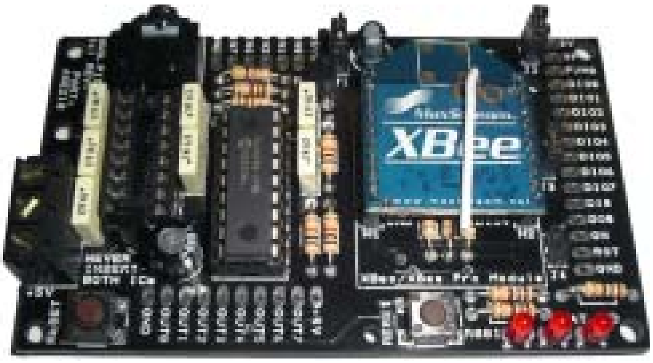
PICAXE AND XBEE TUTORIAL

Contents:

This article provides an introduction into how to use the XBee module with the PICAXE system.



Why use an XBee module for radio communications?

When considering a wireless PICAXE application most users will compare the 2.4GHz XBee modem units with the slightly lower-cost 433MHz RF modules. Whilst the 433 modules are low cost and may be suitable for some very simple PICAXE applications, the XBee modules offer considerable advantages.

The primary advantage is that the XBee modules are 'bi-directional'. Most budget 433 systems only transmit in one direction, so the transmitter has no idea whether the receiver is actually getting the data! The XBee modules transmit and receive in both directions, so you can easily test (at both ends) if the system is working correctly.

The second advantage is that of unique addressing. Each XBee unit has a unique serial number. This means two (or more) units can be set up to exclusively talk to each other, ignoring all signals from other modules. This is not easily achieved with 433 modules.

The third advantage is that the XBee module has in build 'data-packet' building and error-checking to ensure reliable data transmission.

Finally the XBee protocol allows for a number of 'channels'. By setting different units on different channels additional interference can be avoided.
Typical wireless networks could include

PC to remote PICAXE
e.g. remote control of robot/equipment

remote PICAXE to PC
e.g. data collection from a greenhouse

multiple remote PICAXEs to PC
e.g. a security system

remote PICAXE to remote PICAXE
e.g. different sensor modules of a weather station

AXE210 PICAXE Connect Board.

This tutorial assumes that you have already purchased and assembled two PICAXE 'AXE210 Connect' boards. Details on how to assemble these boards are provided in the separate datasheet AXE210.pdf, available from www.picaxe.co.uk

The PICAXE 'AXE210 Connect Board' has been designed to allow two modes of use:

- as a PICAXE to XBee wireless modem interface (PICAXE chip inserted)
- as a computer to XBee wireless modem interface (MAX3232 chip inserted)

This allows, for instance, a sensor connected to a PICAXE chip (on one AXE210 board) to transmit data to a second AXE210 board that is directly connected to a computer – allowing the remote data to be displayed and recorded on the computer screen!

Important Note:
When used as a PICAXE interface the PICAXE chip is inserted in the 18 pin socket. When used as a computer interface a MAX3232CPE chip is inserted in the 16 pin socket. **Never** insert both the PICAXE chip and MAX3232 chip at the **same** time!

PICAXE to XBee Connection

When a PICAXE chip is inserted in the 18 pin socket the following connections are made:

PICAXE output B.7 is connected to the XBee transmit pin (via jumper J1).
PICAXE output B.6 is connected to the XBee sleep pin (via jumper J4)
PICAXE input C.7 is connected to the XBee receive pin.

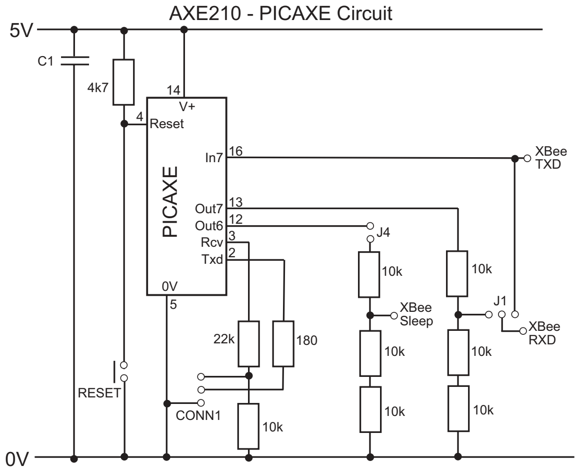
Therefore data to be transmitted by the XBee module can be output via the PICAXE command serout e.g.

```
serout B.7,T2400,("Data")
```

Data received by the XBee module can be processed by the PICAXE command serin e.g.

```
serin C.7,T2400, b1
```

As the PICAXE operates at 5V, whilst the XBee operates at 3.3V, the output pins from the PICAXE are interfaced to the XBee via a simple potential divider arrangement.

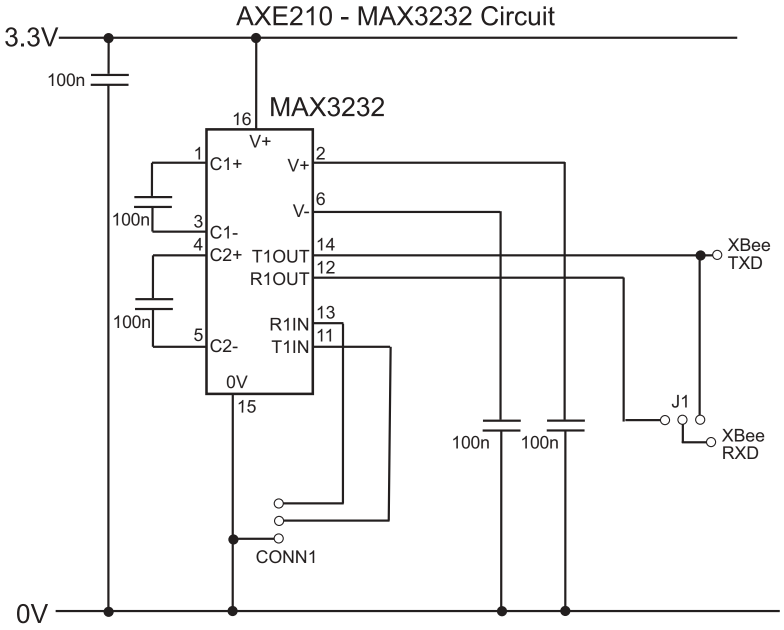


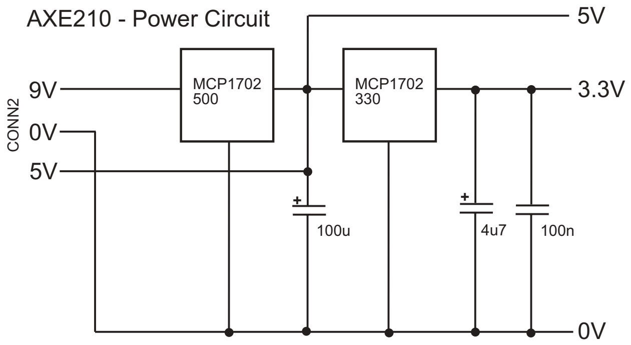
Computer to XBee Connection

When a MAX3232 chip is inserted in the 16pin socket the XBee module is connected (via the download cable) directly to the computer serial port. Therefore data can be transmitted from/received by the computer. The XBee module can also be configured via use of the Programming Editor XBee Wizard or configuration commands (see XBee Manual for more details).

If your laptop computer does not have an old style 9 pin serial port you will require the AXE027 USB cable.

Remember - never insert both the PICAXE chip and MAX3232 chip at the same time.



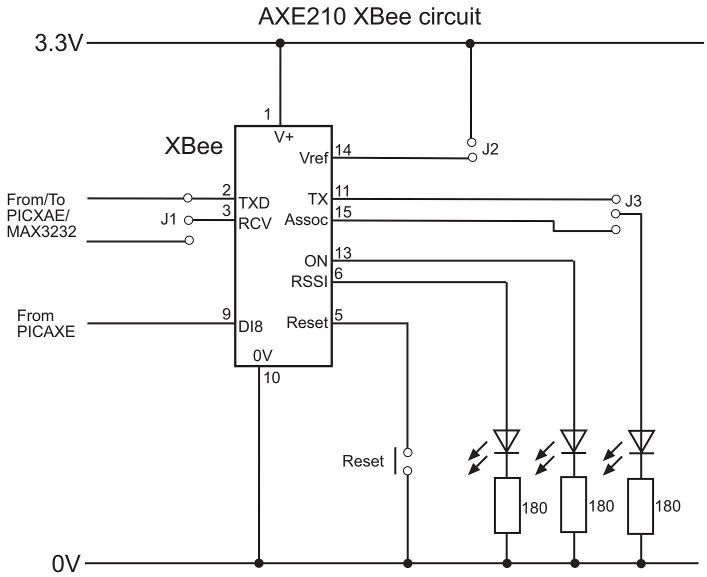


LED Indicators

The AXE210 board has three LED indicators.

RSSI connects to the XBee RSSI pin (pin 6)
STATUS connects to the XBee ON/Sleep pin (pin 13
IND connects via Jumper 3 to either XBee
Associate pin (pin 15) or Transmit pin (pin 11)

RSSI indicates the XBee received signal strength
STATUS indicates whether the XBee module is active/sleeping
IND indicates whether the XBee module has associated with another module
(Associate position) or is transmitting data (Transmit position).



Jumper Function Summary

The AXE210 board has 4 jumpers, labelled J1 to J4.
The default position of each jumper is shown in **bold**.

J1
Open - (not used)
Top - Normal use
Bottom - Loopback test (XBee DOUT connected to DIN)

J2
Open - XBee Vref not connected
Fitted - XBee Vref connected to 3.3V

J3
Open - IND. LED not connected
Top - IND. LED connected to XBee Transmit (DIO4)
Bottom - IND. LED connected to XBee Associate (DIO5)

J4
Open - PICAXE output 6 not connected
Fitted - PICAXE output 6 connected to XBee sleep pin (DI8)

Initial Setup Procedure.

For the initial setup procedure you require two AXE210 connect boards
Board A should contain a MAX3232CPE chip.
Board B should contain a PICAXE-18M2 (or older 18A or 18X) chip.

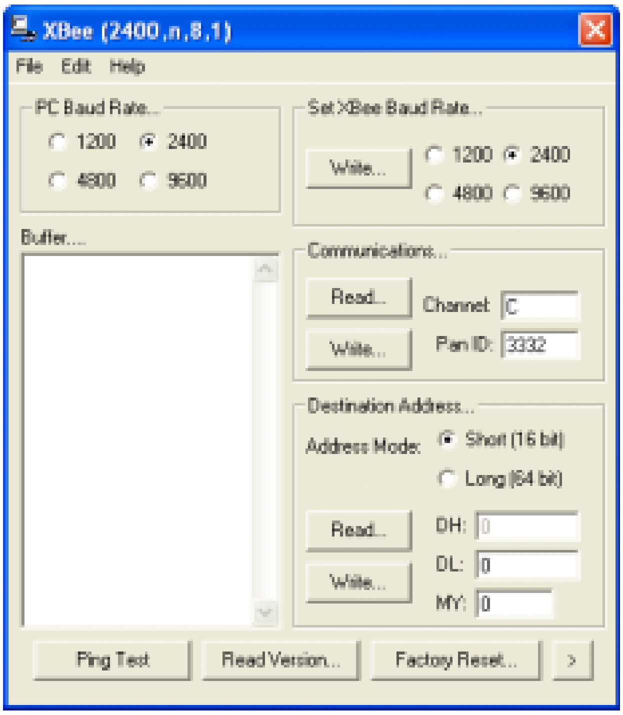
Remember NEVER insert both chips into the same board!

Board A Configuration	**Board B Configuration**
MAX3232CPE chip fitted.	PICAXE chip fitted.
Jumper 1 ‘top’ position.	Jumper 1 ‘bottom’ position.
Jumper 2 fitted.	Jumper 2 fitted.
Jumper 3 ‘bottom’ position.	Jumper 3 ‘bottom’ position.
Jumper 4 not fitted.	Jumper 4 not fitted.

The purpose of this procedure is to reset and then configure the two XBee modules so that they can communicate. It does not use the PICAXE chip (see test 2).

1. Insert one XBee module into board A.
2. Connect board A to the PICAXE download cable.
3. Run the Programming Editor software. Ensure the correct serial port is selected (View>Options>Serial Port menu).
4. Select the PICAXE>Wizard>AXE210 PICAXE Connect>XBee Setup menu.
5. Make sure the PC baud rate option matches the current XBee setting. For new modules this will be 9600.
6. Click the ‘Read Version’ button. This will confirm the module is correctly powered and connected.
7. If not using a new module, click the ‘Factory Reset’ button. This will reset all current settings of the XBee module, and also reset the baud rate to 9600.
8. In the ‘Set XBee baud rate’ option select 2400. Click ‘Write’ to configure the XBee to always use a baud rate of 2400. The ‘PC baud rate’ option will now also automatically change to 2400.
9. Click the ‘Read Version’ button again. This will confirm the module is correctly operating at the new baud rate.
10. Remove the XBee module and put into board B.
11. Repeat steps 5-9 for the second XBee module, so that it is also working at a baud rate of 2400.

Both modules are now set in the default configuration, at a baud rate of 2400. In this condition they will respond to any XBee module (ie they are address independent). We will look at how to use unique addresses next time. The boards are now ready for Communications Test 1.



XBee (2400,n,8,1)

PC Baud Rate... ○ 1200 ● 2400 ○ 4800 ○ 9600

Set XBee Baud Rate... ○ 1200 ● 2400 ○ 4800 ○ 9600

Channel: C

Pan ID: 3332

Address Mode: ● Short (16 bit) ○ Long (64 bit)

DH: 0

DL: 0

MY: 0

Communications Test 1 – Ping Test

Board A Configuration	**Board B Configuration**
MAX3232CPE chip fitted.	PICAXE chip fitted.
Jumper 1 ‘top’ position.	Jumper 1 ‘bottom’ position.
Jumper 2 fitted.	Jumper 2 fitted.
Jumper 3 ‘bottom’ position.	Jumper 3 ‘bottom’ position.
Jumper 4 not fitted.	Jumper 4 not fitted.

To test communication between the modules we first perform a ‘ping test’. In this test the computer sends serial data to the XBee module on board A. Board A then transmits the data over the wireless link. This data is received by board B and immediately ‘pinged’ (echoed) back to module A, which in turn transmits the received data back up the serial cable to the computer. The PICAXE chip is not used, as is it bypassed via jumper link 1 on board B. This jumper link connects the RX and TX lines of the XBee module together, so that anything received by the XBee module is instantly transmitted back as an ‘echo’.

1. Connect board A to the PICAXE download cable.
2. Place board B approximately 1m away from board A.
3. Run the Programming Editor software. Ensure the correct serial port is selected (View>Options>Serial Port menu).
4. Select the PICAXE>Wizard>AXE210 PICAXE Connect>XBee Setup menu.
5. Make sure the PC baud rate option matches the current XBee setting, which should be 2400.
6. Click the ‘Ping Test’ button. If all boards are configured correctly and operating a text string will be transmitted by board A, echoed by board B, and then displayed on the computer screen. The RSSI LED on both boards should light whilst they are communicating.

Communications Test 2 – PICAXE to PC Test

Board A Configuration	**Board B Configuration**
MAX3232CPE chip fitted.	PICAXE chip fitted.
Jumper 1 ‘top’ position.	Jumper 1 ‘top’ position.
Jumper 2 fitted.	Jumper 2 fitted.
Jumper 3 ‘bottom’ position.	Jumper 3 ‘bottom’ position.
Jumper 4 not fitted.	Jumper 4 not fitted.

If you have just performed test 1 remember to move the ‘Jumper 1’ position on board B.

In this second test, data is transmitted from the PICAXE chip on board B. The data is received by board A, which then displays the data on the computer screen. This tests that the PICAXE chip can transmit data over the wireless link.

1. Connect the PICAXE download cable to board B.
2. Download the following program into the PICAXE chip.

```
init:  high 7
       pause 100
main:  serout 7, T2400, (“Value =”, #b1,CR,LF)
       let b1 = b1 + 1
       pause 500
       goto main
```

3. Remove the download cable from board B and connect to board A.
4. Click the PICAXE>Terminal menu and make sure the baud rate is set to 2400.
5. The data being transmitted by board B should now be displayed in the terminal Window.

XBee Addressing

The previous two tests used the XBee module in its default state, where it will broadcast to, and receive data from, any other XBee module. However one of the main advantages of the XBee modules is that they can be configured to communicate with unique devices – e.g. two modules can be setup so they will only 'talk' to each other, ignoring information from any other module. So it is also useful to study a couple of sample projects where the modules are configured to only talk to each other.

Before going any further it is necessary to define some XBee terms. It is also strongly recommended that the XBee datasheet (available from www.maxstream.net) is studied.

Broadcast Channel <CH>
The XBee modules can communicate on different 'broadcast channels'. All XBee modules in the same network must be on the same channel. By placing different networks on different channels, interference can be reduced. By changing channel you may also reduce interference from, for instance, a home computer WiFi setup.

Baud Rate <BD>
To communicate, all XBee units must be operating at the same baud rate. For PICAXE setups it is recommended to use a baud rate setting of 2400, as this is compatible with all PICAXE chips. Although this is fairly slow by modern standards, it is quite adequate for most PICAXE projects.

Network Group <ID>
The XBee modules can be configured to work in 'Network Groups' (referred to as 'PAN ID' in the XBee documentation). Each module will only respond to devices in its own group. The group is defined by a 16-bit number.

Serial Number <SH> <SL>
Each XBee unit has a unique 64-bit serial number (aka 'long address'). This serial number is factory loaded and cannot be changed. When transmitting data you can choose to address the data to a unique device, identified by its serial number. The serial number is printed on the bottom of the XBee module.

Nickname <MY>
Each XBee unit can also be set with a user configurable 16-bit 'my nickname' (aka 'short address'). The advantage of the 'nickname' over the serial number is that it uses less memory in a PICAXE program. It also allows XBee units to be easily interchanged within an existing system by re-programming a replacement module (which would have a different serial number) with the existing nickname.

Summary for successful communication between XBee units:

1) All modules that wish to communicate must use the same baud rate.
2) All modules that wish to communicate must be on the same broadcast channel.
3) All modules that wish to communicate must be in the same network group.
4) Each XBee module can be configured to transmit to:
   - Any module
   - Module(s) with a specific 'nickname'
   - A unique module with a specific 'serial number'
5) Each XBee module can be configured to receive data:
   - From any module
   - Only when its 'nickname' is used
   - Only when its 'serial number' is used

Project Example 1 – Temperature and Light Sensors

In this first example project a DS18B20 temperature sensor and a Light Dependant Resistor are connected to the PICAXE-18X chip. The data from these sensors is then transmitted to a second XBee unit that is connected to a computer.Unique 'nickname' addressing will be used, so the data is only shared by these two modules. In this case the nickname of module 1 will be set to '1234' and the nickname of module 2 to '4321'.

The first task is to configure the two XBee modules with the correct settings using the Wizard in the PICAXE Programming Editor software. To do this insert the MAX3232 chip into the AXE210 connect board (remove any PICAXE chip) and make sure jumper J1 is in the 'top' position. Then click the PICAXE>Wizards>AXE210 Connect>XBee Setup menu.

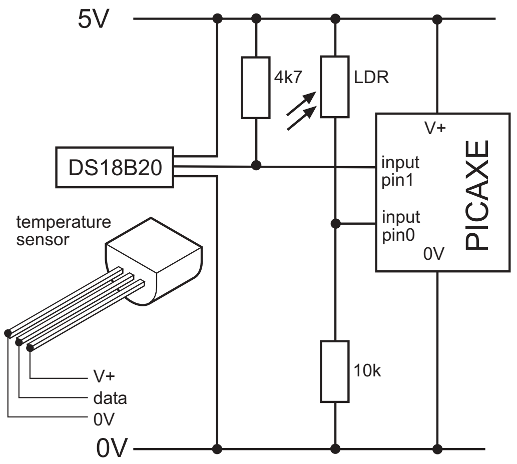
The settings required by each module are shown below:

		XBee (Board A)	**XBee (Board B)**
<BD>	Baud Rate	2400	2400
<CH>	Broadcast Channel	C	C
<ID>	Network Group (PAN ID)	3332	3332
<DL>	Destination nickname	4321	1234
<MY>	My short nickname	1234	4321

Note that all the settings are the same, apart from the 'my nickname' and 'destination nickname' values which are swapped between the two modules so that they can correctly identify each other.

After programming the units (by pressing the 'Write' button in each section) it is a good idea to do a 'Ping Test' to check the two units are communicating correctly with each other. Details of how to do this test are given in the initial setup procedure.

On one AXE210 connect board (board A) insert a PICAXE chip and then connect a LDR light sensor and a DS18B20 temperature sensor to the PICAXE input pins (at the top of the board) as shown below. The LDR is connected to input 0 and the temperature sensor to input 1.

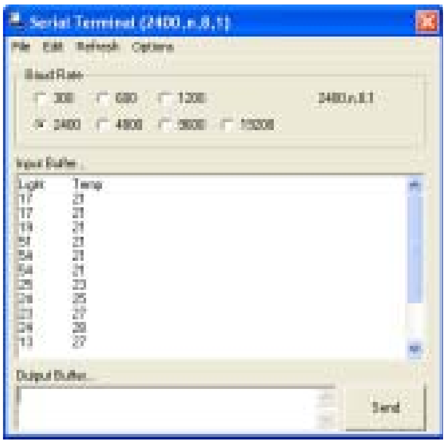


Connect board A to the computer and download the following program into the PICAXE chip. It reads the light and temperature values every second and then transmits them via the XBee unit.

```
symbol TAB = 9

init:       high B.7
            pause 100
            serout B.7, T2400, (“Light”,TAB,”Temp”,CR,LF)

main:       readadc C.0,b0
            readtemp C.1,b1
            serout B.7, T2400, ( #b0,TAB,#b1,CR,LF)
            pause 1000
            goto main
```



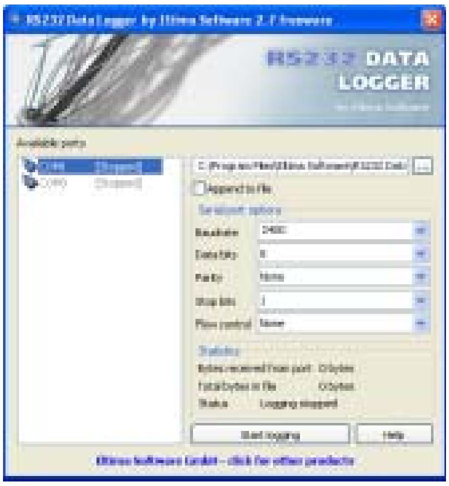
```
Light    Temp
17       21
17       21
19       21
51       21
54       21
54       21
25       23
28       25
23       27
24       28
13       27
```

In this program the ‘init’ section sets the serial pin high and then waits for 100ms. This gives the XBee time to ‘reset and wake up’. The main loop then reads the light value (readadc on input0) and temperature value (readtemp on input1) and transmits the data every second.

Disconnect board A (PICAXE) from the computer and then reconnect board B (fitted with a MAX3232) to the computer. Now click the PICAXE>Terminal menu and make sure the baud rate is set to 2400. The data being transmitted by board 1 should now be displayed in two columns in the Terminal window. That’s it – data is being transmitted wirelessly!

Data logging Software

PICAXE users often ask for simple serial datalogging software to allow recordings from a PICAXE project like this to be stored to a computer file, so that the data can be later analysed in, for instance, an Excel graph.



Baudrate: 2400
Data bits: 8
Parity: None
Stop bits: 1
Flow control: None

Bytes received from port: 0 bytes
Total bytes in file: 0 bytes
Status: Logging stopped

One free piece of software to do this is ‘RS232 Data Logger’ from Eltima Software (www.eltima.com). This software is very simple to use. Just highlight the serial COM port, enter the filename, enter the serial port settings (for PICAXE projects use 2400-8-None-1-None) and click ‘Start logging ’!

Once the logging is complete the file can be opened in, for example, Excel for analysis. To open this type of file in Excel simply click the File>Open menu and then select files of type:
‘Text Files (*.prn, *.csv, *.txt)’

Project 2 – Light and temperature warning LEDs!

This second project uses PICAXE to PICAXE communication. The hardware on board A will stay the same, but board B will now be fitted with a PICAXE chip and two output LEDs (on outputs 0 and 1) so that it can indicate the state of the temperature and light readings from board A. Remember to remove the MAX3232 chip!

The first task is to download a new PICAXE program into board A.

```
init:       high B.7
            pause 100
main:       readadc C.0,b0
            readtemp C.1,b1
            serout B.7, T2400, ($55,$55,b0, b1)
            pause 1000
            goto main
```

Note that it is no longer necessary to have the # before the variable names, because raw byte data is now being transmitted (rather than transmitting ASCII characters). Each transmission is also preceded by two $55 characters - this is a simple method to ensure that only valid data is accepted by the receiver.

It is now necessary to download a matching receiving program to board B.

```
main:       serin C.7, T2400, ($55,$55),b0, b1
            debug
test_LDR:   if b0 > 40 then LDR_high
LDR_low:    low B.0
            goto test_temp
LDR_high:   high B.0
test_temp:  if b1> 20 then temp_high
temp_low    low B.1
            goto main
temp_high:  high B.1
            goto main
```

This program waits until it receives valid data from the transmitter. The LEDs are then switched on and off accordingly. The (optional) debug command can be used to display (on the computer screen) which values are being transmitted during testing – this program uses ‘80’ as the light threshold value and ‘20’ as the temperature threshold, you may need to tweak these values slightly depending on how bright/hot your room is!

Reducing Power Use

If you are designing your own project board and XBee project you are likely to be considering batteries as your power source. Two simple improvements to the project board are immediately obvious. The first improvement is to power the whole circuit from a 3.3V battery pack which will then allow you to remove the need for the two (fairly inefficient) voltage regulators.

Secondly you may choose to enable 'external' sleep control of the XBee module. By adding Jumper 4 on the project board you can connect the 'sleep' pin of the XBee module to the PICAXE chip output 6. This means that with high and low commands you can switch the XBee module into low power sleep mode when they are not being used, hence saving power. For the XBee module to operate in this mode you must configure it to do so (the default is no external sleep control) – this is the carried out via the advanced programming options in the XBee Setup wizard (click the '>' button to see the advanced option).

Two pin controlled options are available here:

Option	**Current Drain**	**Wakeup Time**
Pin Doze	<50uA	2ms
Pin Hibernate	<10uA	13.2ms

Increasing Range

You should expect at least a 30m line-of-sight range from the standard XBee module. Note that signals can be affected by obstacles such as leaves on a tree.

A more expensive, more powerful version of the XBee module is also available as the 'XBee Pro' (with a transmit power of 100mW instead of the standard XBee 1mW).

However by using home-made antenna it is also possible to significantly increase the range of the standard XBee modules. For instance use of the Tritium 'Flatenna' (www.tritium.co.uk) cardboard/foil antenna showed a 6dBA increase in signal strength, so using this antenna at each end would provide a total 12dBA increase which equates to approximately a 4x distance increase! There are a large number of 'self-build' WiFi antenna examples on the internet – try a Google search for ideas!