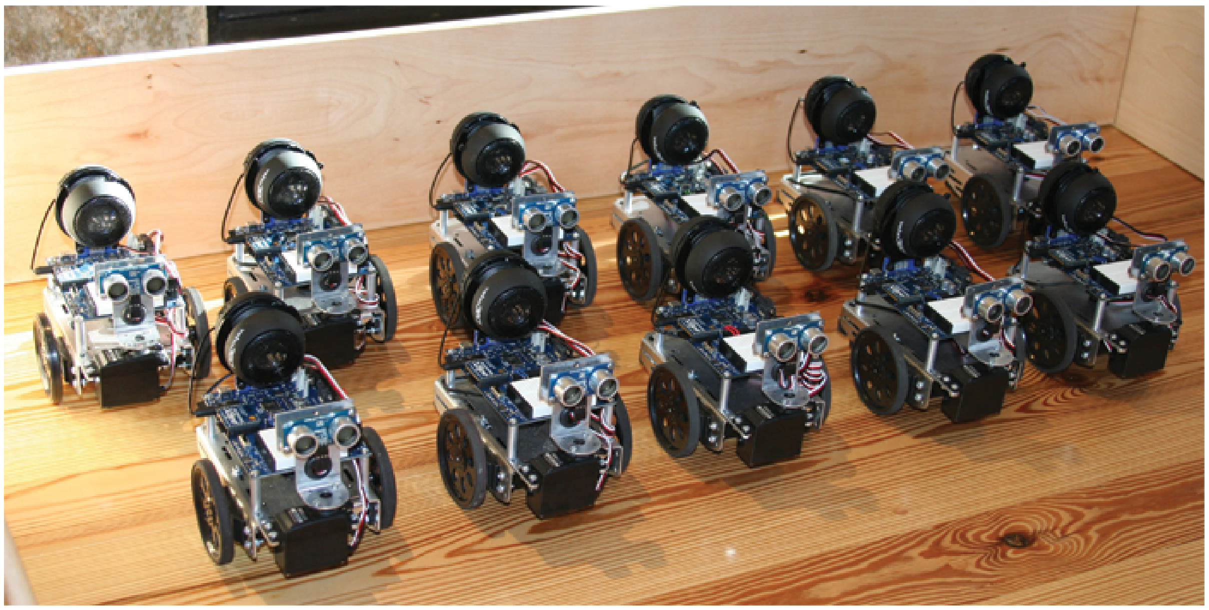
Spin Project: Juke-Bots Propeller Robots

Level: Intermediate
Skills Required: Programming
Hours to Complete: 2-3 hours (depending on number of Juke-Bots built)



View this project's YouTube video on the ParallaxInc YouTube channel.

The Propeller community developers designed a really easy-to-use object that allows you to play back WAV files from an SD card. Borrowing from the efforts of Jon McPhalen, Tomas Rokicki and Jonathon Dummer I was able to quickly couple their objects along with Andy Lindsay's Learn.parallax.com Boe-Bot code to make a fleet of music-playing Propeller robots called Juke-Bots. The robots are controlled from a PC over a pair of XBee modules, giving you easy control to start and stop the robots and their music. EFX-TEK's very successful commercial product AP-16 uses the same WAV file playback code base, showing how the same objects may be quickly adapted for special effects for Halloween, Christmas or museum displays.

What's Required

One Propeller BOE Transmitter may control many Juke-Bots, so you may duplicate the receiver portion for as many robots as you wish to control.

Propeller BOE Transmitter:

- (1) Propeller Board of Education (#32900, discontinued)

- (2) XBee 802.15.4 2 1 mW Chip Antenna (#32406, discontinued)

Propeller Boe-Bot Receiver:

- (1) Propeller Board of Education (#32900, discontinued)
- (1) SD Micro Card (#32319)
- (1) XBee 802.15.4 2 1 mW Chip Antenna (#32406, discontinued)
- (1) Robotics with the Boe-Bot Parts Kit (#28124)
- (1) Ping))) Mounting Bracket Kit (#570-28015)
- (1) Ping))) Ultrasonic Sensor (#28015)
- (1) Veho 360 Speaker Stand for Propeller Board of Education (#725-32900, discontinued)
- (1) Veho 360 Speaker (#900-00018)

Note: Many of these parts are discontinued for sale or manufacture by Parallax, but may be found through other retailers, or have suitable replacements available through Parallax or other retailers.

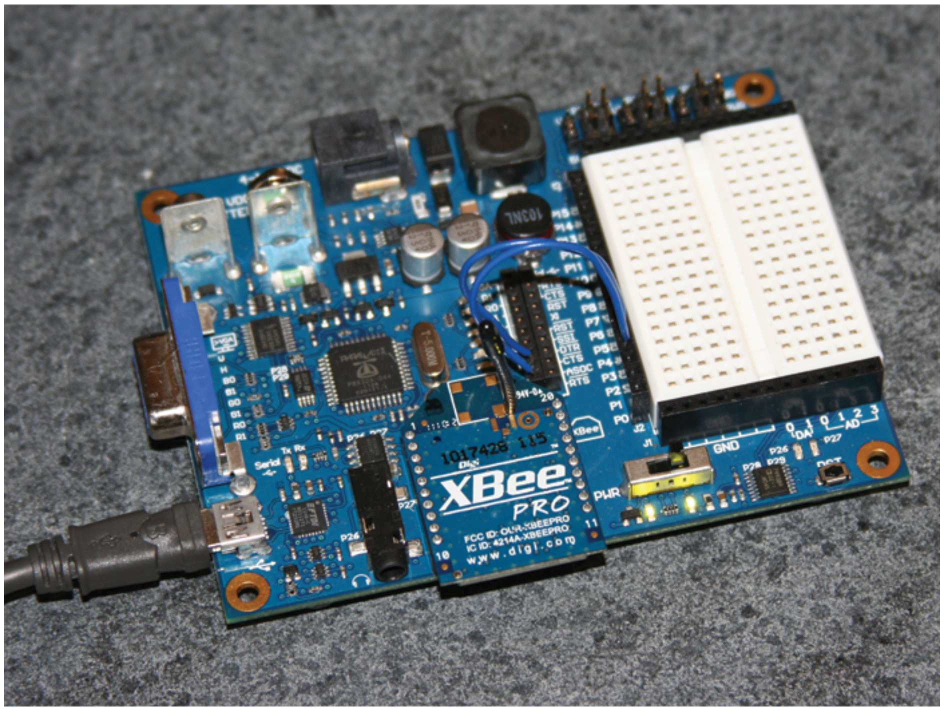
Building the Juke-Bots

Build the Propeller BOE Transmitter

If you have not used an XBee module with your Propeller BOE before, see the Wireless Programming with XBee Spin tutorial from the same download page you found this project. If you try that tutorial, be sure to change your I/O pin connections for the XBee DO and DI pins to the ones shown below when you build this project.

Propeller BOE Transmitter I/O Connections

P3	XBee 802.15.4 Data In (DI) Pin
P4	XBee 802.15.4 Data Out (DO) Pin
P30	USB COM port Tx
P31	USB COM port Rx



Build the Propeller Juke-Bot Receiver

To build a robot with your Propeller Board of Education, follow the directions in the first three sections of the Robotics with the Propeller Board of Education tutorial, also found on the download page as this project.

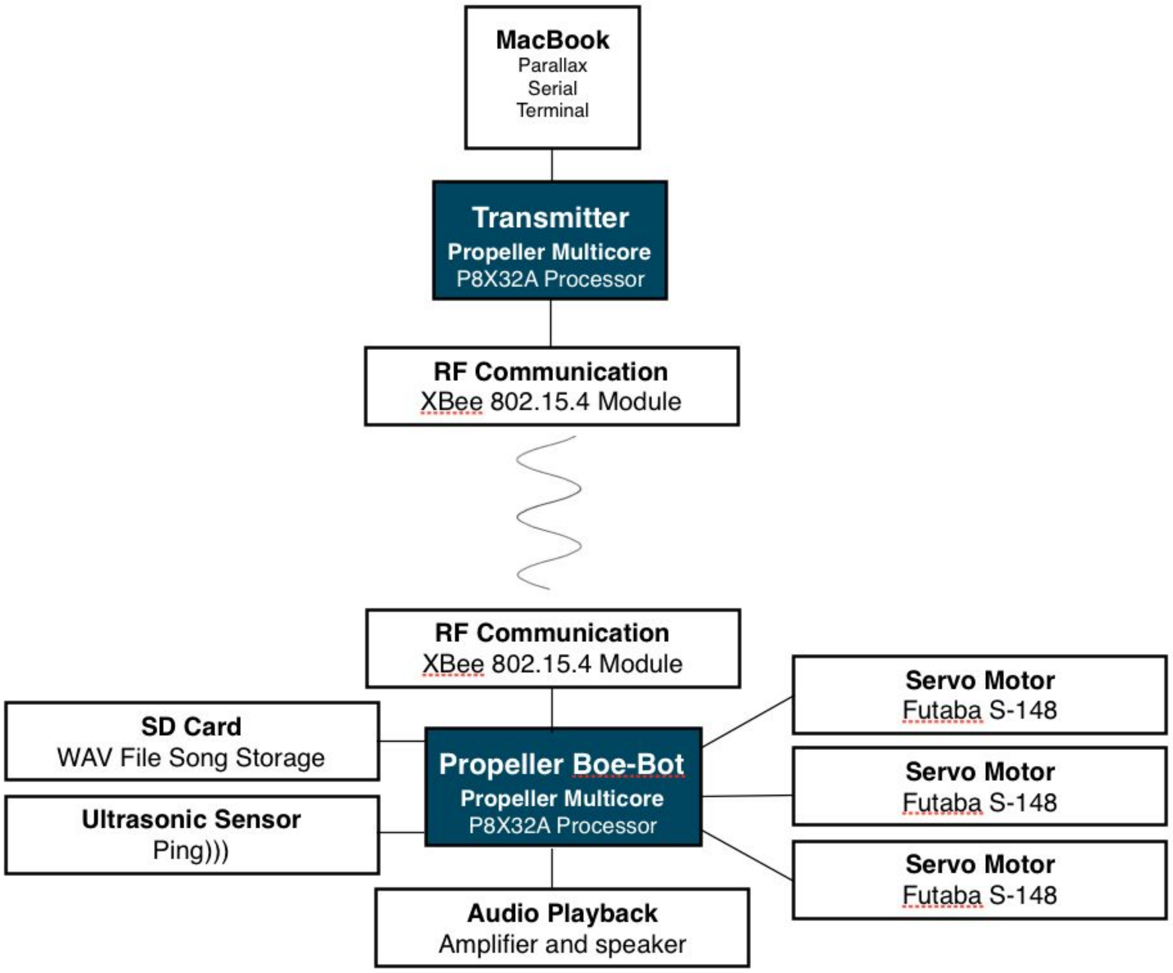
Add the PING))) Sensor and rotating turret, following the directions in the Mounting Bracket Kit and using the I/O pins for the sensor and servo listed below.

If you have not used your Veho speaker before, you may need to charge its battery before using it (USB charging cable included). Mount the Veho Speaker stand to the VGA connector, then open the speaker and slip it over the stand. There should be just enough cord length to plug the speaker into the audio jack on the Propeller BOE. The stereo jack is hard-wired to P26.
The SD card connections are hard-wired into the Propeller Board of Education, so there's no need to build circuits for that.

Propeller Boe-Bot Receiver I/O Connections

● P3	XBee 802.15.4 Data In (DI) Pin
● P4	XBee 802.15.4 Data Out (DO) Pin
● P14	Propeller Boe-Bot Left Servo
● P15	Propeller Boe-Bot Right Servo
● P17	Ping))) Mounting Bracket Servo
● P19	Ping))) Ultrasonic Sensor
● P22	microSD Data Out (DO), pre-configured on PropBOE

- P23 microSD Clock (CLK), pre-configured on PropBOE
- P24 microSD Data In (DI), pre-configured on PropBOE
- P25 microSD Chip Select (SC), pre-configured on PropBOE
- P26 Veho360 speaker through PropBOE audio amplifier



Programming the Juke-Bots

Configure XBee Modules

Using Digi's X-CTU software, configure two 802.15.4 XBee modules to communicate over the same channel (D is fine) and to have a baud rate of 9600 bps.

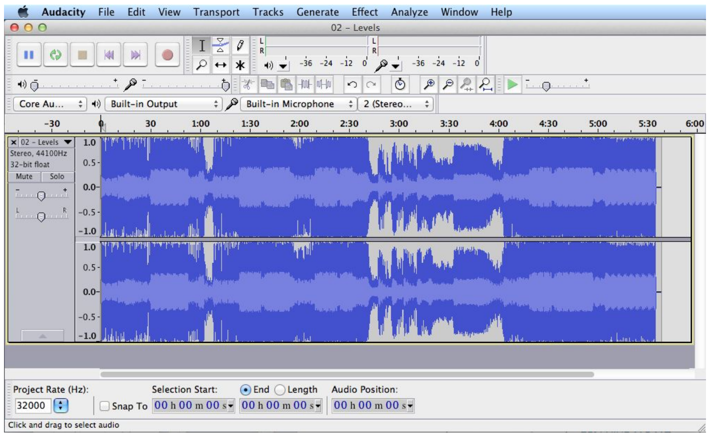
Output WAV Files for SD Card Using Audacity

Audacity is a free, open source, cross-platform software for recording, editing and outputting sounds. You may use it to output a WAV file suitable for playback by the Propeller. You can also record and output your own sound effects with Audacity. EFX-TEK (www.efx-tek.com) has

produced a nice document entitled “Essential Audacity” which provides more detailed instructions for basic recording, mixing, editing and outputting sound effects. Read this document if you are interested in recording and playing back your own sound effects, or simply follow the steps shown below to format any existing sound file for output.
These are the basic steps to output a WAV file that the Propeller can play.

- ✓ Download and install Audacity from http://audacity.sourceforge.net.
- ✓ Choose File/Open to load an MP3 file (or any other format) in Audacity. I purchased my example songs from Amazon.

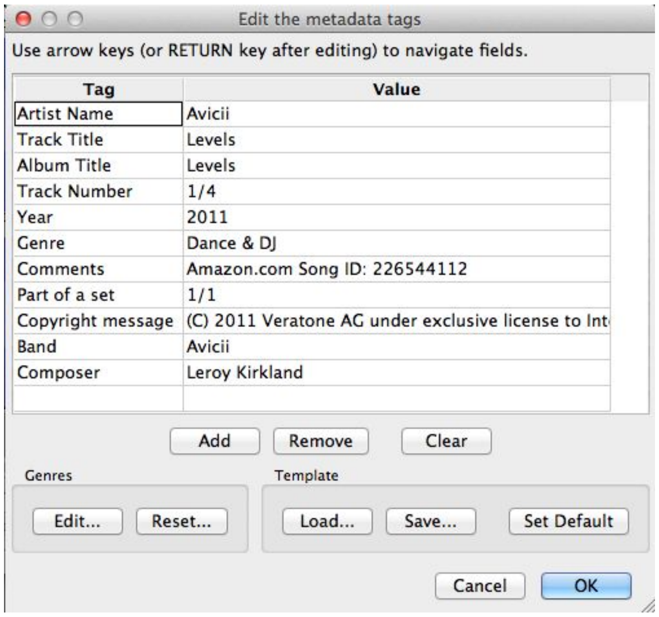
You’ll see something like the following:



02 – Levels
Stereo, 44100Hz
32-bit float

Project Rate (Hz): 32000
Snap To
Selection Start: 00 h 00 m 00 s
End / Length: 00 h 00 m 00 s
Audio Position: 00 h 00 m 00 s
Click and drag to select audio

- ✓ Choose File / Open Metadata Editor to see the metadata that comes with the file. Click “clear” to remove the metadata and then OK since the example code isn’t designed to filter out the metadata. Consider configuring your preferences to ask this question each time you output a file.

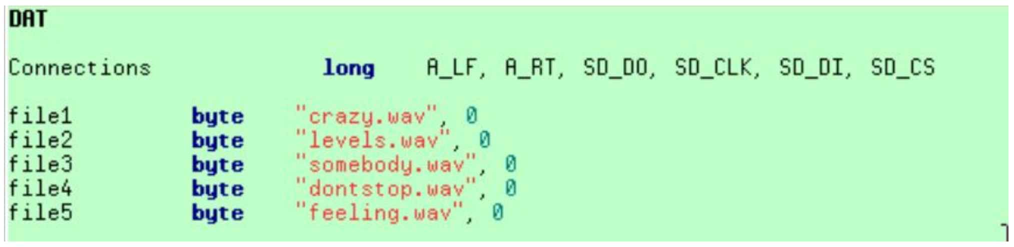

Edit the metadata tags

Use arrow keys (or RETURN key after editing) to navigate fields.

Tag	Value
Artist Name	Avicii
Track Title	Levels
Album Title	Levels
Track Number	1/4
Year	2011
Genre	Dance & DJ
Comments	Amazon.com Song ID: 226544112
Part of a set	1/1
Copyright message	(C) 2011 Veratone AG under exclusive license to Int
Band	Avicii
Composer	Leroy Kirkland

- ✓ Export the file to a WAV format. Choose File / Export and chose WAV 16-bit PCM format. Save the file to your SD card. Name the file the same way you would name it in your Propeller Boe-Bot Receiver program.

For example:



```
DAT

Connections                long    A_LF, A_RT, SD_DO, SD_CLK, SD_DI, SD_CS

file1          byte     "crazy.wav", 0
file2          byte     "levels.wav", 0
file3          byte     "somebody.wav", 0
file4          byte     "dontstop.wav", 0
file5          byte     "feeling.wav", 0
```

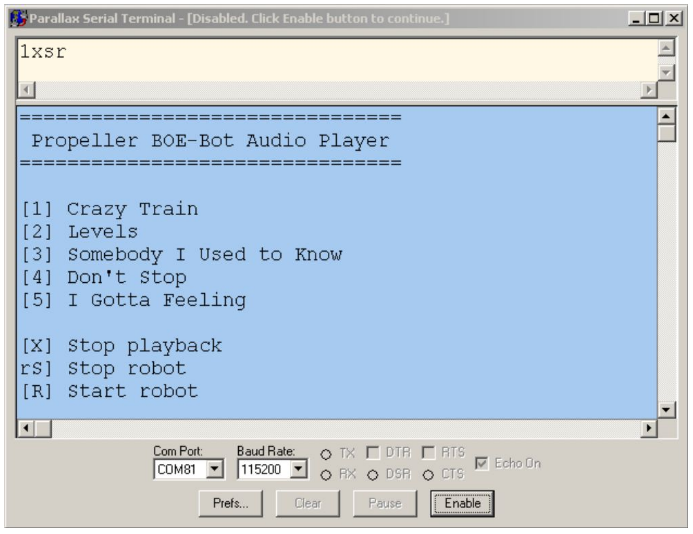
- ✓ Repeat the process for every WAV file you wish to play back, saving them on the micro SD card and naming them in the source code.
- ✓ Put the micro SD card into your Propeller Juke-Bot Receiver.

Load Code into Propeller Juke-Bot Receiver

- ✓ Program the robot(s) using the file named "RF Controlled WAV and Autonomous Roam with PingDar (006).spin".
- ✓ Verify that your DAT section uses the same names of the WAV files as you saved to the SD card.
- ✓ Connect the veho360 speaker to the robot and you're ready!

Load Code into the Propeller BOE Transmitter

- ✓ Download the "Propeller BOE Transmitter.spin" program to the Propeller Board of Education. Start up Parallax Serial Terminal and you'll see a menu system to communicate with the robots.



Parallax Serial Terminal - [Disabled. Click Enable button to continue.]

```
1xsr
```

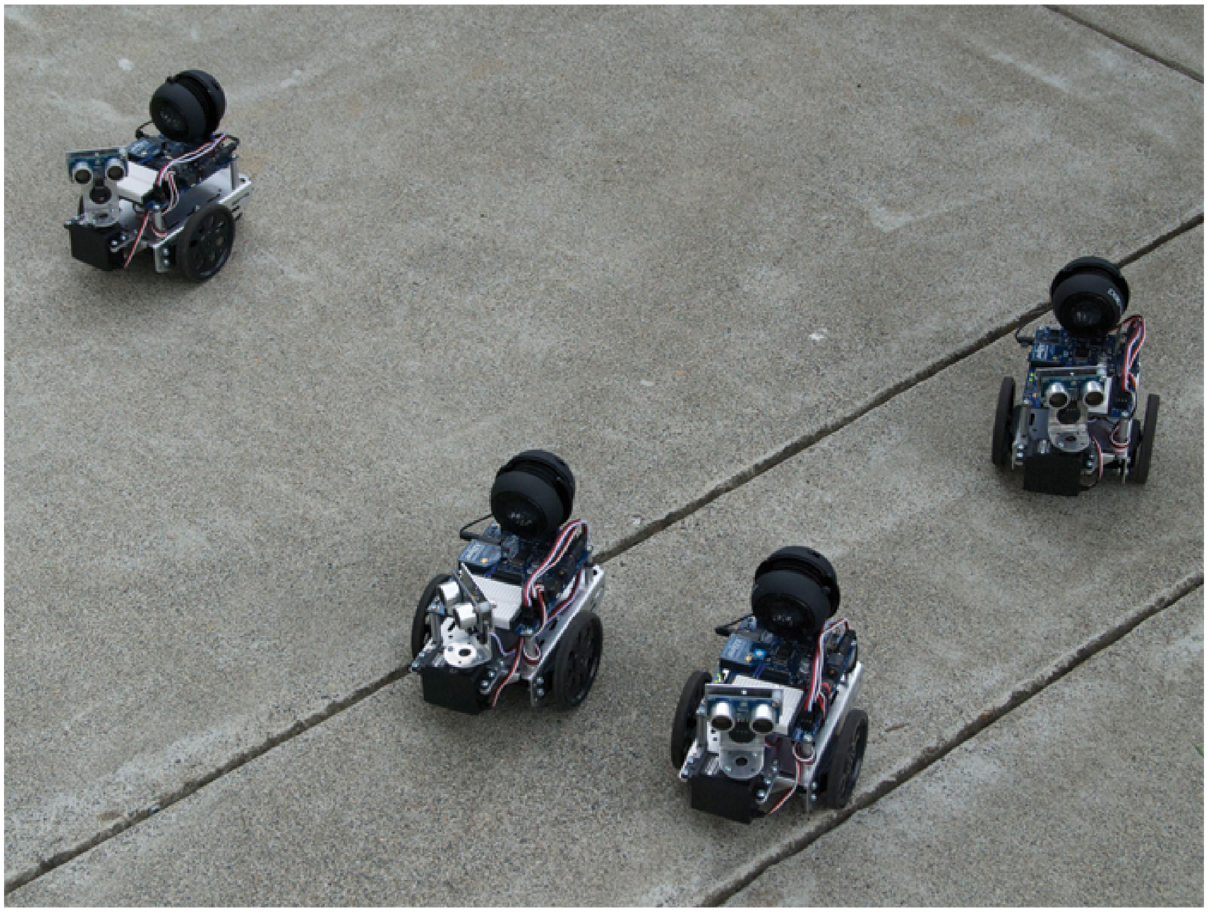
```
================================
 Propeller BOE-Bot Audio Player
================================

[1] Crazy Train
[2] Levels
[3] Somebody I Used to Know
[4] Don't Stop
[5] I Gotta Feeling

[X] Stop playback
rS] Stop robot
[R] Start robot
```

Com Port: COM81 Baud Rate: 115200 Echo On [✓]

✓ Place your mouse in the white pane in the top of Parallax Serial Terminal. Type in the characters you wish to start and stop the Juke-Bots, to select a song, and to stop a song.
✓ Now try them out!



How the Juke-Bots Work

The key part of this project is the WAV file reading and playback code. The other pieces are documented in detail elsewhere. For example, the Getting Started with XBees tutorial is available as a free download. You complete this project with only a minimal understanding of the XBee resources.

Playing a WAV File From Propeller

Playing a standard WAV file is not very difficult but does require the careful management of Propeller resources from the code; four cooperative cogs to be precise.

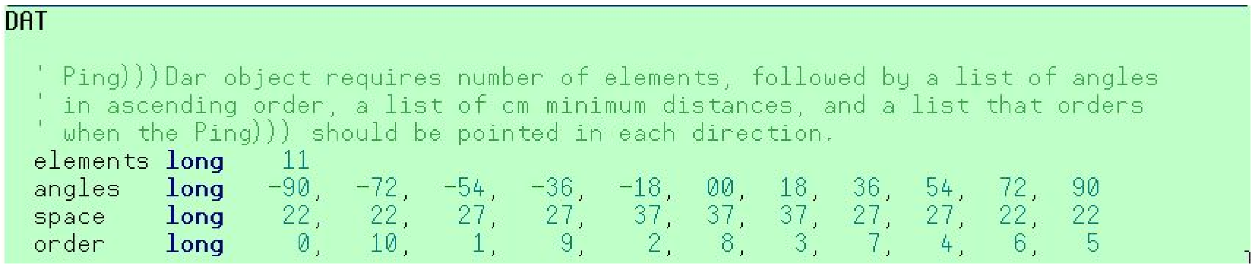
The core of the code is the FSRW object which allows the Propeller to open and read the file from an SD/uSD card. A buffer management cog is launched which reads the file to provide a steady stream of samples to the player cog. The data stream is maintained by using a double buffer approach: as one buffer is being played, the other is being filled with new samples from the open file. The third cog takes the samples and processes them based on the file type (mono or stereo), volume settings for each channel, and the intended playback rate (which can differ from the record rate if desired).

After processing, the samples are made available to a fourth cog which uses the Propellers counters with an external RC circuit to regenerate the analog signals. This cog also employs a special noise reduction technique that removes low-frequency noise (clicks and pops) that can result from cog signal paths across the Propeller chip. This fourth cog could be eliminated by adding a high-speed, 16-bit stereo DAC to the system and writing to that DAC with the player cog.

Adjusting the PING))) Ultrasonic Distance Sensor Thresholds

Depending on the amount of space where your Propeller Juke-Bots roam you might want to adjust several program parameters used with the Ping))) Ultrasonic Distance Sensor. These are found in the DAT section of the Propeller Boe-Bot Receiver.spin top file.

- **angles** constants refer to the position where each of the Ping))) distance samples will be taken. The -90 degrees is far left, and 90 degrees is far right, making a complete 180-degree sweep.
- **space** is the distance (inches) to an object which will trigger the evasive action of turning away. For sensitivity in tight spaces, reduce the space values to short distances of 6 to 12 inches. For wide-open rooms you may set the space values to 24, 36, or even higher.
- **order** refers to the sample precedence. To avoid interference the servo sweeps from one side to the other, but only samples angles that are not immediately adjacent to one another.



```
DAT

  ' Ping)))Dar object requires number of elements, followed by a list of angles
  ' in ascending order, a list of cm minimum distances, and a list that orders
  ' when the Ping))) should be pointed in each direction.
  elements long    11
  angles   long   -90,  -72,  -54,  -36,  -18,  00,  18,  36,  54,  72,  90
  space    long    22,   22,   27,   27,   37,  37,  37,  27,  27,  22,  22
  order    long     0,   10,    1,    9,    2,   8,   3,   7,   4,   6,   5
```

Propeller Juke-Bot Receiver Cog Usage

Cog 0: Setup & Main Loop

- Setup: Launch Cogs 1, 2, 3, 4
- Loop: Check Hub for data from XBEE
- If play/stop WAV, launch/stop Cog 5
- If robot start, launch Cog 6
- If robot stop, stop Cog 7 and Cog 6

Cog 1: XBee

- Receive data from other XBee and put it into
- Shared memory in Hub

Cog 2: Stereo Output

- Rapidly grabs DAC values from Hub put there by Cog 4, and sets DA level with them to generate stereo output.

Cog 3: SD SPI Driver

- Checks indexes from Cog 5 to see which of 2 buffers to fill
- Fills buffers with WAV data from SD card using SPI protocol

Cog 4: WAV to DAC

- Siphons WAV data from buffers
- Updates indexes to say when buffers are empty
- Transforms WAV files to DAC values
- Writes DAC values to Hub for Cog 2 to use

Cog 5: WAV File Spooler

- Directs traffic with indexes for filling buffers
- Checks indexing values from Cog 4 to see if buffers are emptied yet
- Passes directions of where (which buffer) Cog 3 should stuff values into next

Cog 6: Navigation Loop

- Launch Cog 7
- Update Turret servo position
- Trigger / receive distance data from Ping))) sensor, using local variables
- Decide navigation based on turret servo angle and Ping))) distance
- Write new servo speed/positions to Hub

Cog 7: Drive Servos

- Get Wheel servos and Ping))) servo settings from Hub
- Set and refresh signal to position Ping))) servo
- Set and refresh velocity signals for each wheel