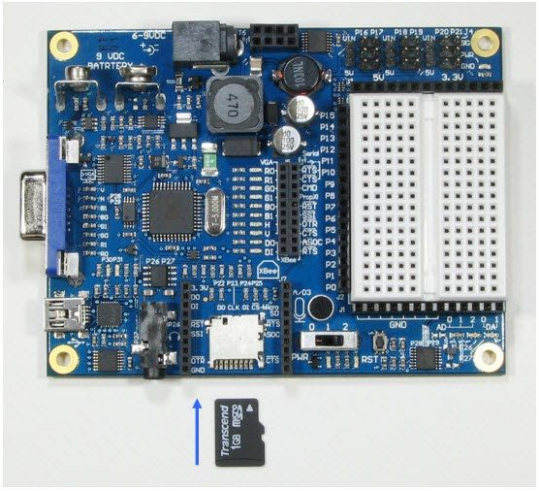
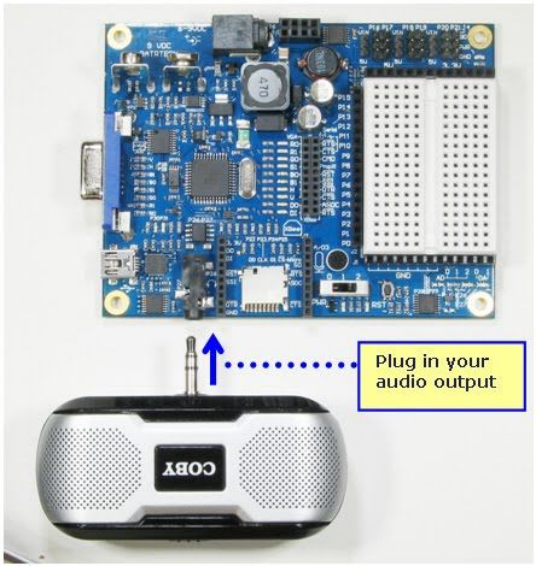
Web Site: www.parallax.com
Learn: learn.parallax.com
Forums: forums.parallax.com
Sales: sales@parallax.com
Technical: support@parallax.com

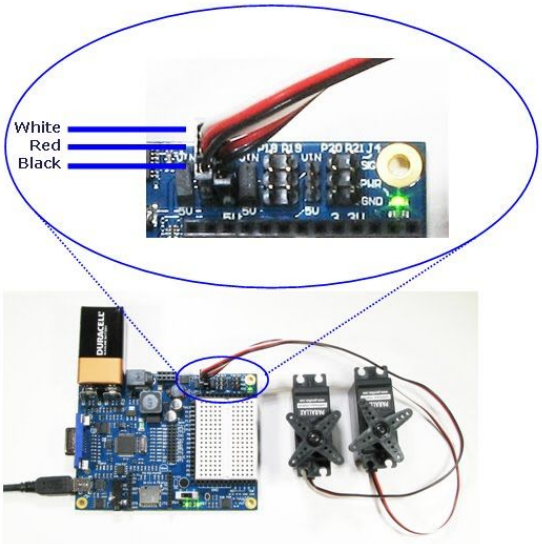
Office: (916) 624-8333
Fax: (916) 624-8003
Sales: (888) 512-1024
Tech Support: (888) 997-8267
EDU Hotline: (916) 625-6801

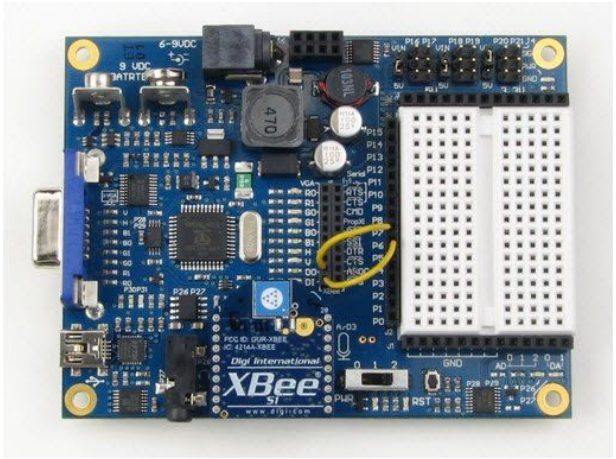
# Propeller Board of Education Project Builders

Add speech synthesis, servo control, wireless XBee communication, or microSD card data storage to your project. These Spin tutorials for the Propeller Board of Education will show you the basics.





# Tutorial 1) Speech Synthesis

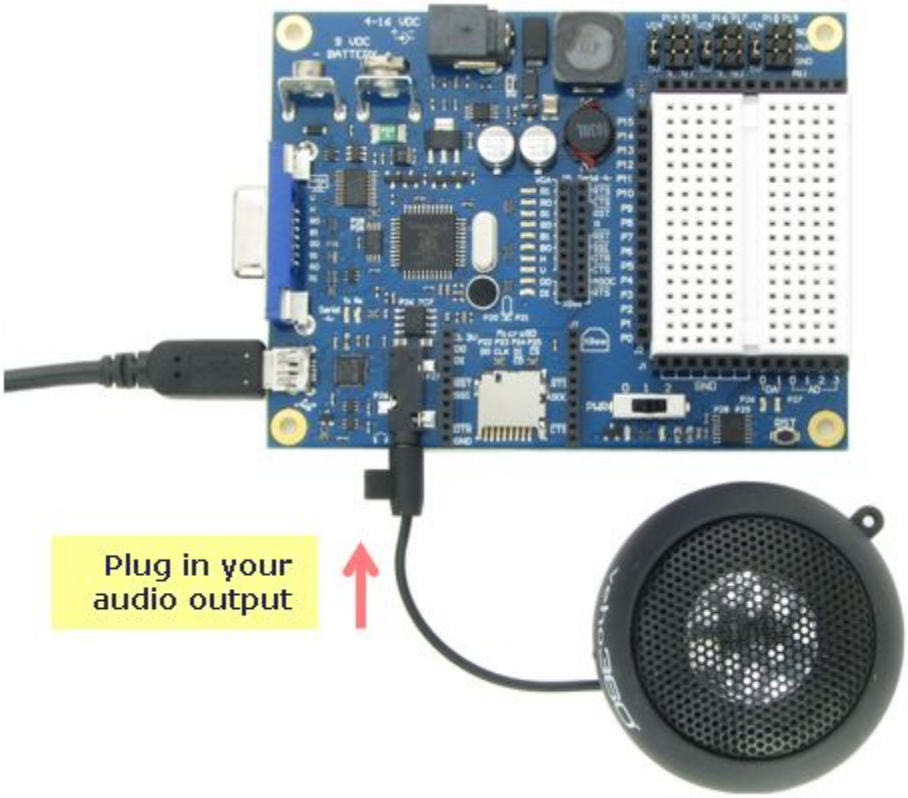
This Propeller Spin project is based on Phil Pilgrim's Phonemic Speech Synthesis (3rd installment, 7 Nov 2006). This is one of the many resources available from forums.parallax.com.

Phil's talk object has methods that you can pass addresses of strings to, and the strings represent phonemes. The result is something akin to "text to speech." The difference is that the object does not analyze actual text and figure out how it is supposed to sound. You have to pass strings (well, actually the addresses of strings) to the object's **`say`** method that describe each word's phonemes.

## Circuit

This is one of the Propeller Board of Education's convenience features—instant audio without having to wire up anything—the circuits that allow the Propeller to generate audio are built right into the board.

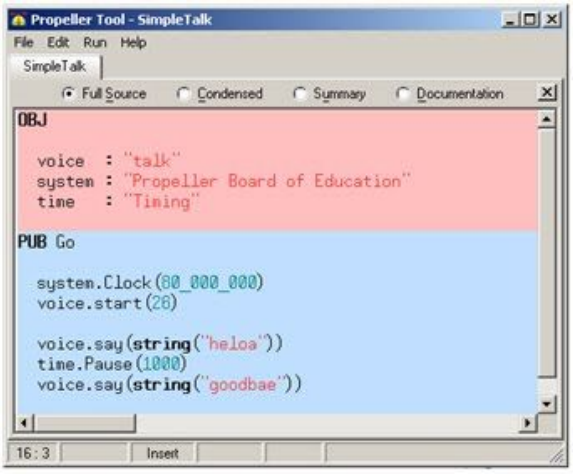
- ✓ Plug your audio device into the Propeller BOE's audio jack.
- ✓ If it requires batteries and on/off switch adjustment, do it now.



# Simple Talk

Simple Talk.spin passes strings of text that represent the phonemes in “hello” and “goodbye” to the talk object’s say method.

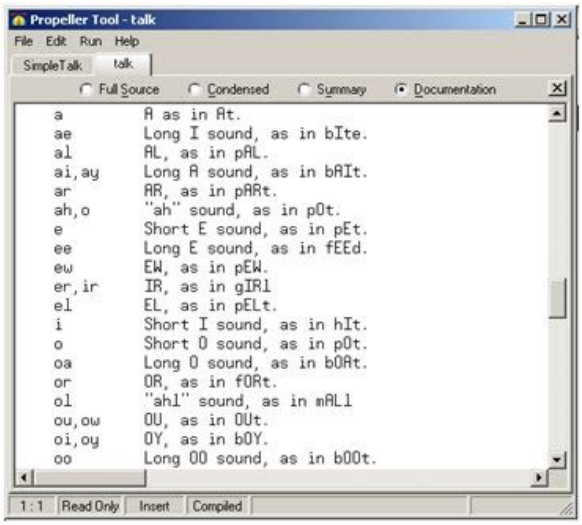
- ✓ **Navigate to the PropBOE_Speech_Synthesis_20120320 subfolder.**
- ✓ Run Simple Talk.spin and listen carefully.




**Make sure to View the Talk Object in Documentation Mode**

You will find a full explanation of the text representations of the Talk object’s phonemes.

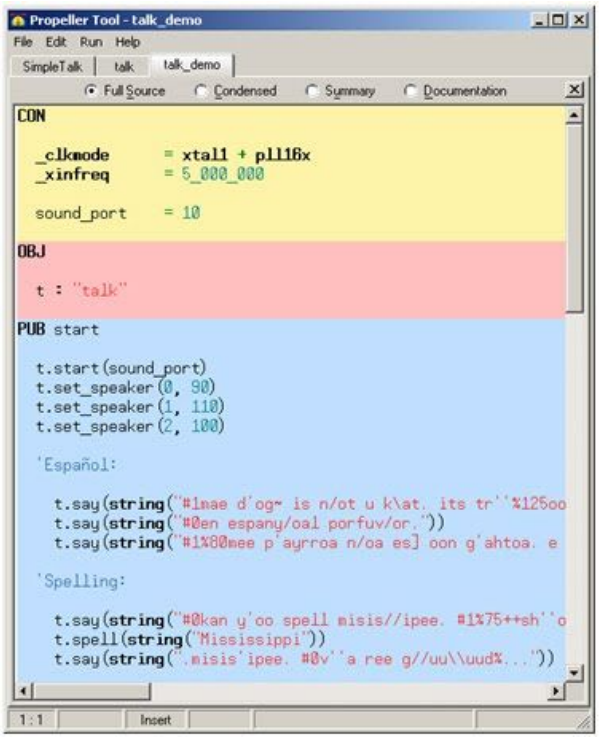
- ✓ Open the Talk object and make sure to examine its phonemes in Documentation mode.



## Your Turn

For fun, listen to all the vocalizations you can get from talk_demo.spin.

- ✓ **IMPORTANT:** Before running the program, you will have to change its sound port constant declaration from 10 to 26.
- ✓ Run the program and listen carefully.
- ✓ Start looking up the symbols the demo uses in the talk object's documentation, and you'll be well on your way to talking applications.




## Projects

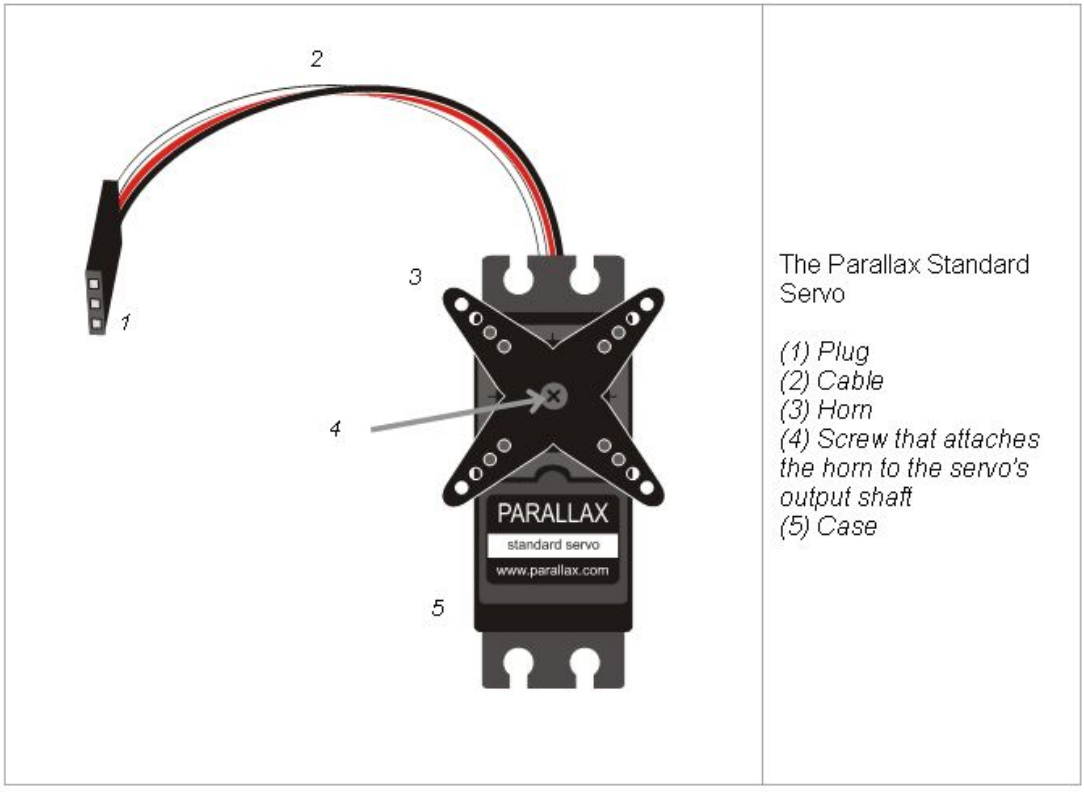
1. Modify Simple Talk.spin so that you can type the text representations of phonemes directly into the Parallax Serial Terminal. You will have to copy Parallax Serial Terminal to the same folder with the other objects, declare it in the OBJ block, and then find a method that returns the address of text that you type into the Parallax Serial Terminal's Transmit windowpane.
2. Add functionality that alerts when a pushbutton has been pressed.

# Tutorial 2) Servo Control

Standard hobby servos are useful in a variety of microcontroller projects.  Here are just a few examples:

- Animatronics
- Robotic armatures
- Humanoid robots
- Turret control for distance sensors
- Throttle control
- Rudder/flap/steering adjustments.

The reason they are useful is because microcontrollers send standard servos signals that cause them to turn that four pointed star in the figure (called a horn) to certain positions.  The servo then takes care of holding that position and resisting forces applied that might try to take its horn away from the position.  The servo's horn can then be attached to anything that needs a position held, like all those applications in the list above.




Continuous rotation servos are a little different.  They are controlled by signals that are similar to the ones for standard servos, but instead of holding positions, they keep turning at settable speeds and directions.

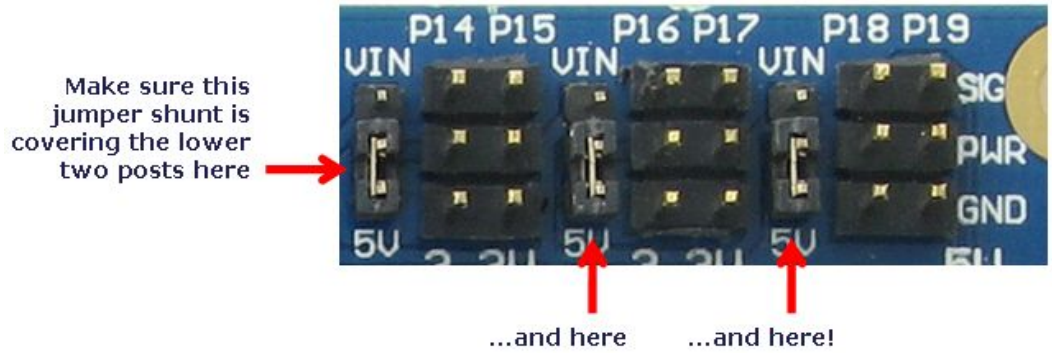
You can use continuous rotation servos in place of standard servos for this lesson. Just keep in mind that instead of turning to and holding certain positions, the continuous rotation servos will turn at certain speeds and in directions instead.

# Activity 1: Connect One Servo to Your Propeller BOE

## Circuit

The Propeller Board of Education has three power selection jumper posts to the left of each pair of servo ports. A conducting sleeve called a jumper shunt (or just jumper) is placed over either the lower to posts to set power for the servos to 5 V, or over the upper two posts to set servo power to VIN (unregulated input voltage). For the 5V setting, which is better for Parallax servos, the jumper sleeve should be covering the lower two posts, like in the picture.

- ✓ Check to make sure the jumper is covering the lower to power selection posts for all three pairs of servo ports. If it's covering the upper pair, just pull the jumper up off the posts, then slide it over the lower two posts.



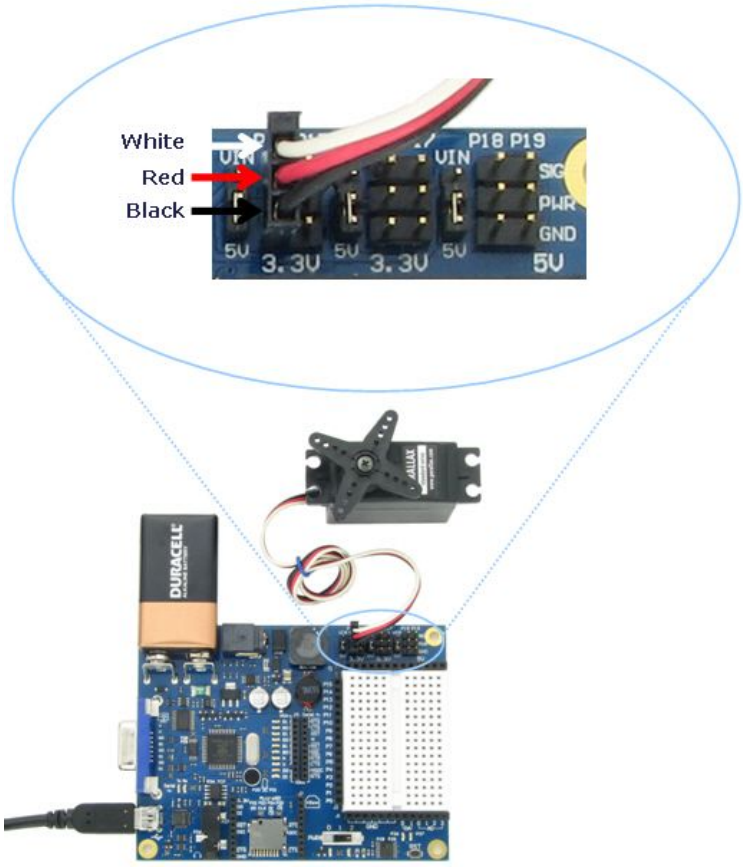

Let's start with just one servo, connected to the port labeled P14. Servos can end up pulling more current than the USB port can supply. So for these lessons, we want to either a 9 V battery, a 6 V battery pack, or a DC supply. If you are using a DC supply that plugs into the wall, a higher current rating, like 1000 mA (also expressed as 1 A) or more is recommended.

- ✓ Cut power to the board buy setting the PWR switch to 0.
- ✓ Connect your external power source (9 V battery, 6 V battery pack or 1000+ mA 6 to 9 V DC supply) to either the 9 V battery clip or 2.1 mm power jack.

With the power to the board is turned off, you can safely plug the servo into the port with the P14 label above it. Make sure it's plugged in the right way by matching the cable colors to the legend in the wiring picture.

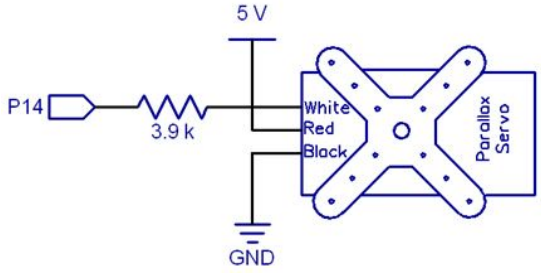
- ✓ Carefully connect your servo to the leftmost servo port (labeled P14). Make sure the order of your cable's color coded wires matches the picture.

## Wiring




## Schematic

On the Propeller BOE, I/O pins are connected to the pins labeled P14, P15, etc in the servo header group at the top-right of the board. These topmost pins get connected to the servo's white signal lead when you plug it in. There are 3.9 kΩ resistors between the Propeller I/O pins and the servo pins to make the Propeller chip's 3.3 V I/O pins compatible with certain sensors with 5 V outputs that are designed to be plugged into servo ports. So long as the 3.9 kΩ is between the 5 V sensor output and the 3.3 V I/O pin input, the pin will safely detect the sensor's signals.

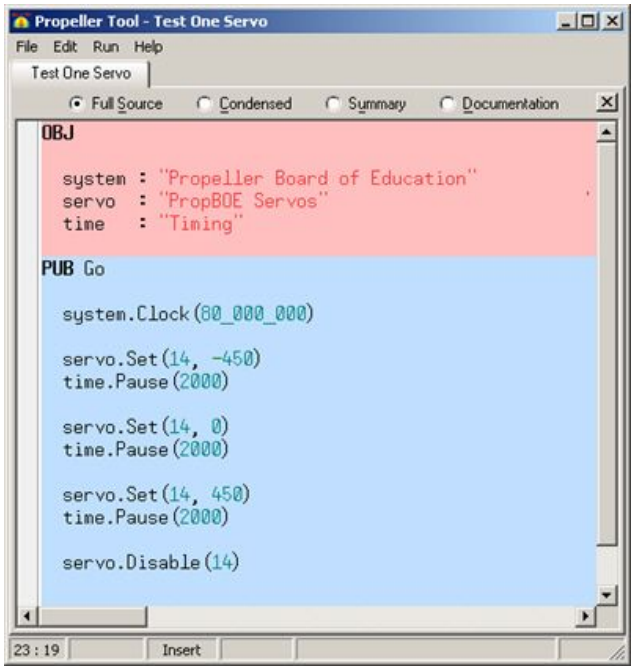


## Activity 2: Test Position Control with One Servo

Test One Servo.spin makes the servo position its horn at –45° for two seconds, then at 0° for two more seconds, and finally at 45° for two seconds. The servo will stay in the 45° position after the program ends, but it only "holds" that position for two seconds. After that, it will not resist twisting force to move its horn away from that position.


Your servo's zero degree position may not be as precise as the one in the picture, but it's easy to adjust. Simply unscrew the screw that holds the horn to the servo's output shaft. Then pull it up and off the shaft, and then push it back on in a position that makes it more precise. Of course, you may need some position that's much different from what's in the picture when you are using the servo to control armatures, flaps, etc. So, adjust the horn to suit your application.

- ✓ Set the PWR switch to 2.
- ✓ **Navigate to the PropBOE_Servos_20120323 subfolder.**
- ✓ Open Test One Servo.spin with the Propeller Tool software, and load it into the Propeller with F11.
- ✓ Watch the servo as it goes to and holds each position for three seconds.
- ✓ Try modifying your code for different values in the 800 to –800 range.




## How It Works

After nicknaming the PropBOE Servos object servo, a call to **`servo.Set`** positions a Parallax Standard Servo connected to a certain I/O pin to a certain number of tenths of degrees from center of its range of motion. For example, **`servo.Set(14, -450)`** positions the servo connected to the P14 port to the–45° position. After a 2 second pause, the **`servo.Set(14, 0)`** call makes the servo position its horn at the 0° position. After another two seconds, **`servo.Set(14, 450)`** positions the servo horn to 45°.

## Did You Know?

Each servo needs to receive a pulse (brief high signal) that instructs it where to position itself 50 times per second. The 0° signal is a pulse that stays high for 1500 µs. For every 10th of a degree you want to add to the position in the counterclockwise direction, add a microsecond to the pulse. Likewise, for every 10th of a degree you want to shave off the position (in the clockwise direction), subtract a microsecond from the pulse.

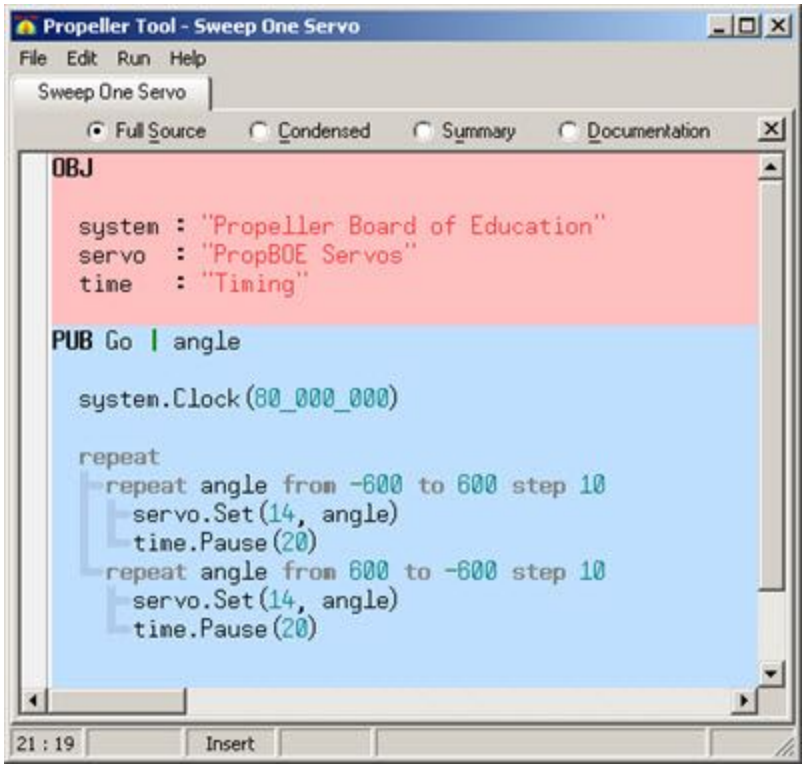
For example, to make the servo hold the 45° position, it needs to send the servo pulses that last 1500 µs + 450 µs = 1950 µs. To make the servo keep holding that position, the PropBOE Servos object has to keep sending those 1950 µs pulses 50 times per second.
The first call to a PropBOE Servos method that sets servo position causes it to automatically launch servo control code into another cog. That one cog will control up to 14 servos. Keep going through the pages in this lesson, and you'll see code examples that control two and three servos. Modifying the code for even more than that is just a matter of following the same pattern until you get to 14. At that point, if you need to control more servos, just make a copy of

the PropBOE Servos object (the file), declare it in your OBJ block, and give it a different nickname.

## Your Turn – Servo Sweeper

Instead of using numbers like –450, 0, and 450, you can use a variable, and the code can change that variable periodically.

- ✓ Save your code under a new name.
- ✓ Modify it to look like below.
- ✓ Load it into the propeller and watch as the servo sweeps back and forth
- ✓ Try increasing the **`step`** size from 10 to 20 in just one of the repeat loops to make it go one direction more quickly than it goes the other direction.




## Did You Know?

You can also use the PropBOE Servos object's **`StepSize`** method to make it turn gradually. Then, you can just give the servo a target position and an amount of time to get there. Like this:

```
''Sweep One Servo with StepSize.spin

OBJ

  system : "Propeller Board of Education"
  servo  : "PropBOE Servos"
  time   : "Timing"

PUB Go | angle

  system.Clock(80_000_000)

  servo.Set(14, -600)
  time.Pause(100)
  servo.StepSize(14, 10)

  repeat
    servo.Set(14, 600)
    time.Pause(2400)
    servo.Set(14, -600)
    time.Pause(2400)
```

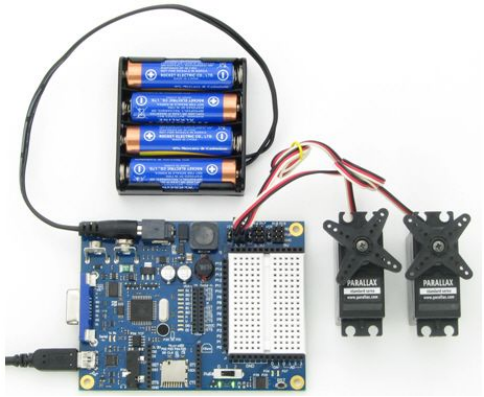
After the **`servo.StepSize(14, 10)`** call, the PropBOE Servos object will only allow the P14 servo to move by 1/10th of a degree every 50th of a second. So, the servo gradually makes its way to its target position, and you don't have to write repeat loops. However, if you want your servo to make it to its destination, your code will have to give it enough time to get there before using making it move to another position.

## Activity 3: Control Two Servos

Adding and controlling a second servo is pretty simple. Just connect the servo to another port, and add set calls to your code with updated pin parameters (that match the number of the port you plugged the servo into). When the PropBOE Servos object sees the new pin parameter in a call like **`servo.Set`**, it automatically adds it to the list of pins that it sends control signals.

### Hardware Setup

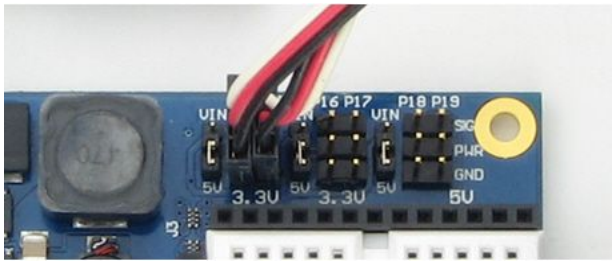
For this example, we'll just connect a second servo to the P15 port, which is just to the right of the P14 port.



## Wiring

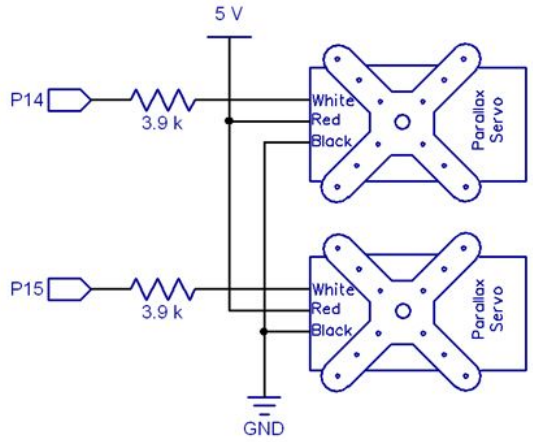
Here's a close-up of the servo port connection.

- ✓ Set the PWR switch to 0 before connecting the second servo.
- ✓ Make sure the P15 servo's white, red, black wire order is the same as P14.
- ✓ Set the PWR switch to 2.




## Circuit

Your second servo's red and black wires draw from the same 5V and GND supplies, but its white signal line is connected to Propeller I/O pin P15 instead of P14.



## Code Example

One quick and easy way to test the second servo connected to P15 would be to add calls like **`servo.Set(15, -450)`** to a copy of the Test One Servo.spin object.

- ✓ Open Test One Servo.spin and save it as Test Two Servos.spin.
- ✓ Modify it by adding the **`servo.Set(15, -450)`**, servo.Set(15, 0), and **`servo.Set(15, 450)`** as shown in the Test Two Servos.spin example below.
- ✓ Set the 3-position PWR switch to the 1 position (middle).
- ✓ Use F11 to load it into the Propeller chip.
- ✓ Set the 3-position PWR switch to the 2 position (far right).
- ✓ Verify that the P15 servo turns to 45° when the P14 servo turns to -45°. After two seconds, they should both turn to 0° for two seconds. Then, the P15 servo should turn to -45° and P14 should turn to 45°.

```
'' Test Two Servos.spin

OBJ

  system : "Propeller Board of Education"
  servo  : "PropBOE Servos"
  time   : "Timing"

PUB Go

  system.Clock(80_000_000)

  servo.Set(14, -450)
  servo.Set(15, 450)
  time.Pause(2000)

  servo.Set(14, 0)
  servo.Set(15, 0)
  time.Pause(2000)

  servo.Set(14, 450)
  servo.Set(15, -450)
  time.Pause(2000)

  servo.Disable(14)
  servo.Disable(15)
```

## Your Turn – Two Servo Sweeper

✓ Try adding **`servo.StepSize`** calls before the first **`servo.Set`** call to make your program gradually move the servos to each destination. See the last program example on the previous page if you need ideas on how to do it.

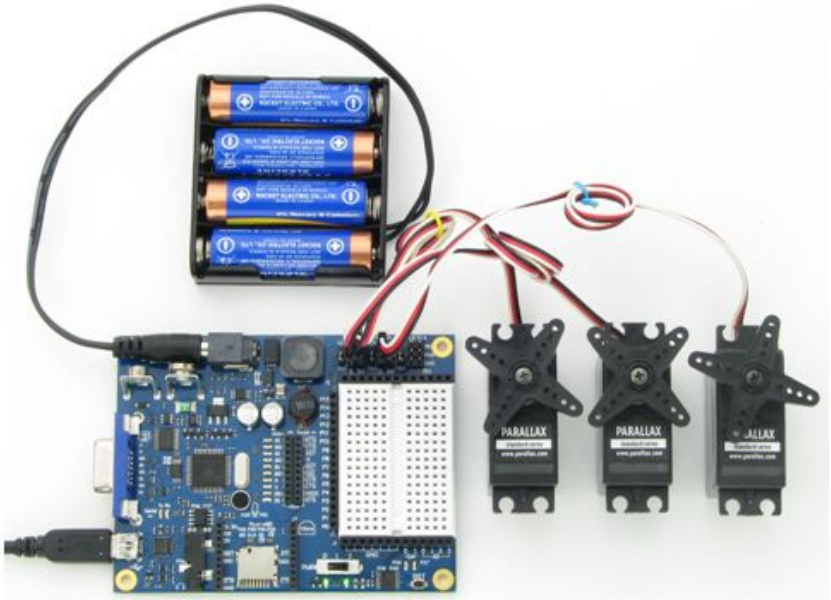
## Activity 4: Set and Sequence Groups of Servos

Simple calls to the **`Set`** method are well and good when you're controlling just one or two servos, but imagine doing that for an animatronics application with 10 servos. Instead of just one or two **`Set`** calls, you might end up having to write ten **`Set`** calls each time you want to update all the servo positions.

Fortunately, the PropBOE Servos object has methods for setting the positions of groups of servos with a single method call. It also has methods for setting up entire sequences of positions for groups of servos along with provisions for tasking another cog with the executing sequences. Used together, these are powerful tools that can greatly simplify coding for animatronics and humanoid robot applications.

### Hardware Setup

For these examples, we'll add a third servo.



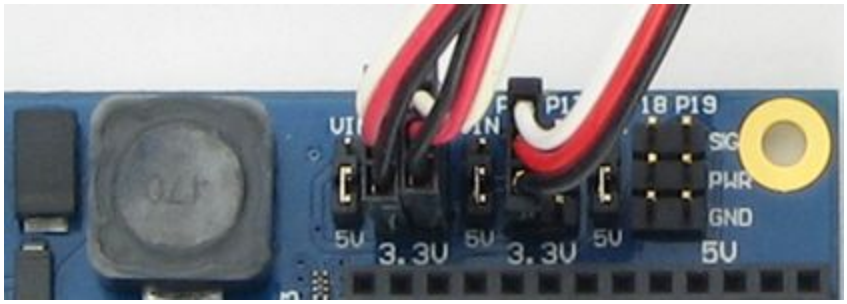

### Wiring

Connect a third servo to the P16 servo port. As before:

✓ Set the PWR switch to 0 before connecting the second servo.

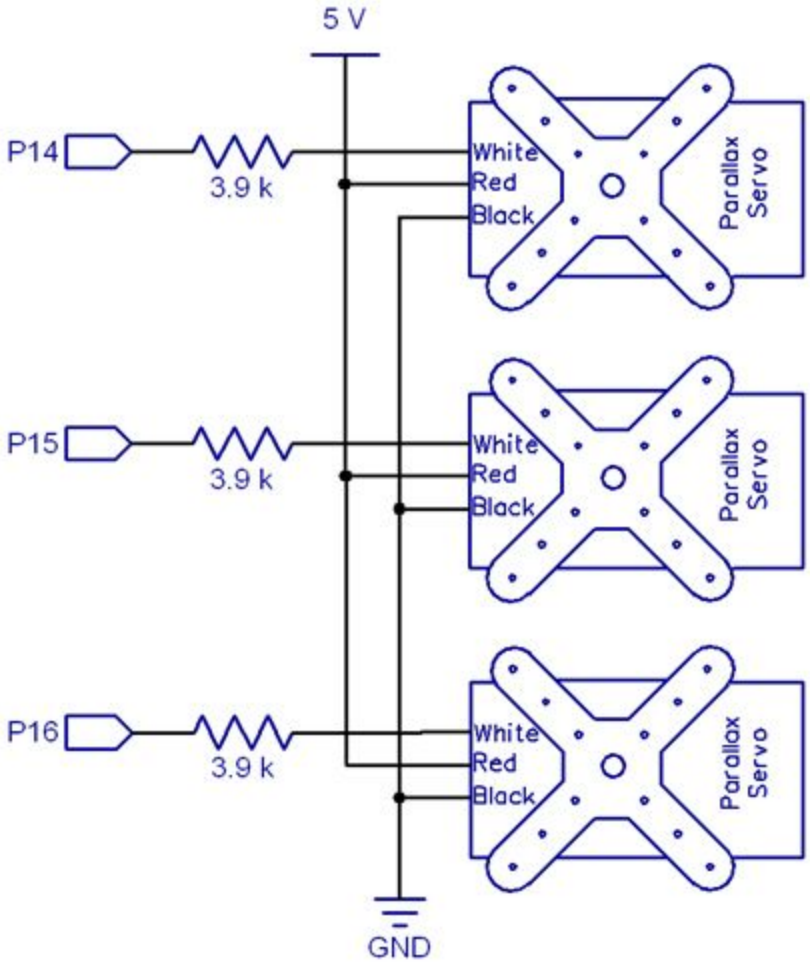
✓ Make sure the P16 servo's white, red, black wire order is the same as P14 and P15 – white on top, then read, then black on the bottom.

✓ Set the PWR switch to 2.




## Circuit

After plugging in the third servo, the schematic will look like this.



## Use SetList to Position Multiple Servos

The PropBOE Servos object has a **`SetList`** method that is designed to receive lists of values that are either stored in an array, or in a **`DAT`** block. The Set Three Servos at Once.spin example uses a **`DAT`** block. This **`DAT`** block has a list named pins, with three servo port values, and then -1 to indicate the end of the list. (The Propeller chip doesn't have any negative I/O pin values!)

The **`DAT`** block also has three sets of servo position values, labeled A, B, and C. The call to **`servo.SetList(@pins, @A)`** passes the PropBOE Servos object's SetList method the address of the pin list, and the address of position set A. The **`SetList`** method then takes care of the rest. It does this for two more position sets, B, and C. Let's try it:

- ✓ Open Set Three Servos at Once.spin.
- ✓ Use F11 to load it into the Propeller chip.
- ✓ Monitor the servos. The P14 servo should turn to 80°, then 0°, then -80°. At the same time, the P15 servo should turn to 0°, then -80°, then 80°. And, at the same time as all that, the P16 servo should turn to -80°, 80°, then 0°.

```
'' Set Three Servos at Once.spin

OBJ

  system : "Propeller Board of Education"
  servo  : "PropBOE Servos"
  time   : "Timing"

DAT

  pins long   14,   15,   16,   -1
  A    long  800,    0, -800
  B    long    0, -800,  800
  C    long -800,  800,    0

PUB Go

  system.Clock(80_000_000)

  servo.SetList(@pins, @A)
  time.Pause(2000)

  servo.SetList(@pins, @B)
  time.Pause(2000)

  servo.SetList(@pins, @C)
  time.Pause(2000)

  servo.DisableList(@pins)
```

## Did You Know?

The PropBOE Servos object remembers the most recent pin list. So, after **`servo.SetList(@pins, @A)`**, the program could have used **`servo.UpdateList(@B)`** and **`servo.UpdateList(@C)`** for the second and third position sets.

## Use Sequence for Multiple Position Lists

After you've got pin and position lists, you can add a sequence list. Take a look at the last two lines in the **`DAT`** block below. It's a sequence that can be used to make the servos turn to the A position list for 2 seconds, then the B position list for 2 seconds, and so on…

The sequence list has a few rules: The first entry in the sequence list has to be the address of the sequence list's name. The second entry has to be the address of the pins list. Then, the sequence is the address of a position list, followed by the number of milliseconds the servos should hold that set of positions. Add as many position set addresses + hold times as your program needs, and then end the list with -1. To make the PropBOE Servos object run the sequence list, just pass the address of the sequence list's name to the **`Sequence`** method, like **`servo.Sequence(@seq)`**.

- ✓ Open Set Three Servo Sequence.spin.
- ✓ Use F11 to load it into the Propeller chip.
- ✓ Monitor the servos. They should execute the same sequence as three Servos at Once.spin.

```
'' Three Servo Sequence.spin

OBJ

  system : "Propeller Board of Education"
  servo  : "PropBOE Servos"
  time   : "Timing"

DAT

  pins long   14,   15,   16,   -1
  A    long  800,    0, -800
  B    long    0, -800,  800
  C    long -800,  800,    0

  seq  long  @seq, @pins
       long  @A, 2000, @B, 2000, @C, 2000, -1

PUB Go

  system.Clock(80_000_000)

  servo.Sequence(@seq)

  servo.DisableList(@pins)
```

## Your Turn

Sequences are really easy to work with. Here's an example that makes the servos backtrack to position set B, then position set A after the A, B, C sequence you've already seen.

- ✓ Start with the last line in the DAT block: `@A, 2000, @B, 2000, @C, 2000, -1`
- ✓ Change it to `@A, 2000, @B, 2000, @C, 2000, @B, 2000, @A, 2000, -1`.
- ✓ Run the modified program and verify that after third position in the sequence, it backtracks through the first two.

## Execute Sequences with Another Cog

Maybe your application needs to monitor sensors as it's ticking through a sequence of servo positions. No problem. Instead of calling `servo.Sequence`, call `servo.StartSequence`. In this last example, it calls `StartSequence`, and then displays counting in the Parallax Serial Terminal while another cog makes the servos move.

- ✓ Run your Parallax Serial Terminal and be ready to click it's Enable button.
- ✓ Open the Set Three Servo Sequence in Other Cog.spin example program
- ✓ Use F11 to load it into the Propeller chip.
- ✓ Click the Parallax Serial Terminal's Enable button.
- ✓ Monitor the servos and the Parallax Serial Terminal. The servos should go through their sequence while the Propeller sends numbers to the Parallax Serial Terminal.

```
'' Three Servo Sequence in Other Cog.spin

OBJ

  system : "Propeller Board of Education"
  servo  : "PropBOE Servos"
  time   : "Timing"
  pst    : "Parallax Serial Terminal Plus"

DAT

  pins long   14,   15,   16,   -1
  A    long  800,    0, -800
  B    long    0, -800,  800
  C    long -800,  800,    0

  seq  long  @seq, @pins
       long  @A, 2000, @B, 2000, @C, 2000, -1

PUB Go | val

  system.Clock(80_000_000)

  ' Launch sequence into another cog.
  servo.StartSequence(@seq)

  ' This cog moves on to display counting.
  repeat val from 0 to 59
    pst.Dec(val)
    pst.NewLine
    time.Pause(100)
  pst.Str(string("All done!"))

  servo.DisableList(@pins)
```

## Did You Know?

In Three Servo Sequence.spin, the program had to wait at the **`servo.Sequence(@seq)`** call for 6 seconds while the cog 0 completed the sequence. In Three Servo Sequence in Other Cog.spin program, **`servo.StartSequence(@seq)`** launched the sequence code into cog 1, allowing cog 0 to immediately move on to the repeat loop where it started counting. At that point, the Parallax Serial Terminal object launched serial communication code into cog 2 for communicating with your computer (and the Parallax Serial Terminal). So, at that point, your application is running three of the Propeller chip's eight processors.

# Tutorial 3) Wireless XBee Communication

XBee modules are versatile radios that work great for communication between microcontrollers, other microcontrollers, and the PC.

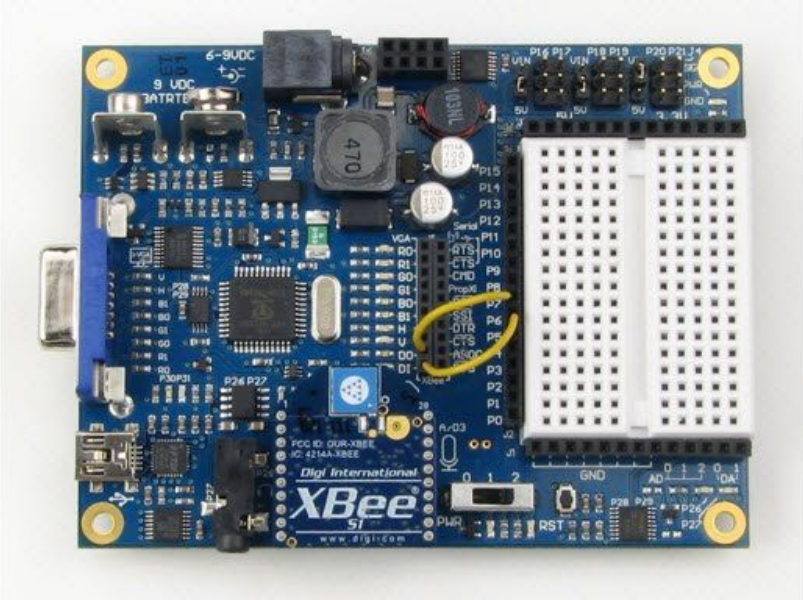
Available as a PDF, the "Getting Started with XBee RF Modules" tutorial by Martin Hebel includes both BASIC Stamp and Propeller example programs. Aside from being highly informative, the PDF textbook is a free download from here, and you may also download the source code from here.

## Circuit

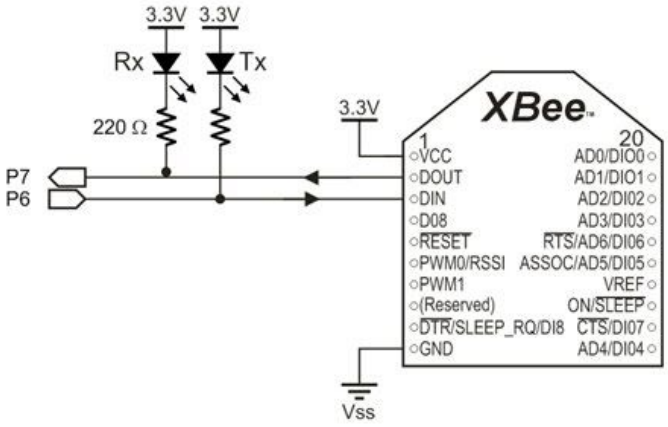
All you have to do to prepare your Propeller BOE to communicate wirelessly with XBee radios is plug in the XBee module into its socket on the Propeller BOE, and connect a couple of jumper wires. One jumper should connect the XBee DO socket to a Propeller I/O pin, and the other should connect XBee DI to an I/O pin.

- ✓ Insert the XBee into its socket on the Propeller BOE as shown below.
- ✓ Connect P7 to XBee DO with a jumper wire.
- ✓ Connect P6 to XBee DI with another jumper wire.

## Wiring



## Schematic

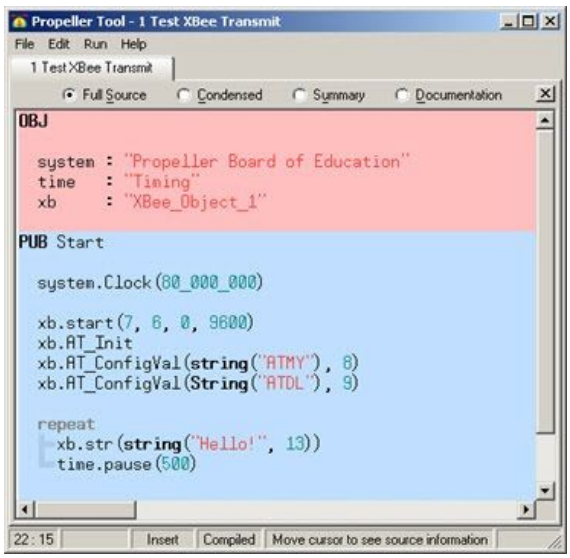



The LEDs in this circuit are built into the PropBOE. All you have to do is connect each I/O pin to the correct XBee socket with a jumper wire.

## Code Examples

There are two code examples to get you started with XBee modules. One sends messages, and the other receives them. You will need a pair of Propeller BOE boards and XBee modules to run this activity.

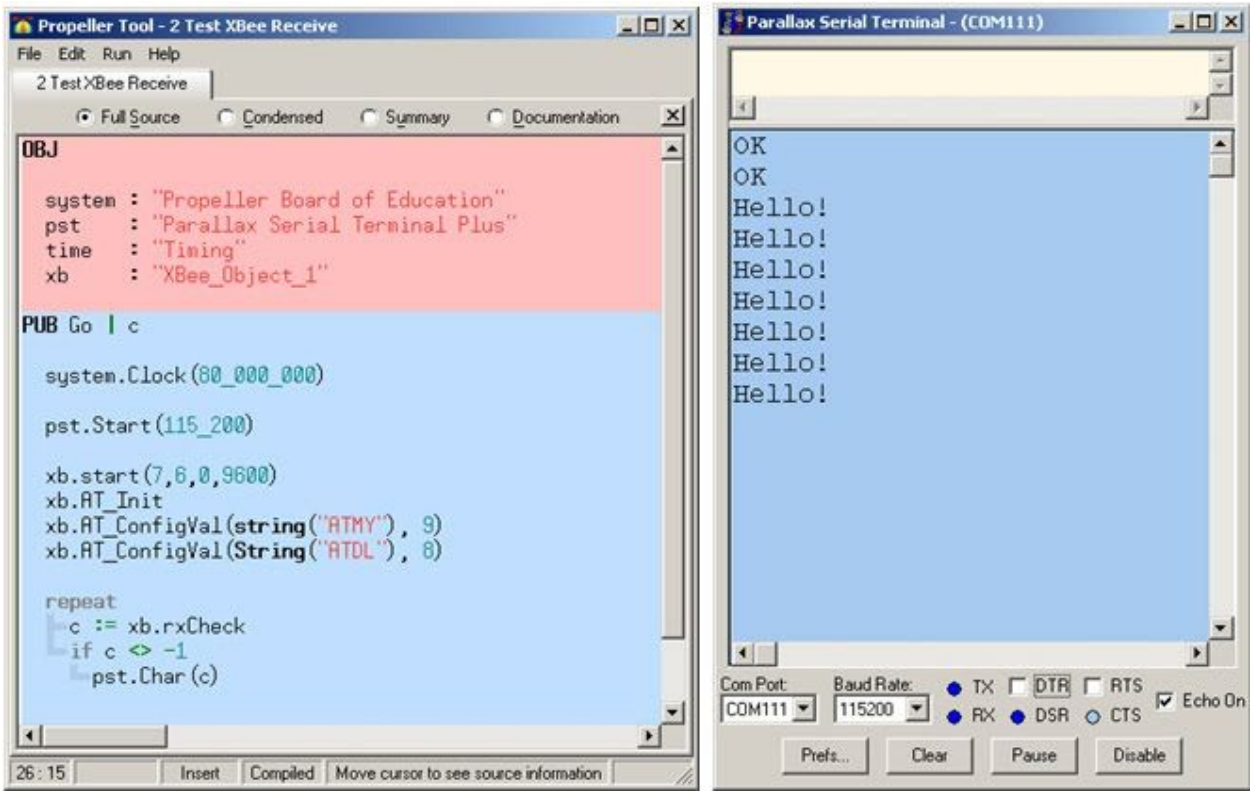
- ✓ **Navigate to the Wireless_XBee_Communication folder**
- ✓ Remember that you have to open programs from within the folder.
- ✓ Open, read, and run “1 Test XBee Transmit.spin”.



✓ Open “2 Test XBee Receive.spin” and use F11 to load it into the Propeller chip.

✓ While the program is loading, click the Parallax Serial Terminal’s Enable button.

It should take five or so seconds for the XBees to sync up. Be patient, the hello messages should start up.




## Wireless Programming with XBee

This project is based on the application note AN007: Soft-loading an Application Image into the Propeller P8X32A via XBee Transceivers.

With a combination of an object running code in one of the Propeller Chip's Cogs and the Propeller XBee Loader software, you can wirelessly program and reprogram your PropBOE's Propeller chip.

### Recommended Equipment

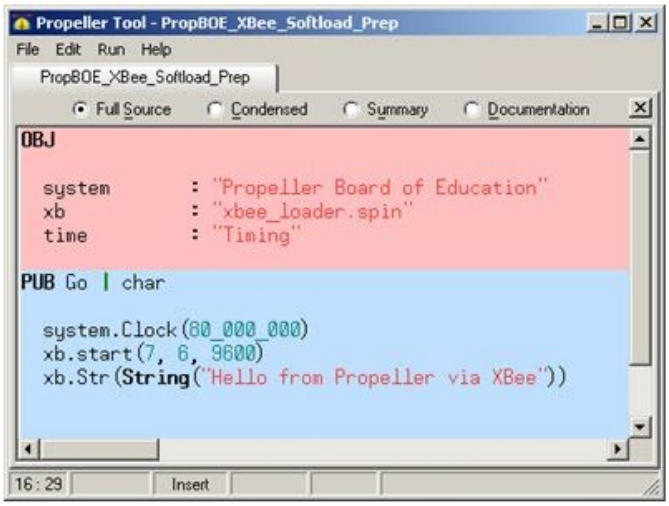
(2) XBee Modules
(1) XBee USB Adapter
(1) Propeller Board of Education

## Hardware Setup

- ✓ Follow the instructions in the XBee USB Adapter product documentation to solder together your XBee USB adapter.
- ✓ Socket one of your XBee modules into the adapter.
- ✓ If you have not already done so, connect your XBee to your Propeller BOE. Use the wiring and schematic from Wireless XBee Communication (above).

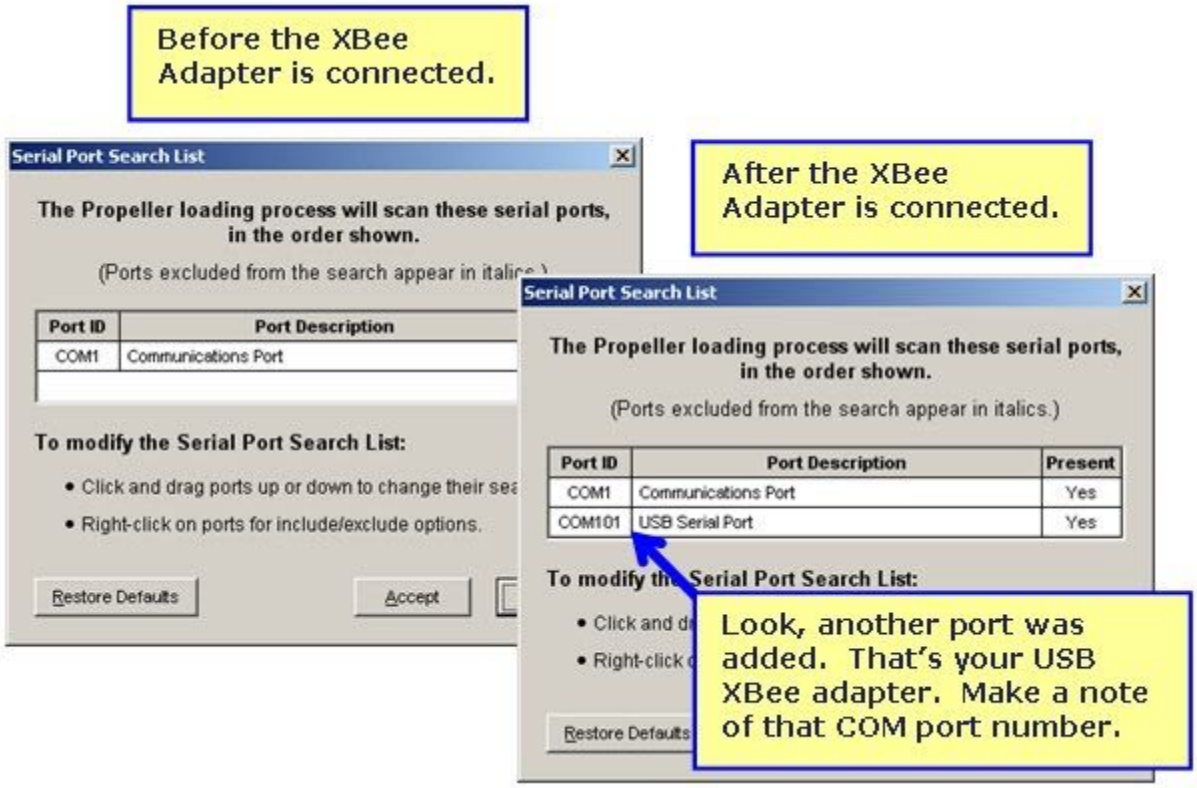
## Software Setup

- ✓ **Open the Propeller_SoftLoad_with_XBee_2011_11_02 subfolder** and then open PropBOE_XBee_Softload_Prep.spin with the Propeller Tool software.
- ✓ Use F11 to load it into the PropBOE's Propeller and EEPROM.



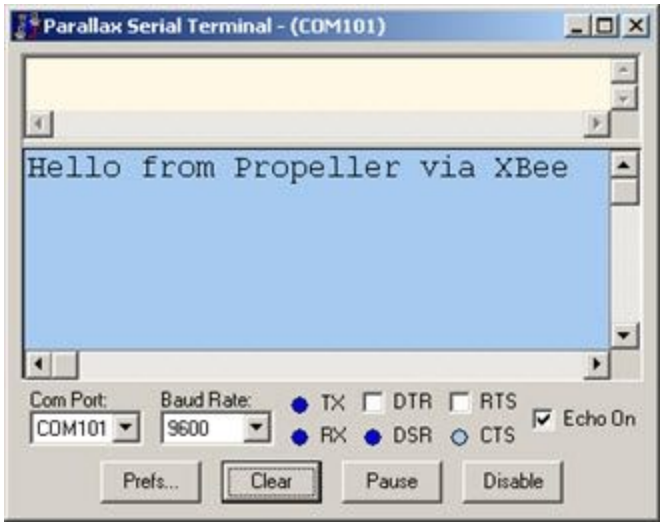

Before running the XBee programming software, let's be sure which COM port the XBee is connected to.

- ✓ In the Propeller Tool, click Edit and Select Preferences.
- ✓ Click the Operation Tab, and then the Edit Ports button.
- ✓ Watch carefully as you connect the XBee USB Adapter to your computer. Another COM port will appear in the list, and that's your XBee port. Make a note of the COM port number.




Before using the wireless connection to load a different .spin program into the Propeller chip, it's a good idea to test and make sure the serial communication link is working. Here's how:

- ✓ Make sure the power is connected to your Propeller BOE.
- ✓ Open the Parallax Serial Terminal.
- ✓ Set the COM port to the value you got from the Propeller Tool software's Serial Port Search List.
- ✓ Set the Baud Rate to 9600, and then click Enable.
- ✓ Press and release the Propeller BOE's rest button to restart the program. The "Hello..." message should appear in the Parallax Serial Terminal.
- ✓ Click the Parallax Serial Terminal's Disable button to make sure the port is open for the Propeller XBee Loader software.



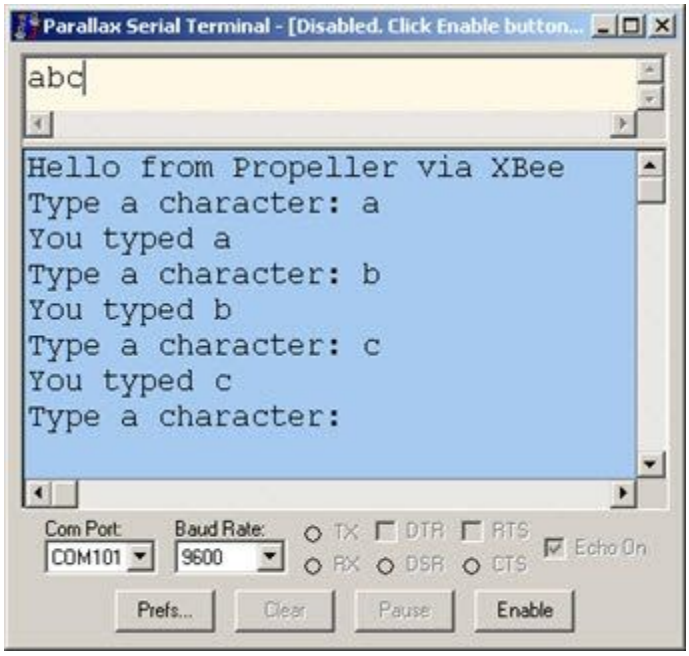

Now, let's use the Propeller XBee Loader tool to wirelessly load a different Spin program into the Propeller Chip.

- ✓ Find LoadPropXB.exe in your Propeller SoftLoad with XBee and run it.
- ✓ Set the COM port to the value you got from the Propeller Tool software's Serial Port Search List.
- ✓ Click the Scan button. The Propeller XBee Loader tool will report how many XBee's it found, and display a unit ID. (You can scroll through them to select which remote XBee you want to program.)
- ✓ Click the Choose button, and use it to open the PropBOE_XBee_Softload_Prep.spin program from your App Dev\Propeller SoftLoad with XBee folder.
- ✓ Click the Load ROM button to load the new program into your Propeller Tool.


With a new program running, your Parallax Serial Terminal will behave differently.

- ✓ Click the connect button on the Parallax Serial terminal. If you don't see any messages, press/release the reset button on your Propeller BOE.

✓ Click the Parallax Serial Terminal's Transmit windowpane, and type characters. The Propeller chip will wirelessly transmit back the character you typed.



# Tutorial 4) SD Card Data Storage

An SD card opens up lots of project options because it can store and retrieve large amounts of data. Especially with a FAT (file allocation table) driver, the same MicroSD card that you can use with a PC to create and modify files will work with the Propeller BOE. It's completely interchangeable.

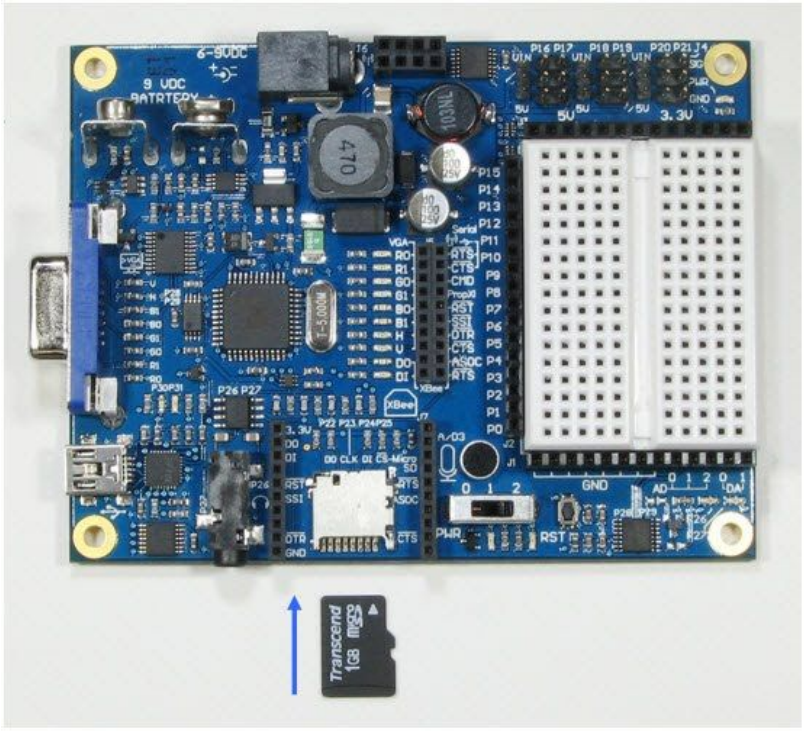
The PropBOE MicroSD object you'll use in this lesson makes it easy to create, open, read, modify and delete files on your MicroSD card. So, your applications can now work from data saved by a PC, or store data for later analysis by a PC. Or, the SD card can be used like a gigantic repository of values for your Propeller BOE application. Even better, you can store lots of different programs for the PropBOE on your SD card, and write code to decide which one to boot.

## Circuit

The PropBeller BOE has a built-in MicroSD card socket.

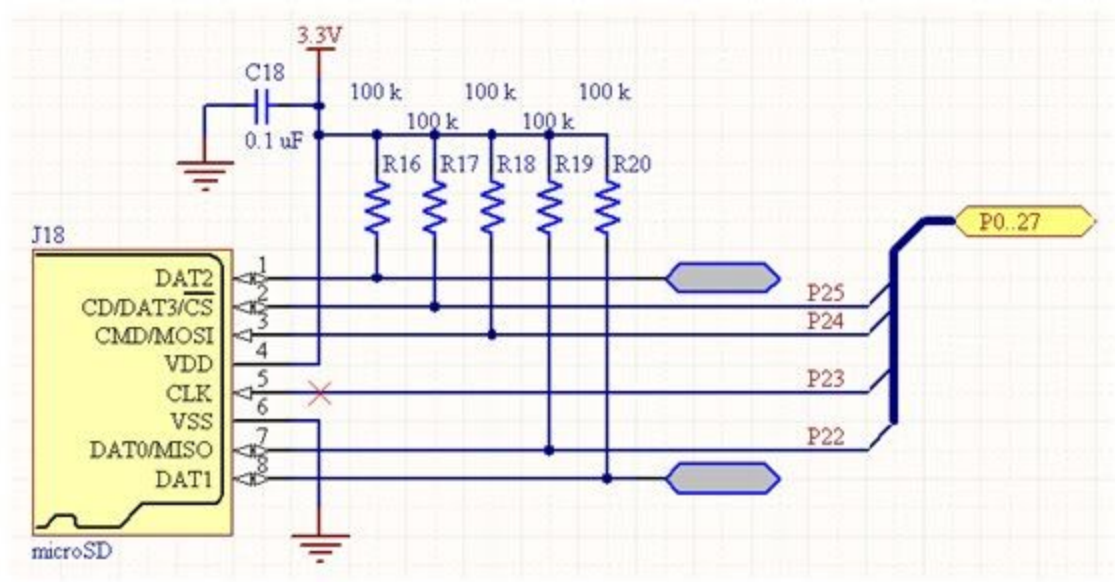
## Wiring

- ✓ Simply insert a MicroSD card into the PropBOE socket.



## Schematic

These connections are built into the Propeller BOE.




The resistors shown in the schematic are here:


## MicroSD Card Data Storage/Retrieval Examples

09_SD_Card_Storage.zip has example programs that demonstrate how to work with text files, store values in a spreadsheet format, in the Propeller chip's native format. It also has an

example that demonstrates how one program can boot another program that is stored on the MicroSD card.

- ✓ **Open the SD_Card_Storage folder.**
- ✓ Remember that you have to open programs from within the folder.
- ✓ Follow these sections below for a guided tour through each example program and how it works.

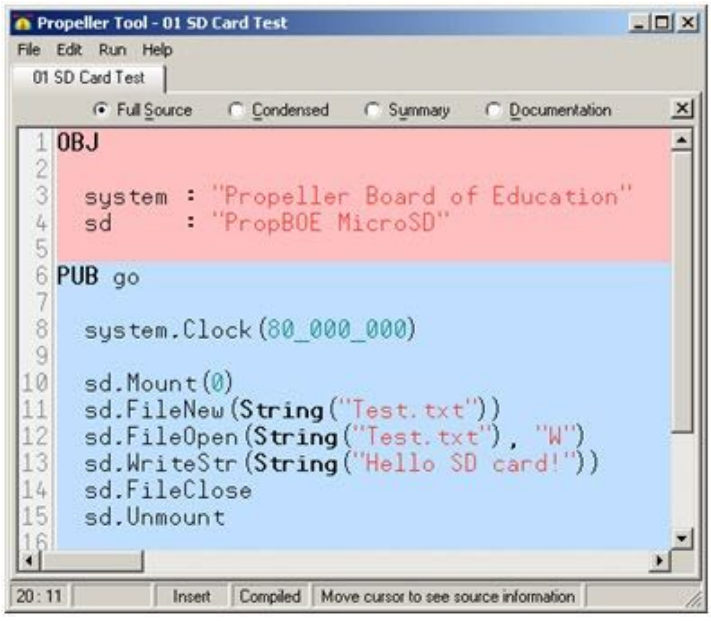
# Activity 1: Create, Modify and View a Simple Text File

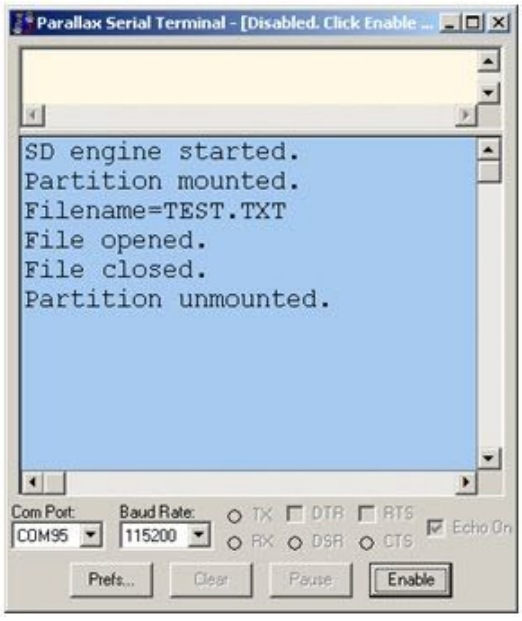
The PropBOE MicroSD object has methods for creating, modifying and reading text files. This is useful for applications that need to keep a running log of events that occur.

## SD Card Write Test

The 01 SD Card Test.spin object creates a file named Test.txt and leaves a message in it.

- ✓ Open "01 SD Card Test.spin" and use F11 to load it into the Propeller chip.
- ✓ While the program is loading, click the Parallax Serial Terminal's Enable button.
- ✓ Verify that the messages look similar to those shown in the Parallax Serial Terminal below. If you run the program twice, it will display a message indicating that it cannot create a new file with that name because one already exists.




## How it Works

You have to mount the file system before you can use methods to create, read, modify and close files. The **`sd.Mount(0)`** method call mounts partition zero, which is typically the only partition you'll see on a MicroSD card. After that sd.FileNew creates a file named Test.txt and then **`sd.FileOpen`** opens it. **`sd. WriteStr`** stores a text message in the file, and **`sd.FileClose`** closes it. Before removing the SD card from the socket, it has to be unmounted, with **`sd.Unmount`**.

## Did you Know?

The "W" parameter in the **`FileOpen`** method stands for Write. You can also use "R" for read, or "A" for append.

## Your Turn

You can add a second line to your text file by appending "Hello SD card" with 13, 10. Then, add a second line with a second **`WriteStr`** method call.

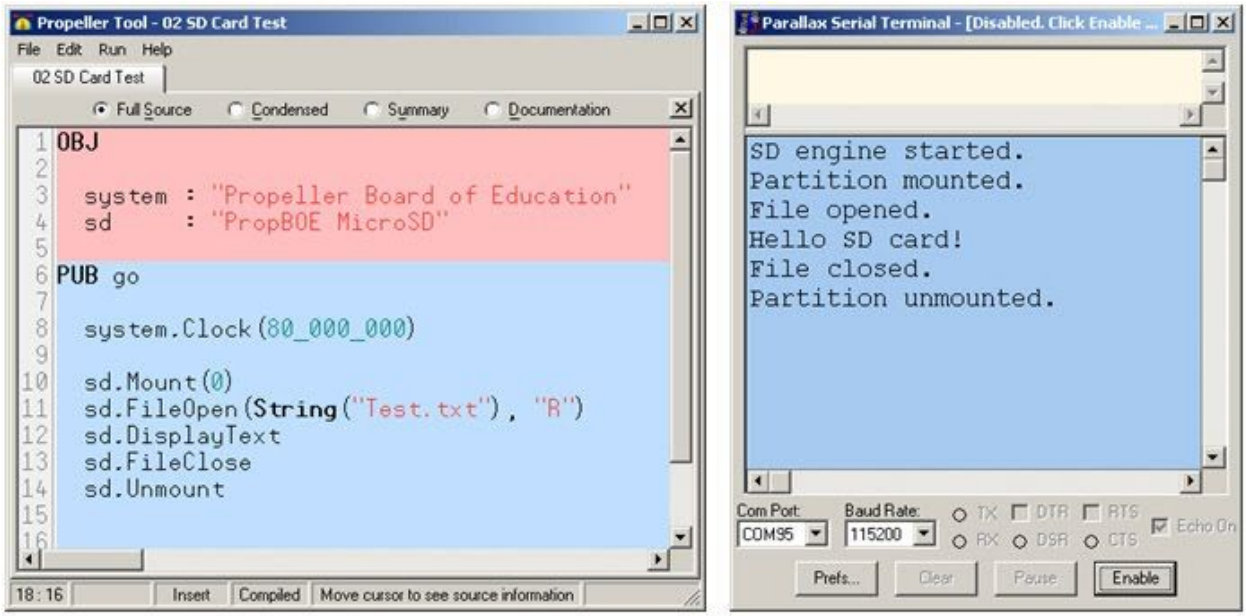
## Learn More about the Methods

- ✓ In the Propeller Tool software, click Run -> Compile Current -> View Info… F8. The F8 next to the menu selection means that you can use the F8 key as a shortcut to this feature.
- ✓ Double-click the PropBOE MicroSD object to open it. Then close the Object Info window.
- ✓ The PropBOE MicroSD object should be the active tab in your Propeller Tool software now. Click the Documentation radio button.
- ✓ Read the documentation for the following methods: **`Mount`**, **`FileNew`**, **`FileOpen`**, **`WriteStr`**, **`FileClose`**, and **`Unmount`**.

## SD Card Read Test

You can also program the Propeller to read text from a text file. This can be useful for displaying from a large dictionary of text messages that a computer stored on an SD card. It can also be useful to display what the first test program stored in the text file. Let's try it.

- ✓ Open "02 SD Card Test.spin" and use F11 to load it into the Propeller chip.
- ✓ While the program is loading, click the Parallax Serial Terminal's Enable button.
- ✓ Verify that the "Hello SD Card!" message and anything you added is displayed between the "File opened" and "File Closed" methods.



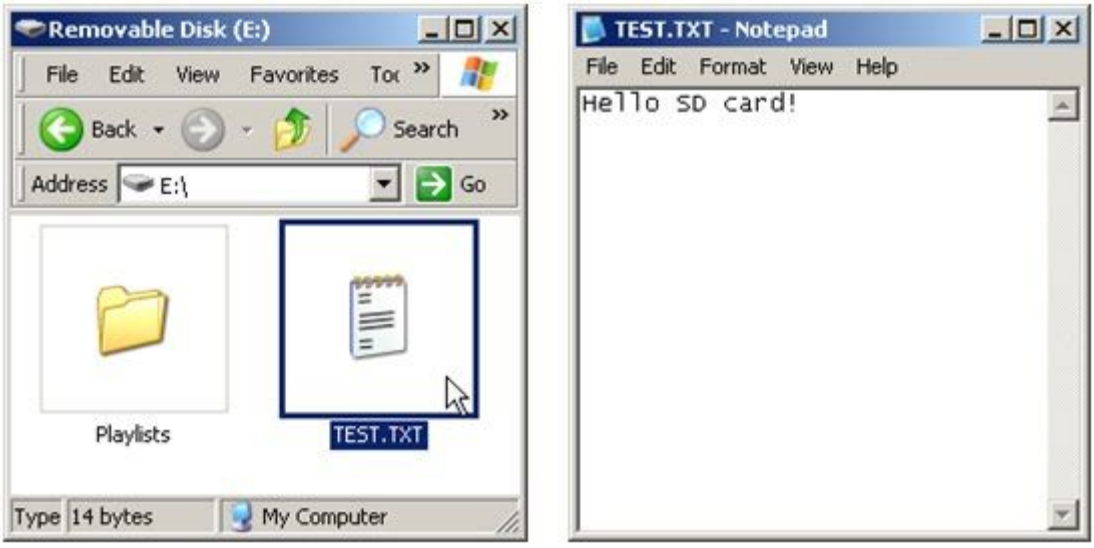

## How it Works

Notice how the sd.FileOpen call passes an “R” to the method’s mode parameter. This opens the file in Read Only mode. Then, the a call to the DisplayText method displays the contents of the file.

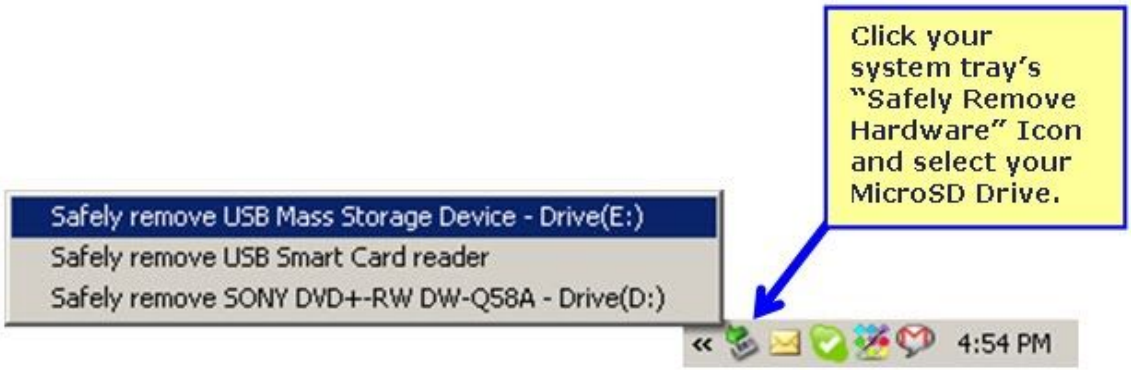
## View the Data with NotePad

You might need an SD to USB adapter to connect your MicroSD card to your computer. These are available at most electronics stores and online.

- ✓ After you insert the MicroSD card into the SD card reader and plug that into your computer, it should display it as a disk drive.
- ✓ When you open it, the TEST.TXT file should be visible.
- ✓ Double click it to open, and since it’s extension is .TXT, the Notepad program should open it.




- ✓ When you're done, make sure to close all the files and windows that access the MicroSD card drive.
- ✓ Then, click your system tray's "Safely Remove Hardware" icon and select your MicroSD drive. Wait until the system tray displays the "Safe to Remove Hardware" message before you disconnect the drive.



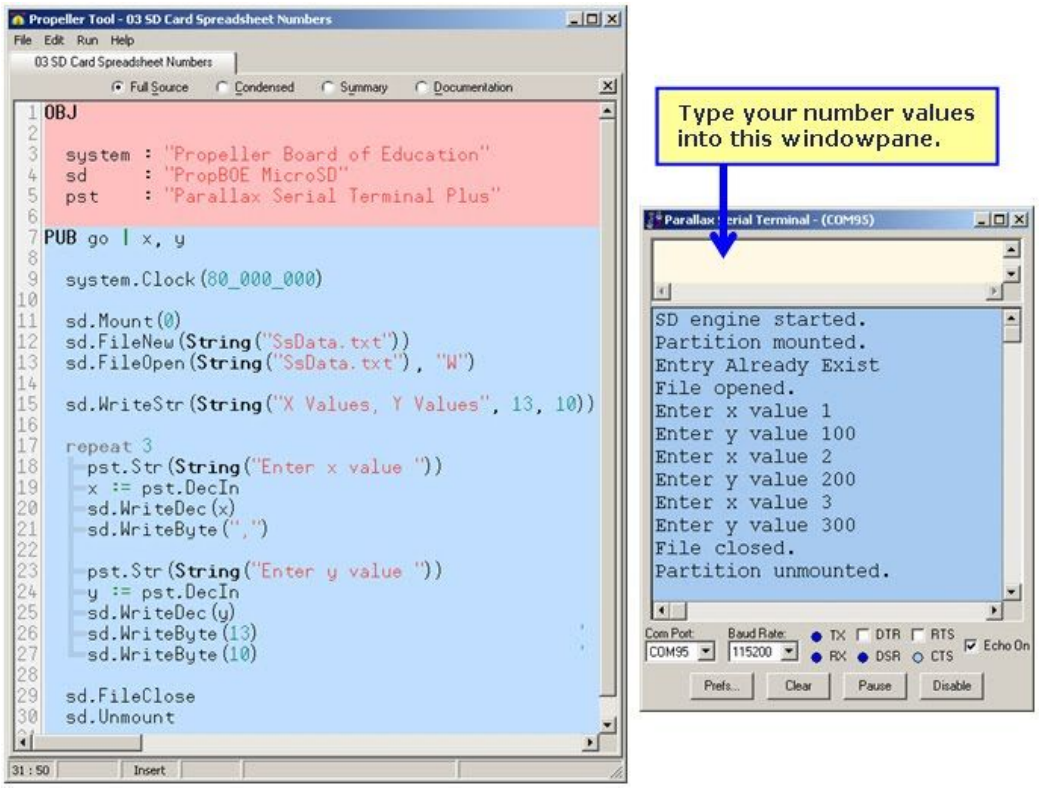

## Activity 2: Save Data and Import to Spreadsheet

This next example program saves values in their decimal text representations separated by commas for cells on the same row and carriage returns for the next row.

### SD Card Spreadsheet Numbers

This next example program saves values in their decimal text representations separated by commas for cells on the same row and carriage returns for the next row.

- ✓ Open “03 SD Card Spreadsheet Numbers.spin” and use F11 to load it into the Propeller chip.
- ✓ While the program is loading, click the Parallax Serial Terminal’s Enable button.
- ✓ Click the Parallax Serial Terminal’s Transmit windowpane and type in three pairs of x, y values. Make sure to press the Enter key after each value.




## How it Works

The x and y variables are signed, 32-bit binary numbers in the Propeller chip’s RAM. The **`sd.WriteDec`** method stores character representations of those x and y variables to the SD card. For the carriage return in Notepad, the program uses **`sd.WriteByte`** to write a 13 (carriage return) followed by a 10 (linefeed).

## Did you Know?

*Carriage return* and *line feed* are old printer and typewriter terms.

## Your Turn

Try viewing the text with 02 SD Card Test.spin. Keep in mind that the 13 followed by the 10 will cause the cursor in the Parallax Serial Terminal to add a carriage return that you will not be able to see with the Notepad program.

## Learn More about the Methods

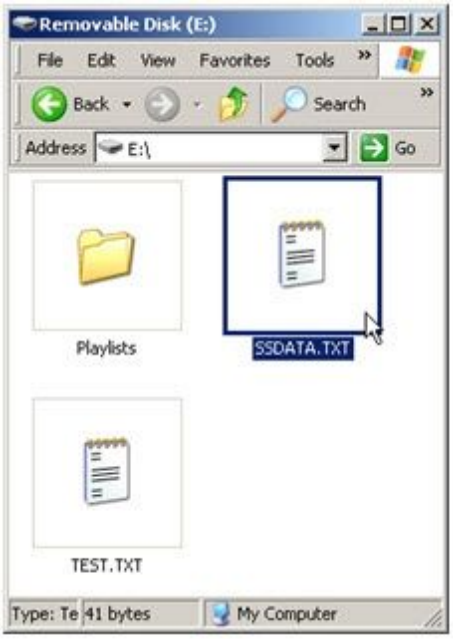
We are using some more new methods from the PropBOE MicroSD object.

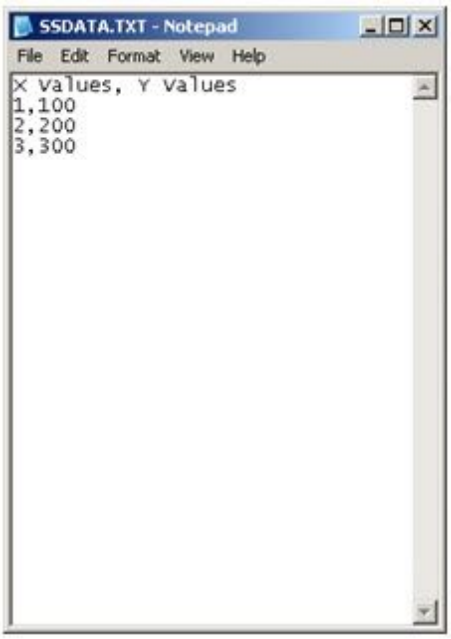
- ✓ In the Propeller Tool software, click Run -> Compile Current -> View Info… F8. The F8 next to the menu selection means that you can use the F8 key as a shortcut to this feature.
- ✓ Double-click the PropBOE MicroSD icon to open it. Then close the Object Info window.
- ✓ The PropBOE MicroSD object should be the active tab in your Propeller Tool software now. Click the Documentation radio button.
- ✓ Read the WriteDec and WriteByte method documentation. How does it relate to the example program?

## View the Text File with Notepad First

Before important the file into a spreadsheet, let's take a look at the text file first.

- ✓ View the MicroSD drive with your PC and card reader, and open the SSDATA.TXT file.
- ✓ Examine the comma delimited data.
- ✓ Close the text file when you're done.

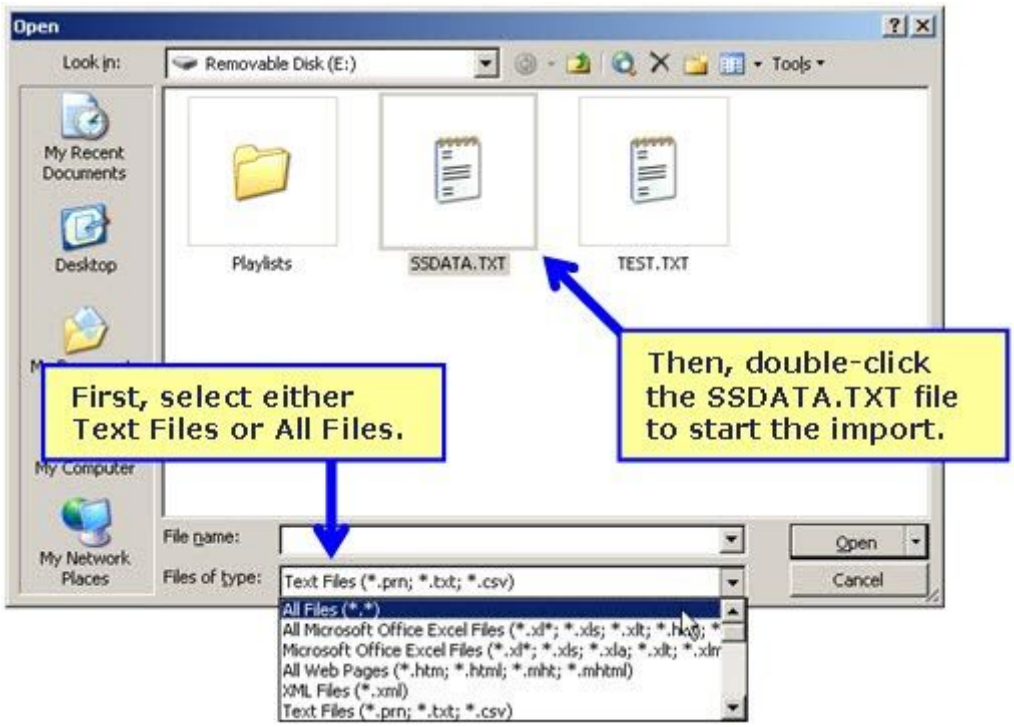



## Import the File into Microsoft a Spreadsheet

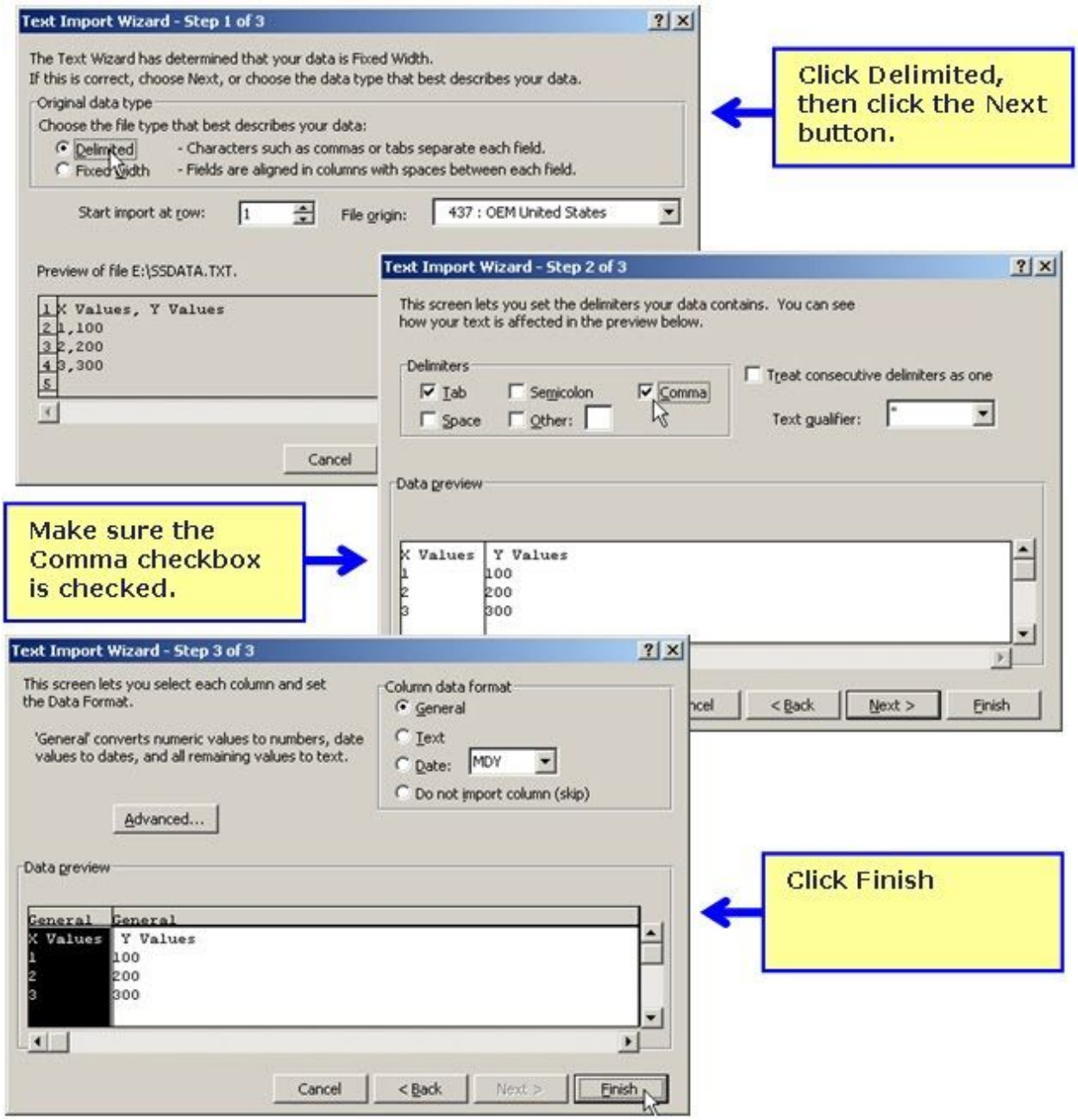
This is a Microsoft Excel example, but the process is similar for most spreadsheet software packages.

- ✓ In Microsoft Excel, click File and select Open.
- ✓ Set the Files of type field to either Text files or All files. The example below uses All files.
- ✓ Double click your SSDATA.TXT file.

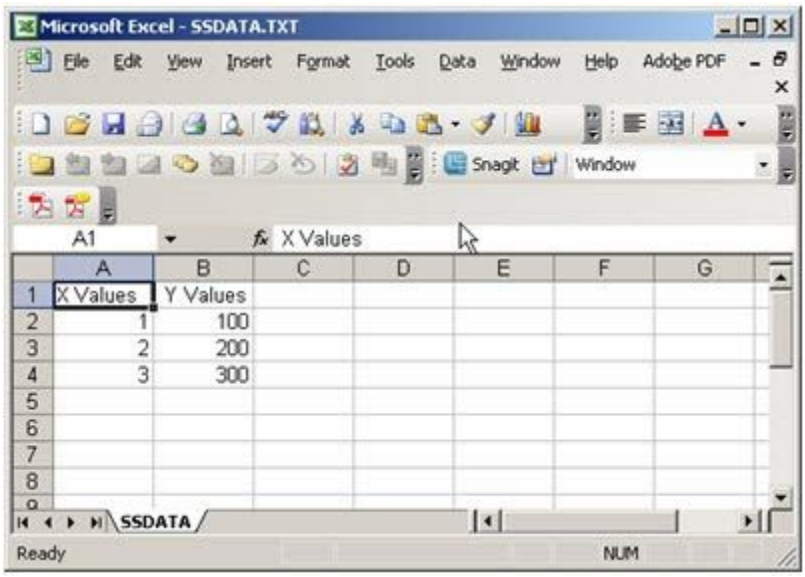



The Text Import Wizard will need to know how the values are formatted in the text file.

- ✓ In Step 1 of 3, click the Delimited radio button.
- ✓ Since your values are comma delimited, click the comma checkbox in step 2 of 3.
- ✓ If the preview looks good in Step 3 of 3, click the finish button.




Here's the result, in a spreadsheet. Now, the data is ready for analysis.




### Your Turn

- Use the spreadsheet's Chart Wizard or other graphing feature to make a graph of the values.

## Activity 3: Store and Retrieve Values from SD Card

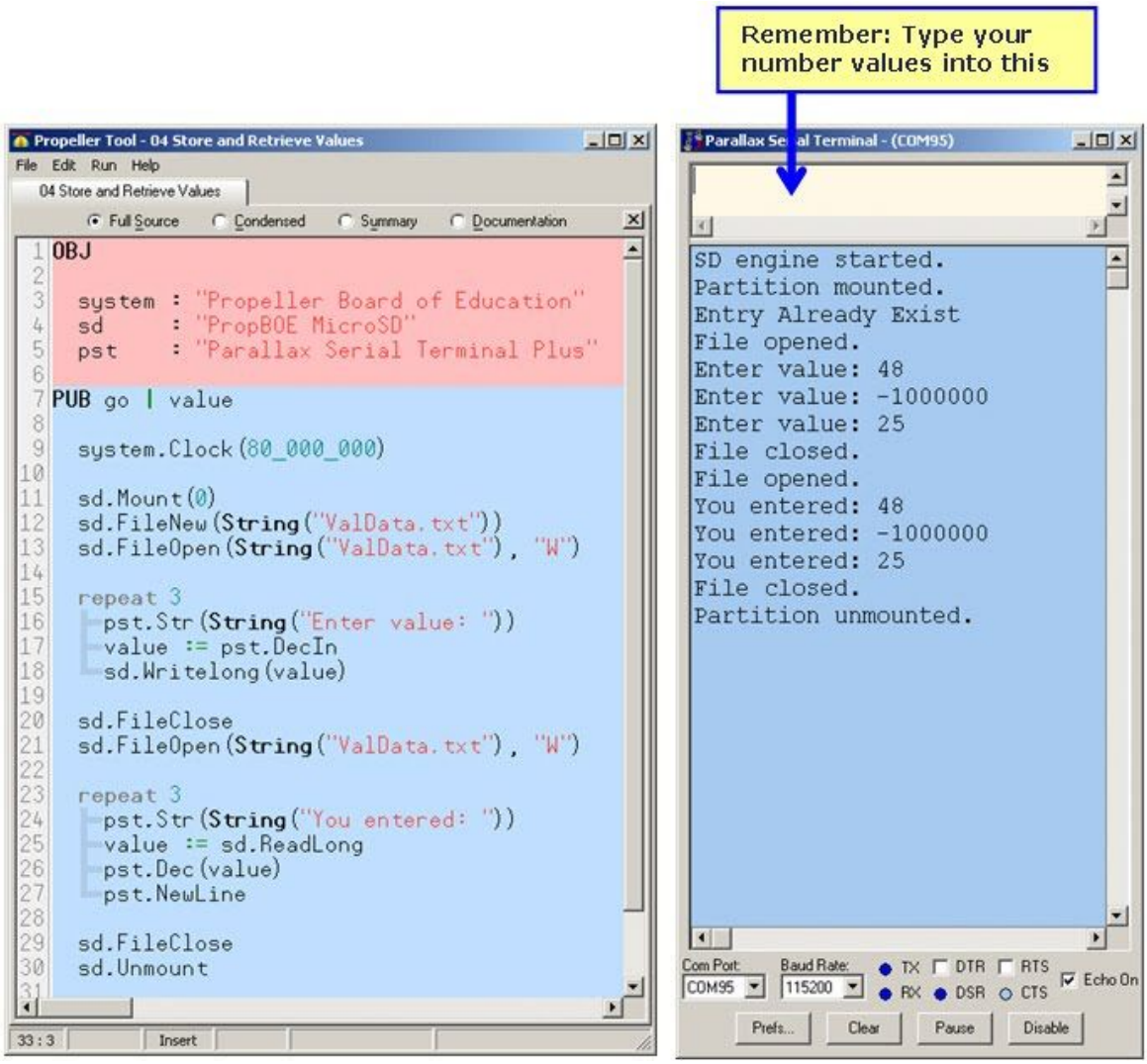
If it's just the Propeller chip that needs to store and access values, there's no need to store them as characters. It's more efficient to store them as binary values that occupy the successive byte values in the text file. It's kind of like storing values to and retrieving them from RAM. However, if you need to store values that are modified frequently, you should consider connecting a RAM chip to your Propeller instead. SD card memory cells can wear out after a certain number of write cycles.

### Store and Retrieve Values

✓ Open "1 First Program.spin" and use F11 to load it into the Propeller chip.

✓ While the program is loading, click the Parallax Serial Terminal's Enable button.

✓ Procedural instructions.




## How it Works

This application uses the PropBOE MicroSD object's WriteLong and ReadLong objects for storing and retrieving the values.

## Did you Know?

You can also use sd.WriteWord, sd.WriteByte, sd.ReadWord, and sd.ReadByte.

## Learn More about sd.Write and sd.Read

- ✓ In the Propeller Tool software, click Run -> Compile Current -> View Info… F8. The F8 next to the menu selection means that you can use the F8 key as a shortcut to this feature.

✓ Double-click the PropBOE MicroSD object to open it. Then close the Object Info window.

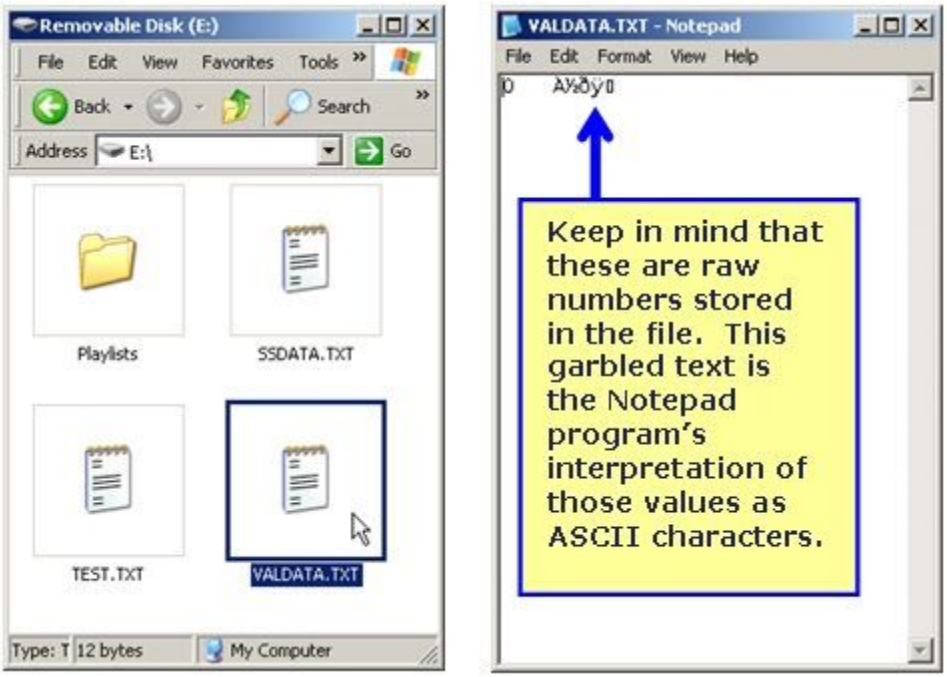
✓ The PropBOE MicroSD object should be the active tab in your Propeller Tool software now. Click the Documentation radio button.

✓ Read the **`WriteLong`** and **`ReadLong`** method's documentation. How does it relate to the example program?

## Examine the Values in Notepad

Keep in mind that Notepad is designed to display ASCII values. In contrast, the values that you stored are 32-bit binary values.

- View the MicroSD drive with your PC and card reader, and open the SSDATA.TXT file.
- Note that the numbers don't make any sense when viewed with notepad.
- Close the text file when you're done.




## How it Works

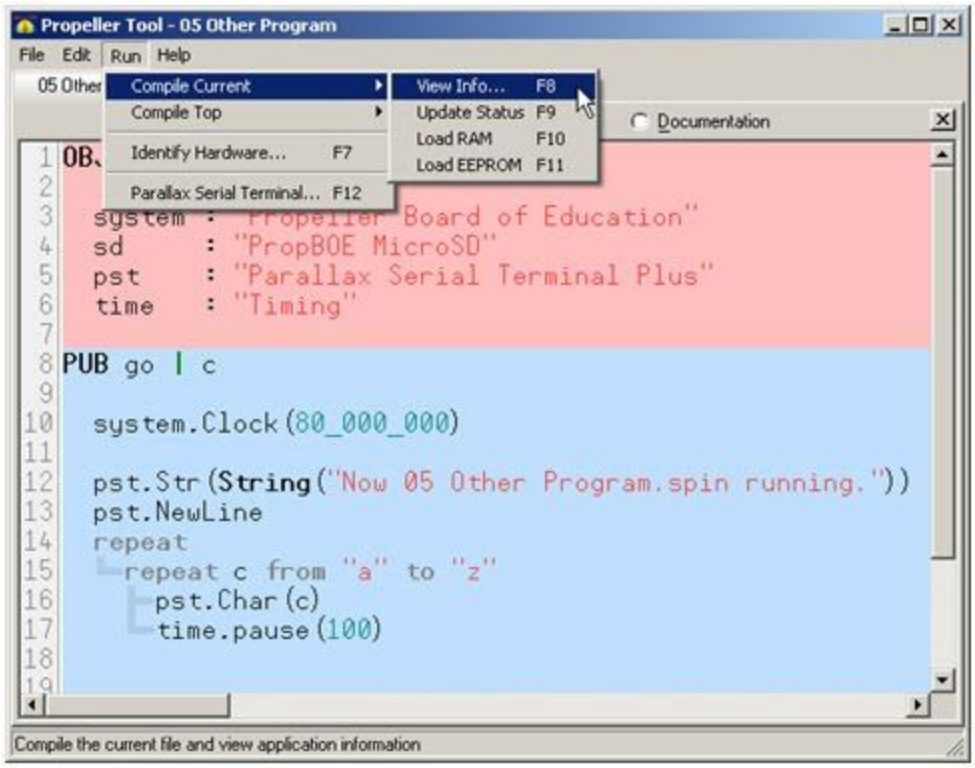
Remember that your Propeller program successfully stored and retrieved the values from VALDATA.TXT. Well, in their 32-bit signed binary format, they don't make a lot of sense in Notepad because they are not in a format that is designed for Notepad.

# Activity 4: Make One Program Launch Another

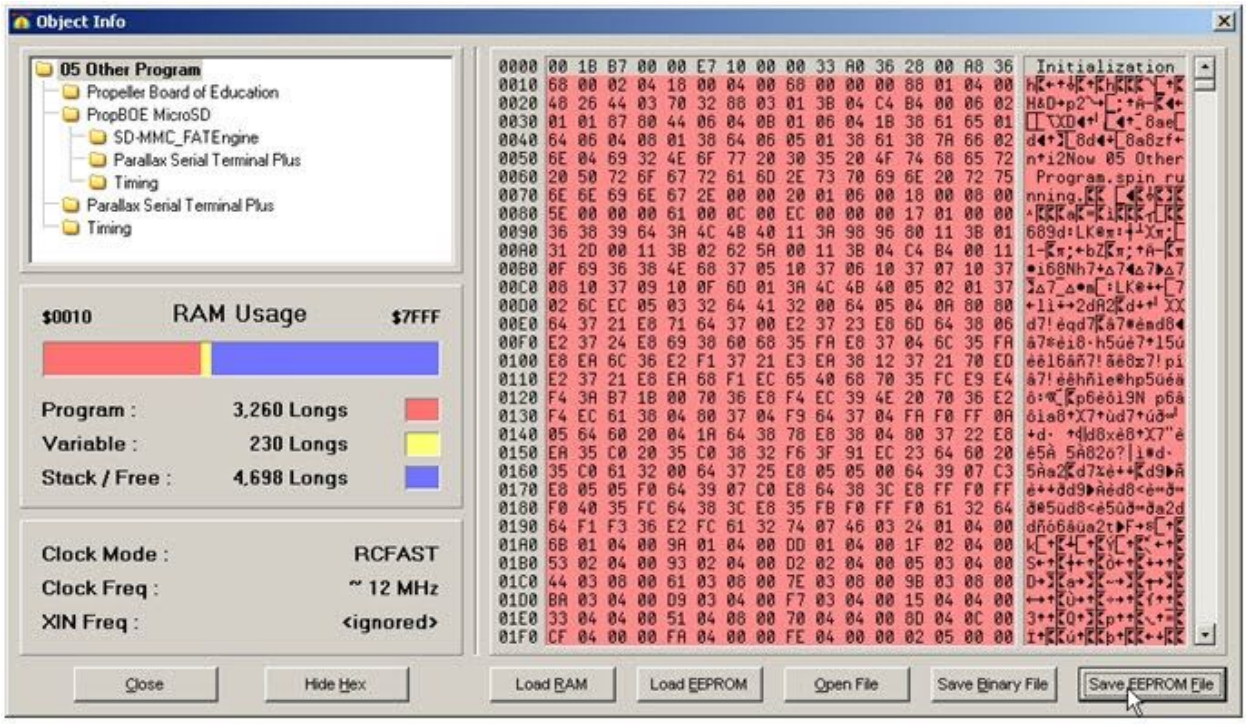
You can use the Propeller Tool to store many programs on the MicroSD card, and then write programs that load them into the Propeller chip and launch them. It's called a *soft boot*.

## Program to Be Launched to SD Card

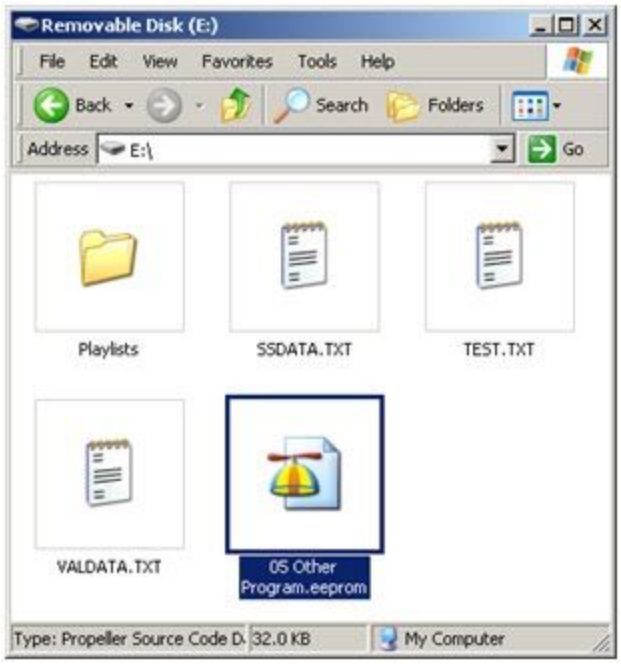
- ✓ Open "05 Other Program.spin".
- ✓ Click Run, then select Compile Current and View Info… F8.




- ✓ Use your computer and MicroSD card reader to view your MicroSD drive.
- ✓ If just the left side of the Object Info window is visible, click the Show Hext button to get the rest of it to display.
- ✓ Click the Save EEPROM File button.
- ✓ Save the file to your MicroSD drive.




✓ Check your MicroSD drive to verify that the “05 Other Program.eeprom” file is there.



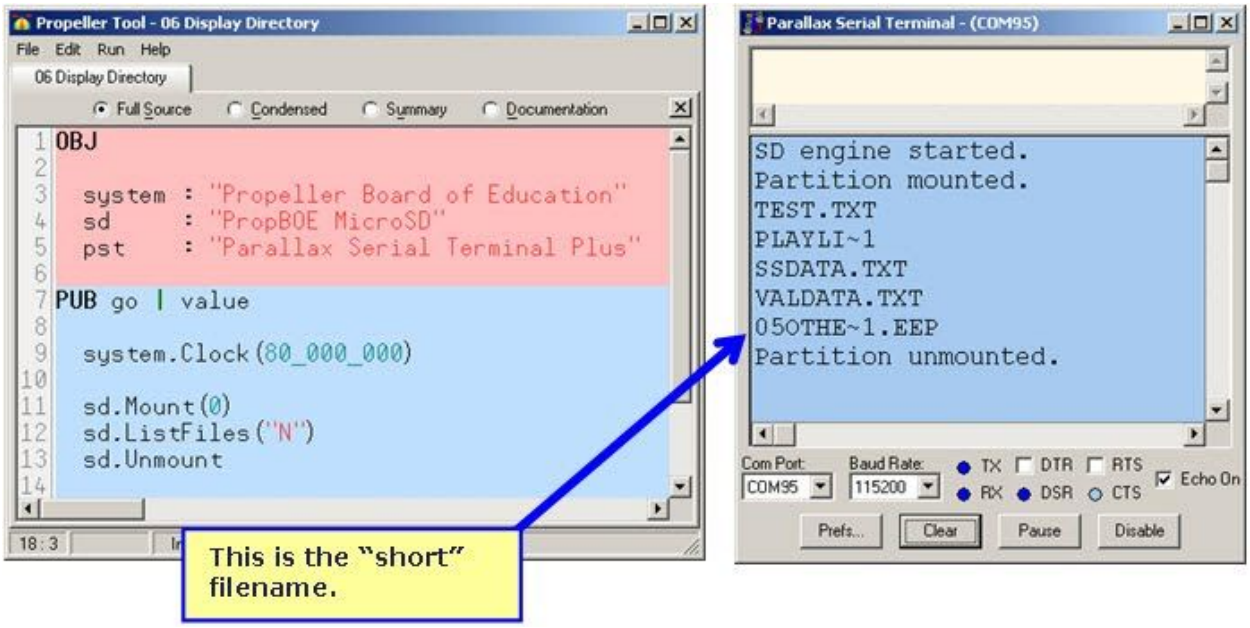

## How it Works

The .eeprom file is a binary image of the program that can be loaded into the Propeller chip and run. Since it’s on the MicroSD card, it can be accessed by the PropBOE MicroSD object.

## Display Directory

The SD_MMC_FATEngine object the PropBOE MicroSD card object abstracts is designed to work with the short filename. To see the short filename, you can use the sd.ListFiles method.

- ✓ Insert the MicroSD card into your PropBOE's socket.
- ✓ Use the Propeller Tool to load 06.Display Directory.spin into the PropBOE.
- ✓ While the program is loading, click the Parallax Serial Terminal's Enable button.




## How it Works

When you write code to perform operations on files, use the short filename. This program allows you to view it.

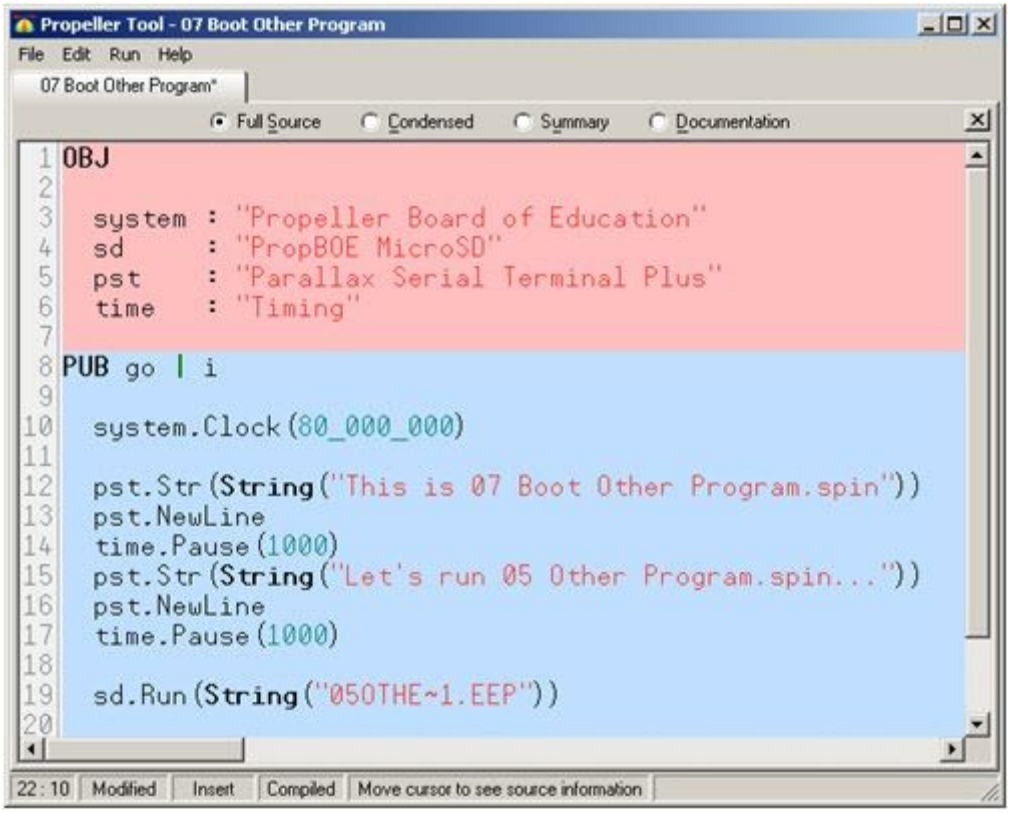
## Did you Know?

The Short filename is also called the 8.3 filename.

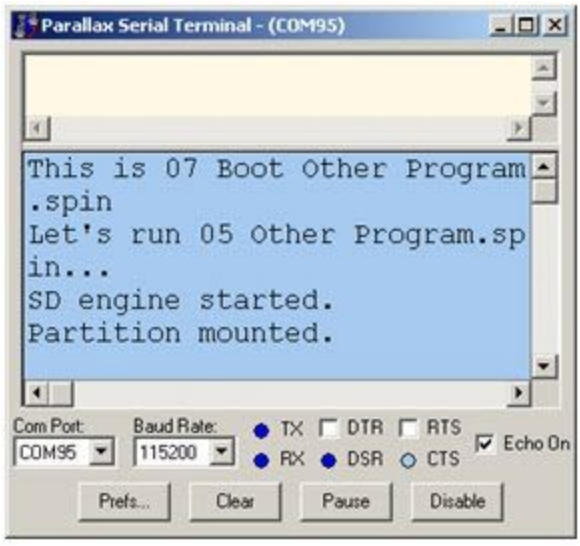
## Boot One Program with Another

Alright, now we can load a program into the Propeller that boots the program we stored on the MicroSD card.

- ✓ Open "07 Boot Other Program.spin" and use F11 to load it into the Propeller chip.
- ✓ While the program is loading, click the Parallax Serial Terminal's Enable button.




✓ Watch carefully, the display on the left is 07 Boot Other Program.spin running. Then, the screen will clear and you'll see the output from 05 Other Program.spin when it loads.





## How it Works

After running and doing some jobs, “07 Boot Other Program.spin” passes the short filename of 05OTHE~1.EEP to the PropBOE MicroSD card object’s sd.Run method. This loads the 05 Other Program.spin EEPROM image into RAM, and the program starts running. Since it’s loaded into RAM, it loads quickly. The 1 second delay is for the time.Pause method call in the 07 program to give you time to read the display.

## Did you Know?

You can chain programs together, or use IF or CASE statements to choose which program to load.

## Your Turn

Try writing a program that uses IF statements to decide which program to run based on either Parallax Serial Terminal input characters or button presses.

## Learn More about the Run Method

- ✓ In the Propeller Tool software, click Run -> Compile Current -> View Info… F8. The F8 next to the menu selection means that you can use the F8 key as a shortcut to this feature.
- ✓ Double-click the PropBOE MicroSD object to open it. Then close the Object Info window.
- ✓ The PropBOE MicroSD object should be the active tab in your Propeller Tool software now. Click the Documentation radio button.
- ✓ Read the Run method’s documentation.