, LOSS OF TIME, INCONVENIENCE, COMMERCIAL LOSS, LOST PROFITS OR SAVINGS, OR OTHER INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PRODUCT, TO THE FULL EXTENT SUCH MAY BE DISCLAIMED BY LAW. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES. THEREFORE, THE FOREGOING EXCLUSIONS MAY NOT APPLY IN ALL CASES. This warranty provides specific legal rights. Other rights which vary from state to state may also apply.

The XBee, XBee-PRO, USB2SER device, and any other devices used with the AppBee boards are covered under their manufacturer warrantees.