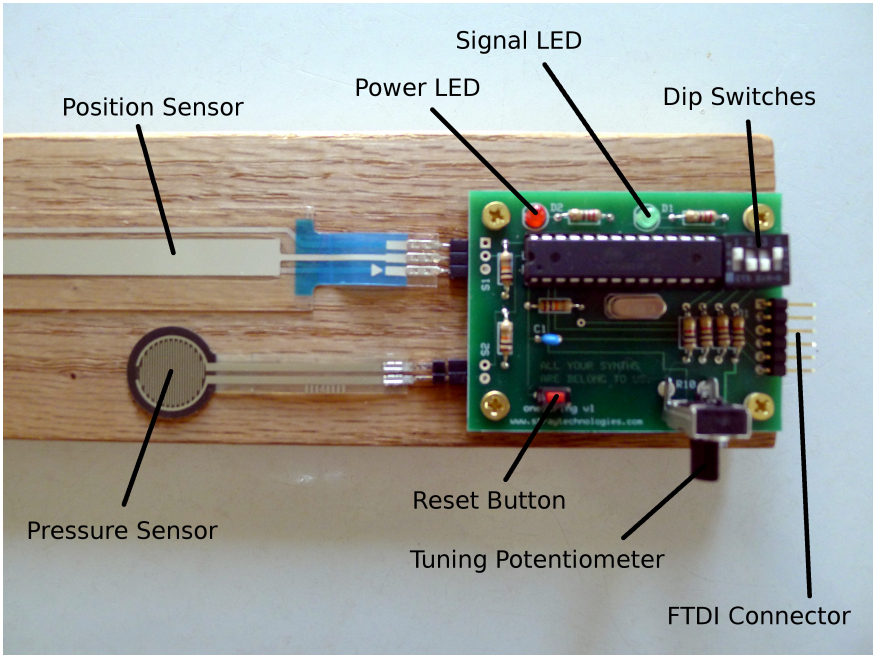
**oneString Controller** Wil Lindsay
open source USB ribbon midi controller www.straytechnologies.com
**Getting Started with Software** wil@straytechnologies.com

This guide serves as the roadmap to getting started with the oneString Controller, and getting your first grumbles out of the device. As there are many flavors and version of several operating systems, with hundred of potential pieces of software in a musician's workflow, there is no possibility of a comprehensive guide. With that in mind, the goal of this document is to walk you though the simplest means of getting your instrument working. After that, transference of knowledge or a bit of help from the community should be able to get you through most applications.

## The oneString Device




**Power LED**
Red light confirms power is supplied to the device

**Signal LED**
Lights green during initialization sequence and while sending signals to the computer. The normal initialization signal (when the oneString is first plugged in) is a single quick flash, followed by a pause, and then 3 flashes.

**Position Sensor**
Detects touch-finger position along its length to convey what note is being played

**Pressure Sensor**
Detects force of a finger press to read when and with what amount of volume a note plays

**Tuning Potentiometer**
Turns like a knob to change the overall instrument tuning

**Reset Button**
Pressing this button restarts the device firmware and flashes the initialization pattern again

**FTDI (USB-to-serial) Connector**
Connects via USB to a computer (flat side up, black wire toward the dip-switches)

## How it works

As simple as this device is, there is a complex chain of hardware and software which translates your input gestures (touching the oneString) to output (sound). Any one missing element breaks the entire chain. Here's the general system map:

**User > Sensors > oneString Firmware > USB-to-Serial (FTDI) Cable > Serial-to-Midi Bridge > Virtual Midi cable > Music Software > Audio Out**

I'll address these units in an order so that the signal flow and software installation is most easily understood.

**User**

See mirror for more detail

**Sensors**

The standard format of the oneString includes 2 main sensors:
a position sensor (the long one) that detects where along the length of the instrument your finger is.
A pressure sensor (the small round one) that detects how hard it is being pressed

The exact use of these and the other controls are explained below

**oneString Firmware**

The Arduino based firmware lives on the 28-pin Atmel IC in the middle of your oneString circuit. Its job is to read the sensors, dip-switch setting and tuning potentiometer, convert the signal to a useful MIDI data sequence and forward it to the computer software via the cable.

**USB-to-Serial (FTDI) Cable**

Once upon a time, your grand-daddy's computer had a metal port in the back referred to as a "serial communications port" or "COM port" which took simple data strings and conveyed them to software. These ports disappeared with the advent of "USB" (Universal Serial Bus) a universal replacement for older ports that can handle many data types including serial data. As Arduino microcontrollers and MIDI both have a heritage of using "serial" signals, the FTDI cable is used to wrap our midi signal into something modern computers can read. A tiny chip inside the cable handles this task, and passes the wrapped signal to the operating system.

These cables require a "virtual com port" driver to be installed on Windows, OSX, or Linux which is available here:
http://www.ftdichip.com/Drivers/VCP.htm in MANY flavors.
Users will need to know how the device is named on their specific system once this driver is installed. On Windows, check your 'Device Manager' for the newly installed Port (COM & LPT) to find the setting: something like "COM3." On OSX and Linux the installed driver should show up as something like "/DEV/TTY/USBSERIAL-FTxxx" in the software mentioned below.

When plugged into the circuit board, the cables flat side must be up with the visible black wire being closest to the dip switches and the green being closest to the tuning potentiometer. Plugging it in upside-down will not cause an explosion, but won't work.

**Virtual Midi Cable**

Using the oneString with most music software will require some sort of “Virtual Midi Cable” system to be installed or set-up on your computer before using the instrument. If you don't already use midi software, setup is somewhat different on various operating systems, and is quickly outlined here.

OSX based systems:

On OSX there is a built in system called IAC (Inter Application Communication) which can be found in Applications > Utilities > Audio Midi setup. Click on IAC Driver in the Midi Window, and verify that the “Device is online” box is checked. Under “More Information,” you can add additional midi “busses” or ports as needed. In other software these should show up as “IAC DRIVER MIDI 1” or whatever number your port is.

WIN based systems:

In Windows you'll need a third-party “Midi loopback device” installed on your system. There are several versions such as “LoopBe1,” Jamie O'Connell's “Midi yoke” and “Hubi's Midi loopback system”, found on the Internet with their respective installation instructions.

If you do not have a virtual midi system installed on your Windows machine, I've had great success on most flavors of Win with “Maple Midi” found at http://www.maplemidi.com/Maple_driver.html follow the instructions there for installation. Be sure to reboot when you're done regardless of previous experience. This software tricks Windows into believing you have a new piece of MIDI hardware attached to your computer; very clever!

Linux based systems:

This is a WIP, and Linux users are probably already used to doing things on their own. So, I will simply point you to the ALSA library's virtual midi card “VirMidi,” and in some cases “Jack” MIDI depending on your system and choice of software. I suspect Linux users with more experience than mine will be around the forum as the project grows.

**Serial-to-Midi Bridge**

Now that the signal is passed from the oneString through the FTDI cable and driver, a small piece of software is needed to “bridge” the serial data to your computer's virtual midi system. For all operating systems, I will recommend Angus Gratton's awesome “Hairless MIDI to Serial Bridge” located on github: http://projectgus.github.com/hairless-midiserial/

This piece of software is simple, convenient and actively maintained. That page has an excellent installation guide. Once installed a few specific settings are required for the oneString:
Be sure to setup the “serial port” pull-down with the name of your FTDI cable, and the “Midi Out” with the name of your first usable virtual midi cable, ex: “Maple Midi Out: Port 1”
In File > Preferences, the “Baud Rate” MUST be set to 38400 for the oneString controller. The other settings on that page are usually 8,None,1,None (all default).

**The First Test**

To verify function of the oneString unit and driver installation, we can do a silent test from Hairless. Plug in the oneString controller to the FTDI cable, and the FTDI cable to your computer's USB port before Hairless is opened. The red LED will light steadily. The green signal Led will flash once, pause, and flash 3 more times. Set the 4 dip switches to: OFF-OFF-OFF-OFF.
Then open Hairless, verify the correct “Serial Port”, “Midi Out” and Baud speed. Place a check in the box marked “Serial < - > Midi Bridge On.” Place a check in the box marked “Debug MIDI Messages” At this point pressing the pressure sensor should produce a string of messages, starting with a time stamp, “Serial IN: Ch 0” a note value and various velocity values. Pressing the Position sensor at the same time will change the “Note” number. If this works, it will simply be a matter of connecting the virtual midi to your music software. If it doesn't work, check the “troubleshooting” section below before moving on to more complex connections in the system chain.

**Music Software**

Almost all software synthesizers or hosts seem to have an input for midi controllers. If you're already working in software such as Renoise, Ableton, FL Studio, Jeskola Buzz, Reaktor, Reason, EnergyXT, Max-MSP, Puredata or the myriad of other software synths, hosts, and music studios out there, the setup should be pretty simple. If the FTDI driver and virtual midi cable is installed, and the serial-to-Midi Bridge is running, the oneString should appear to these software packages as a “Midi Controller” or an “External Midi Keyboard” via your selected virtual Midi Port. Each software package should have its own instructions on using a midi keyboard, that will be exactly the same for getting the oneString working with your software at this point.

Once a virtual synth is seeing your virtual midi cable, a final test can happen. Set the Tuning potentiometer so the flat side of the “knob” is facing down, and press the pressure sensor as above. A mid-range note should occur without needing to press the position sensor.

If no note sounds, and Hairless is showing data in the debug window, verify that your software is listening to MIDI channel 0 (zero) and the correct virtual port is selected. If Hairless is not showing midi data, but the LEDs light correctly on the instrument; verify FTDI driver installation and recheck the troubleshooting section below.

## Using the oneString

In general, think of the oneString as a two-handed instrument. Try this method first, and change it up as you get comfortable with the interface. Set the instrument dip switches to (OFF-OFF-OFF-OFF). On a table-top (or your lap), position the oneString with the circuit board to the right. Left hand can be used to activate the position sensor, and the right is used to activate the pressure sensor. It may be easiest to think of the device as a one-string guitar.

In most modes, the pressure sensor activates the sound (like plucking a string), while the position sensor allows note selection (like fretting a guitar). Touching the position sensor without pressing the pressure sensor will not produce a note. (If you fretted a guitar string without plucking it, no note would sound).

### The Pressure Sensor

The pressure sensor is only sensitive in the area of the gold circle. It may be easiest to start with your right index finger on the sensor with hand relaxed and fingers curved. Placing your right thumb against the side of the instrument can serve as an anchor, so that you can repetitively find the sensitive tap-spot without looking. After some time, you may be comfortable using other fingers, or even doing fast alternating 2-finger taps as playing styles are discovered.

### The Position Sensor

The position sensor is only sensitive along the 500mm length of the 12mm wide printed metallic stripe. Running your finger width way across a portion of the stripe will reveal a slight indented groove (or dent) between the layers of plastic in the sensor that runs the entire length. That is the sensitive sweet-spot.

If positioned as above, the left most position is the “bass” low-note, and the end closest to the circuit board is the highest note. Note that the note-steps are not equally spread across the length of the sensor. As you move toward the high notes, the notes are closer together than on the bass end. In this way the instrument emulates a stringed instrument, where frets are closer together at the high end. Note that the different modes allow more octaves along the sensor length making this spacing more evident as you get used to the instrument. As more users adopt the instrument, ideas on marking the octaves and notes of the device should be accessible on the wiki (www.straytechnologies.com/wiki).

### The Tuning Potentiometer

While tapping on the pressure sensor with you left hand, turn the tuning potentiometer left and right. This knob tunes the bass note of the instrument, so that the entire scale can be tuned to other instruments or the oneString can be used as a bass or treble instrument. Many music software packages have a virtual keyboard that shows what note is being played by the external midi instrument. Using this, the instrument can be precisely tuned to almost any of the 127 notes available on a full midi keyboard.

If you encounter note “bouncing” between two notes, give the potentiometer a slight “tweak”, to prevent it from seeing 2 notes at once.

### The Dip-Switches

The various style modes on the onestring controller are controlled by the 4 dip-switches on the circuit. The styles are broken into 4 major modes (controlled by switches 1 & 2), and variations of these modes (controlled by switches 3 & 4). The switches are read as binary and have the following settings and attributes:

| **Mode** | **Switches 1-2---3-4 (ON is toward the LEDs)** | **description** |
|---|---|---|
| Retrigger Mode | OFF-OFF---ANY-ANY | ⇧ Each new position sends a new midi noteOn message.<br>⇧ First Pressure trigger sends a new midi noteOn message.<br>⇧ Bass note is open and tunable |
| (0---0) | OFF-OFF---OFF-OFF | Retrigger Mode – one octave |
| (0---1) | OFF-OFF---OFF-ON | Retrigger Mode – two octave |
| (0---2) | OFF-OFF---ON-OFF | Retrigger Mode – four octave |
| (0---3) | OFF-OFF---ON-ON | Retrigger Mode – full midi range<br>( 127 notes minus bass note value) |

| **Mode** | **Switches 1-2---3-4 (ON is toward the LEDs)** | **description** |
|---|---|---|
| Pitch Bend Mode | OFF-ON---ANY-ANY | ⇧ First Pressure trigger sends a new midi noteOn message.<br>⇧ Each new position sends a noteBend message on the last triggered note<br>⇧ Bass note is open and tunable |
| (1---0) | OFF-ON---OFF-OFF | Pitch Bend Mode – one octave |
| (1---1) | OFF-ON---OFF-ON | Pitch Bend Mode – two octave |
| (1---2) | OFF-ON---ON-OFF | Pitch Bend Mode – four octave |
| (1---3) | OFF-ON---ON-ON | Pitch Bend Mode – full midi range<br>(127 notes minus bass note value) |

| Mode | Switches 1-2---3-4 (ON is toward the LEDs) | description |
|---|---|---|
| One-Hand Retrigger Mode | ON-OFF---OFF-ANY | ⇧ Each new position sends a new midi noteOn message.<br>⇧ Bass note is tunable, but not open<br>⇧ Pressure Sensor sends Modulation (Mod Wheel) Data MID_CC1 |
| (2---0) | ON-OFF---OFF-OFF | One-Hand Retrigger Mode – two octave |
| (2---1) | ON-OFF---OFF-ON | One-Hand Retrigger Mode – four octave |
| One-Hand Pitch Bend Mode | ON-OFF---ON-ANY | ⇧ First Non-Zero Position trigger sends a new midi noteOn message.<br>⇧ Each new position sends a noteBend message on the last triggered note<br>⇧ Bass note is tunable,but not open<br>⇧ Pressure Sensor sends Modulation (Mod Wheel) Data MID_CC1 |
| (2---2) | ON-OFF---ON-OFF | One-Hand Pitch Bend Mode – two octave |
| (2---3) | ON-OFF---ON-ON | One-Hand Pitch Bend Mode – four octave |

| Mode | Switches 1-2---3-4 (ON is toward the LEDs) | description |
|---|---|---|
| MIDI CC Mode | ON-ON---ANY-ANY | ⇧ No Note information Sent<br>⇧ Position sensor and Pressure Sensors send MIDI CC data only<br>⇧ Tuning Pot sends MIDI CC data |
| (3---0) | ON-ON---OFF-OFF | ⇧ Position Sensor send MIDI CC 20<br>⇧ Pressure Sensor sends Modulation (MIDI CC 1)<br>⇧ tuning knob sends MIDI CC 21 |
| (3---1) | ON-ON---OFF-ON | ⇧ Position Sensor send MIDI CC 20<br>⇧ Pressure Sensor sends MIDI CC 21<br>⇧ tuning knob sends MIDI CC 22 |

| Mode | Switches 1-2---3-4 (ON is toward the LEDs) | description |
|---|---|---|
| (3---2) | ON-ON---ON-OFF | RESERVED – NO SIGNAL SENT |
| Classic Ribbon Mode<br><br>(3---3) | ON-ON---ON-ON | ⇧ Position Sensor sends Pitch Bend Data only<br>⇧ Pressure Sensor sends Modulation (MIDI CC1)<br>⇧ Tuning Knob sends Volume (MIDI CC7)<br>⇧ Your associated keyboard must be on MIDI Channel 0 to trigger notes |

NOTES:

- ⇧ Everything happens on MIDI channel 0 (zero)
- ⇧ “Octaves” as defined here include one additional note above ex: C3 to C4 is one octave (13 notes), C3 to C5 is 2 octaves (25 notes), etc...
- ⇧ There are only 127 possible midi notes. If the open-bass note is tunes to midi note 120, there will only be 8 possible notes, regardless of variation setting.
- ⇧ The sensor is not “linear” so higher octaves take less physical length than lower octaves.
- ⇧ Similarly, the Classic Mode's position “Center” is not in the sensor's middle, as the sensor is not “linear” in the way it senses and sends data. “Center” is closer to 1/3 distance from the Circuit Board.
- ⇧ Some synths may sound like they're “stepping” in bend mode, especially in lower octaves. This is a result of resolution, as the Arduino/sensor combination has less possible range than pitch bend in the midi specification (10 bit resolution vs. 12 bit). Many synths resolve this in software with a fast “portamento,” but reducing the pitch-bend range on the synth will also smooth the stepping.
- ⇧ Pitch-Bend range (in octaves) is entirely controlled by the software synth, and can not be changed on the oneString. This is true of all midi controllers following the MIDI spec.

## Troubleshooting

I have set up a forum at : http://www.straytechnologies.com/wiki/tiki-forums.php
for help or contribution within the community of users. If you're at a loss on how to get your device humming, feel free to ask the list after acquiring some minimal information: what OS, OS version, software, and symptoms are you seeing? Does the basic setup above work first? Be as descriptive as possible, and it will be easier for folks to chime in and help.

To help determine the cause, the checks below can also make troubleshooting a bit easier, by checking each part of the system individually in this order.

Is the red power LED on? No-- Check FTDI cable, connection (6-pin and USB side)

Does the green signal LED light when the pressure sensor is pressed? No-- check sensor connections, re-check initialization blink for firmware errors.

Does the correct FTDI cable label show in Hairless MIDI? No, check cable connection and driver installation.

Does Hairless report MIDI data in the debug window? No, verify Baud speed and other settings in the software preferences.

Does the correct virtual MIDI cable show up in Hairless AND in your music software? No, check virtual midi installation.